UNIVERSITAT POLITÈCNICA DE CATALUNYA

DEPARTAMENT DE FÍSICA APLICADA

PAYLOAD DATA HANDLING, TELEMETRY AND DATA COMPRESSION SYSTEMS FOR GAIA

JORDI PORTELL I DE MORA

THESIS SUBMITTED FOR THE DEGREE OF

DOCTOR IN PHILOSOPHY

ADVISORS:

ENRIQUE GARCÍA-BERRO MONTILLA

XAVIER LURI CARRASCOSO

Barcelona, June 2005

Per la Montse,

per fer-me sentir viu.

Agraïments / Acknowledgements

Durant gairebé cinc anys he tingut la sort no només de poder treballar en un projecte apassionant, sinó també de poder-ho fer amb uns col·laboradors excel·lents en tots els aspectes –sense oblidar a tots els amics que m'han acompanyat durant aquest període i me l'han fet molt més agradable.

During almost five years I have been so lucky not only for working in such an exciting project, but also for doing it with an excellent team of collaborators –without forgetting all the friends that have accompanied me during this period and have made this much more pleasant.

En primer lloc vull agrair amb tot el meu cor als millors directors de tesi que hom podria desitjar. A Enrique, por no tener bastante con soportar durante todo un Proyecto Final de Carrera a este pesado que os escribe. I a en Xevi, per obrir-me les portes a aquest gran projecte i acostar-me als seus responsables. Tots dos han vetllat per mi des del primer moment i m'han estat ajudant, guiant i revisant la feina, per molt ocupats que estiguessin. Sense ells aquesta tesi no seria el mateix.

Gràcies també al Jordi Torra, per haver-me inclòs a l'Equip Gaia i pel seu recolzament (moral, professional i material) i confiança des del primer dia.

I moltes gràcies al Jordi Isern, per acollir-me a l'IEEC durant quatre fantàstics anys i recolzar-me en tots els meus passos. Per mi ha sigut un autèntic plaer treballar a l'Institut amb ell i amb tots els altres companys.

Gràcies a l'Eduard Masana qui, junt amb en Xavier Luri, m'ha suportat en les meves interminables consultes sobre GASS i el model de telemetria. A la Carme Jordi, per la seva ajuda amb els sistemes de referència i l'MBP. A la Cesca Figueras i a tot l'Equip Gaia de Barcelona, per la seva ajuda tant professional com personal, i als meus nous companys de feina (Dani, Francesc, Pablo, Josep Manel i Guillem), per fer-m'ho passar tant bé. Muchas gracias a Claus Fabricius por leer este tocho y ayudarme en los últimos trámites de la tesis.

À tous mes amis dans le bâtiment de Hipparque à Meudon: Anique, Daniele, Anita et beaucoup d'autres, pour leur accueil chaleureux pendant mon premier séjour de trois semaines et pour tous les autres temps que j'ai été là-bas.

Très spécial grâce à Frederic Arenou, pour accueillir m'à Meudon et me dirigeant dans mes premières étapes dans Gaia, et aussi pour suggérer l'étude de PDHS et m'aidant avec cela. Merci aussi à Carine Babusiaux pour être un guide de Paris excellent, et pour réviser (ensemble avec Shan Mignot) l'intégration correcte de mon traite PDHS avec le simulateur GIBIS et le travail fait à Meudon dans ce champ.

To all my friends in the Gaia Team at ESTEC, for their warm welcome during my stay and for their help in many issues related to Gaia and this work: Karen O'Flaherty, for her excellent work with the Proceedings of the Symposium and all the Gaia documentation in general; Jos de Bruijne, for his help on several issues on clocks, timing, and the parameter database; Alex Short, for his help on specifications of the CCDs and radiation issues; and Salim Ansari, for his friendliness and continuous encouragement, and for providing me with the large computational power of GaiaGRID. Together with Mark T. Linden, he offered me an excellent assistance and let me overload and crash the system so many times...

Very special thanks go to Michael Perryman, who has appreciated and encouraged my work since the very beginning. Hearing him saying that he was proud of me has been the best reward.

A very big thank you to Uwe Lammers, for his kind help and attentions during my three months at ESTEC, and for his help in several issues related to this work, including the study of improved contact times with the ground station, the science telemetry, the timing schemes, and C++ coding issues (as well as the European Ph.D. report).

A mis amigos de Groenhazengracht (Marcos, Itziar y Berta), y a Roberto, Josevi, Miguel, Jaime, Oriol, María, Ester, Lisa, Jordi, Ruth y tantos otros, por hacerme pasar tan buenos ratos y hacerme sentir como en casa.

To Lennart Lindegren for his kind help with the European Ph.D. report and on the definition of reference systems and timing schemes. To Uli Bastian for his invaluable comments and corrections on these same issues. Jean Kovalevsky also kindly reviewed my work on reference systems and provided crucial comments and corrections. Also thanks to Frederic Safa at Astrium and Torgeir Paulsen in the Gaia Project Team, who reviewed my work on the PDHS of Gaia.

Durant aquests anys he tingut el privilegi i el plaer de ser co-director de quatre grans Projectes Finals de Carrera. L'Enrique Martin Geijo va fer un excel·lent estudi sobre el canal de comunicacions de SIXE, ens va donar un cop de mà en la seva possible aplicació a Gaia, i va notificar-nos dels possibles problemes que els errors de transmissió poden provocar en certs esquemes de compressió. El Javier Castañeda va desenvolupar un excel·lent simulador del sistema de rellotges de Gaia, i em va ajudar en alguns aspectes de programació en C++. En José Pérez Gordo ha continuat aquest estudi dels rellotges obtenint-ne resultats molt bons i detallats; uns resultats que són primordials per fixar els requeriments de la càrrega útil de Gaia. El Ricard Castro ha fet un gran treball millorant el meu Telemetry CODEC, fent les dades de GASS més realistes i incloent-hi la conversió a l'estàndard de telemetria de la ESA –a més d'incloure-hi el meu esquema adaptatiu, cosa que no era gens trivial.

En quan a l'aspecte material i econòmic, cal agrair a un munt de gent i entitats la gran ajuda que m'han ofert. Puc ben assegurar-vos que sense ells no s'hauria pogut dur a terme aquesta tesi:

- *Generalitat de Catalunya / AGAUR, per la beca 2001FI 00715, que m'ha donat de menjar durant quatre anys i m'ha pagat l'estada de tres mesos a ESTEC.*
- *Al Departament de Física Aplicada de la UPC i l'Enrique García-Berro, per pagar-me viatges a reunions i congressos; gràcies a:*
    - *MCYT, Plan Nacional de Astronomía, Estadios avanzados de la Evolución Estelar: Teoría y Aplicaciones a Escala Galáctica, AYA2000-1785*
    - *MCYT, Plan Nacional de Astronomía, La población de enanas blancas en el halo galáctico, AYA2002-04094-C03-01*
    - *MCYT/Max Planck Gesellschaft, Acción Integrada Hispano-Alemana, La función de luminosidad de las enanas blancas como trazadora de la evolución de la Galaxia, HA2000-0038*
- *Al Departament d'Astronomia i Meteorologia de la UB i en Jordi Torra, per solucionar-me els primers mesos de doctorat (quan encara no tenia beca), pagar-me l'estada de tres setmanes a Meudon i ajudar-me en alguns viatges; gràcies als projectes:*
    - *MCYT, ESP1997-1803*
    - *MCYT, ESP2001-4531-PE*
    - *MCYT, ESP2003-04352*
- *To ESTEC, for accepting me as a trainee during three months.*

En l'àmbit personal, no tinc paraules per agrair el suport que he arribat a tenir de tots els amics i de la família. Molt especialment de la Montse, que ha experimentat més que ningú les meves estones d'estrès i canvis d'humor durant les temporades de més feina. Ella sempre m'ha animat a tirar endavant, tant si jo era al seu costat com si era a Canàries, França, Holanda o qualsevol altre lloc.

Moltes gràcies a l'Isidro, no només per la seva gran amistat sinó també per suportar les meves discercions tècniques quan anava atabalat; fins i tot va repassar la meva feina sobre el PDHS des del seu punt de vista de Doctor en Electrònica... El meu bon amic Juanjo també ha hagut de patir interminables paranoies doctorals; i voldria donar les gràcies també a en Jaume, en Gil i en Toni, per ser tan bons companys de pis. També, molt especialment, a en Miguelón (el de Monzón de Aragón, encara que ja no treballi a Retevisión!). Perquè sí. Perquè no canviï mai.

Ja fa gairebé 12 anys que vaig marxar de Torelló per anar a estudiar a Barcelona, i encara que durant el doctorat he fet bastantes escapades a casa (incloent tot un any treballant des d'allà), he

hagut d'estar massa temps lluny de la Plana... Per sort els bons amics no es perden mai. Per això vull fer una menció especial als meus grans amics de Torelló. Començant per en Pere Moreno, amb qui desafortunadament no ens hem pogut veure tant com volíem, cosa que ens ha fet deixar ja massa enrere un apassionant tema que teníem entre mans pràcticament des que anàvem a l'escola. Junt amb en Marc Ordi, tots tres hem passat unes estones genials; espero que sempre més puguem anar trobant forats a les nostres agendes per reviure-ho. Amb en David Moreno, l'Albert Pijoan i en Jesús Frutos també ens ho hem passat de conya.

Gràcies també a tot l'equip SETI@home-Catalunya, per la seva amistat i per donar-me ànims amb la tesi, i també per la seva ajuda i comprensió durant les llargues temporades en què no em podia dedicar gaire a aquest altre gran projecte –possiblement més gran del que qualsevol de nosaltres ens puguem imaginar. Estic segur que ens esperen uns anys apassionants i amb moltes sorpreses.

A en Jean Michel, en Mike, en Hans i en Lebo; a en Bob, la Dido, en Moby, en Dru i en Lenny; a la Nelly, la Sophie i la Kylie; a en Joe, en Rickster i els germans de la jungla; a la 303 i els seus botons, al Technics, a l'Adeva i al Farley Jackmaster Funk... i a tants altres que m'han animat i posat la pell de gallina durant les llargues estones invertides en aquesta tesi.

Als meus pares. No sé ben bé quin motiu donar, n'hi ha tants... M'he esforçat a intentar recompensar-los d'alguna manera tot el que han fet per mi, però em sembla que encara queda bastant camí per recórrer. Espero que hi siguin durant molts i molts anys més per poder-lo fer junts. I als meus germans i germanes, cunyades i "cuñaaaaaaos", i a tota la família Portell. Per ser tant genials.

«Una vez oí el relato de un hombre
que se dividió en dos.
Una parte nunca cambió;
la otra creció y creció.
La parte que no cambió siempre fue fiel,
la parte creciente siempre fue nueva;
y yo me pregunté, cuando terminó el relato,
qué parte era yo y qué parte eras tú.»

Orson Scott Card

Realment han sigut cinc anys inoblidables...

Jordi

Table of Contents

Introduction

Presentation

Gaia is the new astrometric mission of the European Space Agency. With a scheduled launch in 2011, Gaia will observe more than one billion stars and other objects with unprecedented accuracy, providing a sample of more than 1% of the stellar content of our Galaxy. Its ambitious objectives completely outperform the competing missions of other agencies. At the end of its lifetime, the largest and most complete three-dimensional map of our Galaxy will be produced.

Such a mission implies large technological and design efforts, since it will have to detect, select and measure hundreds of stars every second, sending their data to the Earth –more than one million and a half kilometers away. We have focused the work of this thesis on some important aspects of the mission, namely, we have proposed designs for the payload data handling system of Gaia, its science telemetry, and its data compression system. Our final goal was to make possible the transmission to the ground station of the large amount of data generated by the on-board instruments, taking into account the limited capacity of the communications channel. This required the design of a lossless data compression system offering the highest possible compression ratios and guaranteeing at the same time the integrity of the transmitted data. All in all poses a great challenge for the information theory methods and the design of data compression systems.

Despite the large amount of teams working on the Gaia project, these technological issues were still untouched or only preliminary drafts were available at the beginning of this thesis, as Gaia itself was on a preliminary stage. Therefore, our work has been received with enthusiasm by scientists and engineers of the mission. As a first point, before starting the design process, the mission and its payload elements had to be analyzed: the design of such an optimal data compression system requires the knowledge of which data will be generated, which format will they have and in which way they must be fed to the communications system in order to fulfill the adequate standards. Thus, the first issue to review is the operational environment of our study, described in the first part of the thesis. It includes the several reference systems and conventions that we have proposed in order to unify the measurements, references to data and designs. This proposal has been used as an initial reference in the mission and is currently being extended and improved by other scientists. This first part also lists the main features of the astrometric instrument (Astro) as well as its main operational guidelines, which have also been taken into account by other teams.

The second part of the thesis deals with the payload data handling system of Gaia. There we describe our proposal for this system which has been used to present the scientific requirements to the industrial teams, and constitutes in itself a viable (although simplified) implementation option. In the next part we study the science telemetry, compiling the data fields to be generated by the instruments and proposing an optimized codification and transmission scheme –also being used by other teams of the mission as the base for further developments. This design reduces the occupation of the communications channel and is ready to include an optimized data compression system. The latter will be described in the fourth and last part of the thesis, where we will see how the compression requirements are almost completely fulfilled by our proposal, offering twice the ratios achieved with the best standard compression systems. Therefore, our design represents the best solution currently available for Gaia, and its performance has been taken as the baseline by other teams.

Finally, we must note that the results of our work extend beyond the release of this thesis document, complementing it with a set of software applications that have helped in designing, optimizing and verifying the operation of the systems proposed here. It is worth noting that the complexity of our work has been increased due to the need of continuously updating it to the changes that the mission has suffered in its design during the four years of the Ph.D. To conclude, we can say that we feel satisfied with our results, since most of them have been (or are currently being) taken into account by many teams involved in the mission and by the European Space Agency for the final design.

Presentació

Gaia és la nova missió astromètrica de la Agència Espacial Europea. Amb un llançament programat pel 2011, Gaia observarà més de mil milions d'estels i altres objectes amb una exactitud sense precedents, el qual representa més d'un 1% del contingut estel·lar de la nostra Galàxia. Els seus ambiciosos objectius desbanquen completament les missions rivals d'altres agències. Al final de la seva vida útil es generarà el major i més complert mapa tridimensional de la nostra Galàxia.

Una missió d'aquestes característiques suposa grans esforços tecnològics i de disseny, ja que caldrà detectar, seleccionar i mesurar centenars d'estels cada segon, per enviar-ne posteriorment les dades cap a la Terra –a més d'un milió i mig de quilòmetres. Hem centrat el treball d'aquesta tesi en aquesta vessant de la missió, proposant dissenys pel sistema de gestió de dades dins la càrrega útil de Gaia, per la seva telemetria científica, i pel seu sistema de compressió de dades. El nostre objectiu final és fer possible la transmissió a l'estació terrestre d'aquesta immensa quantitat de dades generades pels instruments, tenint en compte la limitada capacitat del canal de comunicacions. Això requereix el disseny d'un sistema de compressió de dades sense pèrdues que ofereixi les millors relacions de compressió i garanteixi la integritat de les dades transmeses. Tot plegat suposa un gran repte pels mètodes de la teoria de la informació i pel disseny de sistemes de compressió de dades.

Tot i el gran nombre d'equips treballant pel projecte Gaia, aquests aspectes tecnològics encara estaven per estudiar o bé només es disposava d'esborranys preliminars –ja que la missió mateixa estava en una etapa preliminar en quan varem començar aquesta tesi. Per tant, el nostre treball ha estat rebut amb entusiasme per part de científics i enginyers del projecte. En primer lloc, abans de començar el procés de disseny, cal analitzar la missió i els seus elements de càrrega útil: per dissenyar el sistema òptim de compressió de dades cal saber quines dades es generaran, quin format tindran i de quina manera s'han d'enviar al sistema de comunicacions per tal de complir els estàndards adequats. Així doncs, el primer aspecte a revisar és l'entorn operacional del nostre estudi, descrit a la primera part de la tesi. Això inclou els diversos sistemes de referència i les convencions que hem proposat per tal d'unificar les mesures, referències a dades i dissenys. Aquesta proposta s'ha utilitzat com a referència inicial en la missió i actualment altres científics l'estan ampliant i millorant. Aquesta primera part també mostra les principals característiques de l'instrument astromètric (Astro) així com les seves directrius operacionals, el qual també s'ha tingut en compte en altres equips.

La segona part de la tesi ens durà ja al sistema de gestió de dades de la càrrega útil de Gaia. Aquí descriurem la nostra proposta per aquest sistema, la qual ha estat utilitzada per presentar els requeriments científics als equips industrials i representa en sí mateixa una opció d'implementació viable (tot i que simplificada). En la següent part estudiarem la telemetria científica, recopilant els camps de dades a generar pels instruments i proposant un esquema optimitzat de codificació i transmissió –també utilitzat per altres equips de la missió com a base per nous desenvolupaments. Aquest disseny redueix la ocupació del canal de comunicacions i està preparat per incloure un sistema optimitzat de compressió de dades. Aquest darrer serà descrit a la quarta i última part de la tesi, on veurem com la nostra proposta compleix gairebé totalment els requeriments de compressió, arribant a duplicar les relacions de compressió ofertes pels millors sistemes estàndard. Per tant, el nostre disseny representa la millor solució actualment disponible per Gaia, i el seu rendiment ha estat assumit com a disseny base per altres equips.

Finalment, cal dir que els resultats del nostre treball van més enllà de la publicació d'aquesta memòria de tesi, complementant-la amb un conjunt d'aplicacions de software que hem desenvolupat per ajudar-nos a dissenyar, optimitzar i verificar la operació dels sistemes aquí proposats. També cal indicar que la complexitat del nostre treball ha estat augmentada degut a la necessitat d'actualitzar-lo contínuament als canvis que la missió ha sofert en el seu disseny durant els quatre anys del doctorat. Per acabar, podem dir que estem satisfets amb els resultats del nostre treball, ja que la majoria han estat (o estan essent) tinguts en compte per molts equips involucrats en la missió i per la mateixa Agència Espacial Europea en el disseny final.

Chapter 1

Introduction

1.1. THE GAIA MISSION

Gaia is the most ambitious astrometric space mission currently envisaged, adopted within the scientific programme of the European Space Agency (ESA) in October 2000. It aims to measure the positions and proper motions of an extremely large number of stars and other types of objects with unprecedented accuracy, complemented with multi-band photometry and spectrometry. As a result, the most complete and accurate three-dimensional map of our Galaxy will be obtained, also including Solar System objects and extragalactic sources. This space observatory will be a technological challenge in all its aspects, from its instrumentation and on-board data handling to the on-ground data processing and analysis.

Gaia is the successor of Hipparcos, the first astrometric satellite, operated also by ESA from 1989 to 1993 (ESA 1997). Hipparcos was a very successful space mission, and some of its operating principles have been adapted to Gaia. Other space missions similar to Gaia – such as OBSS [IR.3.] and JASMINE [IR.11.] – have been proposed, but are still under consideration. SIM [IR.8.], measuring a limited number of stellar sources, has already been approved, but some others – like DIVA [IR.10.] and FAME [IR.13.] – have been already discarded.

1.1.1. General features

With a planned launch in 2011, Gaia will orbit around the Sun-Earth L2 lagrangian point, 1.5 million kilometers from the Earth opposite to the Sun. The launch vehicle will be a Soyuz rocket with an additional Fregat stage, which will insert the satellite into its transfer orbit towards the L2 point. After some 4 months, Gaia will maneuver to enter the final Lissajous orbit around L2, where it will operate for about 5 years of mission lifetime.

The operation of Gaia is based on a continuous all-sky scanning. The satellite will spin around its own axis, with this axis maintaining a fixed angle with respect to the Sun while at the same time also performing a precession motion. Although every astronomical object measurable by Gaia will be observed several times during the mission, this scanning law will lead to a non-uniform sky coverage. The observations will be made in the visible spectrum and will include position and proper motion of the objects measured, as well as photometric and spectrometric data. Two telescopes will perform the astrometric and broad-band photometric measurements, while a third telescope will be in charge of the spectrometric and medium-band photometric measurements.

A complex, high-performance on-board payload data handling system will perform the adequate operations to retrieve the data from the instruments, preparing them for being transmitted to ground. This transmission will happen only during 8 hours a day on average, when Gaia will have direct visibility from the Cebreros ground station in Spain. At this point, the data will be transmitted to the data processing center, where they will enter an extremely complex process of data reduction. First, a preliminary cross-matching process and a fast-look analysis will be performed, feeding the data afterwards to the Gaia data base. The Global Iterative Solution (GIS) process will then start, needed to obtain the final Gaia catalogue. It can be considered as one of the most complex processing systems of our times, with a data base size estimated to be around 1 to 2 petabytes (PB), and the data processing needs in some zettaflop (10^{21} FLOPs). All in all justifies that this system is sometimes compared to the Human Genome project. This huge complexity has required a preliminary study, the Gaia Data Access and Analysis Study (GDAAS) to prove its viability. Its second phase, led by the

University of Barcelona, the Supercomputing Center of Catalonia and a Spanish software company (GMV), has finished during this year with satisfactory results.

The overall budget of Gaia is about 500 million euros, and the mission is currently finishing its Phase B1 (and, therefore, the Definition Phase). The Implementation Phase should be started during this year with the Design Phase (B2) which should last for about one year. It is worth to note that Gaia suffered a redesign during 2002 in order to reduce costs, which implied major changes in its design –such as the combination of the two astrometric fields of view into a single focal plane, rather than using two independent focal planes. Fortunately, this redesign maintained almost intact the scientific capabilities of the mission.

Figure 1.1 (taken from [IR.12.]) illustrates how the Gaia satellite will look like, including the payload module (PLM), the service module (SVM), the sunshield, the active antenna and the FEEP thrusters. The latter will correct the attitude of the satellite with a thrust of some millinewtons (mN). Also, the antenna has been designed as an electronic array of antennae, thus avoiding any kind of parabolic antenna which would require a continuous pointing towards the Earth and, therefore, unacceptable mechanical perturbations.

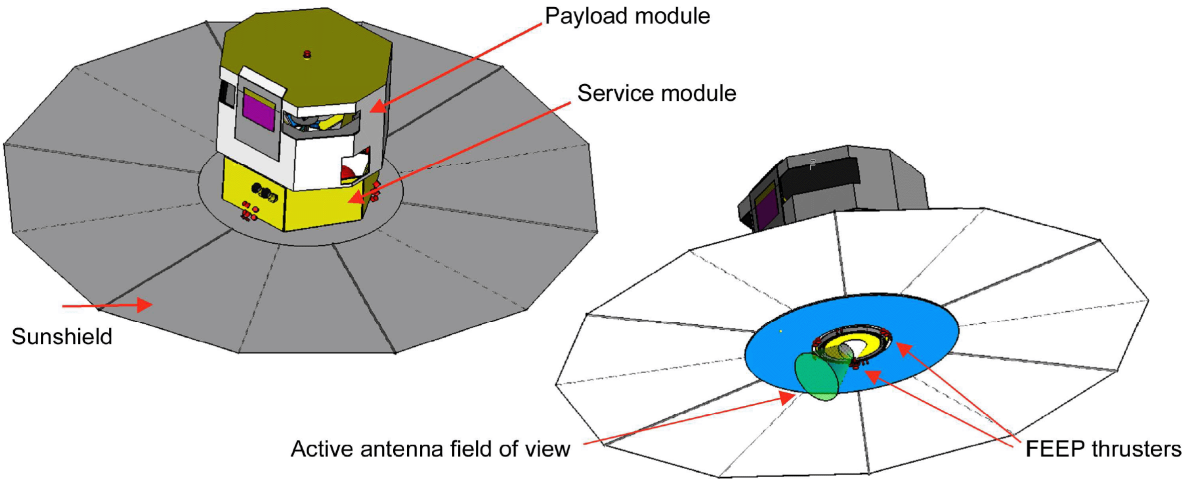


Figure 1.1: Overview of the Gaia spacecraft

1.1.2. Scientific objectives

The observations of Gaia will not be based on any input catalogue, as it happened with Hipparcos, but it will rather detect and measure the stellar objects autonomously without any *a-priori* selection. Due to its spin motion, all the Gaia telescopes will project the stars with an almost linear motion over the focal planes. Image detection systems will be implemented on-board, making possible the selection and measurement of any star (or astronomical source, in general) down to a limiting magnitude of about 20. Owing to this excellent sensitivity, about 1.2 billion objects will be measured, which represents about 1% of the stars of our Galaxy. This, enriched with the several characteristics measured for every star, will reveal the structure and kinematics of the Milky Way.

The state-of-the-art instrumentation of the satellite, together with complex on-ground algorithms for data reduction, will make possible to obtain the positions and motions of the objects with unprecedented accuracy. Taking into account the original design, at 15^{th} magnitude Gaia will measure parallaxes with an accuracy of about 10μas (microarcseconds), which is equivalent to the diameter of a human hair seen from 2 thousand kilometers. At the limiting magnitude the accuracy will be about 160μas, while at 10^{th} magnitude it will be about 4μas. These values have slightly changed due to the latest redesigns of the spacecraft. The tangential velocities will also be measured by the astrometric system with an extremely high accuracy, while the radial velocities will be obtained from the spectrometric instrument –thanks to the shifts in the spectral lines. The final accuracies for both velocities will range between 1 and 10 km/s [IR.12.].

Gaia will not only measure stars, but also any astronomical object with enough apparent brightness to be detected and observed by its instruments. Millions of binary stars, some hundred thousand white dwarfs, more than 50 thousand brown dwarfs and millions of galaxies will be measured, as well as thousands of solar system objects including NEOs (Near-Earth Objects). Also, about 10^5 supernovae and 30 thousand extra-solar planets are envisaged to be catalogued, as well as some 500 thousand quasars and some 200 gravitational microlensing events.

1.1.3. Payload features

The Gaia payload is basically composed of:

- Two astrometric telescopes combined onto a single focal plane.
- One photometric and spectrometric telescope projecting its central part onto a radial velocity spectrometer, and its outer parts onto a medium-band photometric focal plane.
- Additional systems such as a basic angle monitoring device (BAMD) or a wide field star tracker. The latter will surely be physically placed in the SVM, as a part of the AOCS. The BAMD is extremely important since the astrometric operation is based on an excellent knowledge of the angle between the two astrometric telescopes, and vibrations occurred during the launch phase will probably change this angle wrt its ground nominal value.
- A payload data handling system (PDHS), which will also include a multi-priority storage system. The communications system will transmit the science data to the ground station at about 4 Mbps during about 8 hours a day.

We shall note at this point that the focal planes of Gaia, all of them based on CCDs (charge coupled devices), require a special operating mode due to the spin motion of Gaia. Their operation is based on TDI (time delayed integration), which synchronizes a charge shift (from one pixel line to the next) with the satellite rotation. In this way, images with long integration times can be continuously acquired with a small blur. This operation mode, as well as many other features of the payload, will be better explained later in this document. We will focus our work on the payload module of Gaia, and more specifically on the astrometric instrument (abbreviated as Astro).

1.2. REQUIREMENTS OF THE DATA MANAGEMENT SYSTEMS

The payload data handling system (PDHS) of Gaia is one of the key challenges of this mission, since it will have to detect, select and measure some hundred sources every second, compress their data, and store them on board until the next contact with the ground station. The system will obviously have to keep measuring sources while the data are transmitted to the ground, which implies concurrent operations. Several priorities, both for the sources in the instruments and for the measured data, shall be used in order to always fill the downlink channel with the most important science data. The importance of this data handling system is such that if not all the predicted sources can be measured (or transmitted), the astrometric budget will be affected and, therefore, the final catalogue will have degraded accuracies (and completeness).

1.3. PURPOSE OF OUR WORK

It is important to emphasize that our work is focused on the Astro instrument, although we may comment a few details on the overall operation of the medium band photometer (MBP) or radial velocity spectrometer (RVS) instruments. Some of our designs or ideas could be applied not only to Astro but also to the MBP or RVS with a few modifications. Finally, part of our

work can be considered common to all of the instruments, such as the PDHS or the data transmission system.

In order to better explain our work, we will describe some elements of the spacecraft that have been designed or proposed by other science or project teams. For example, a general operation of the PDHS had already been proposed by Astrium, but we have refined and improved its design and operation. The following subsection describes which are our main areas of work, as well as their status at the beginning of our study.

1.3.1. Main objectives

Our most important objective is to make possible the reliable transmission of all the necessary science data from Gaia to the ground station, with the adequate accuracy and priorization. In order to achieve this, the following items should be previously studied:

- Definition of a set of reference systems and conventions, in order to unify the references to components of the spacecraft, measurements and data in general. This item is specially important since, at the time of beginning of this work (end 2000), many Gaia working groups already existed and the need of a uniform set of references was fundamental. We undertook this work, trying to offer this set of tools not only for our work but also for the Gaia community in general.
- Identification of the operative environment, including the various mission parameters, its main features and the operation of its instruments. Part of this work was already done, although a handy guide with a compilation of the most important values was also useful for the Gaia community. Another part of this work, specially related to the operation of the Astro instrument, still needed much effort.
- Definition of the data to be transmitted and their formats. This work had also been done in part, but a general review was needed so it was undertaken by us in cooperation with the GDAAS team at the University of Barcelona. The reason is that they are the developers of GASS (the Gaia System Simulator), which outputs the data as science telemetry and, therefore, requires an adequate definition for the data fields.
- Design of an optimized and concurrent PDHS, capable of covering all the processing needs of the instruments. This had already been done in a simplified way. Consequently, several updates and improvements were required, as well as a refinement of the scientific requirements.
- Definition of optimal codification and transmission schemes for the science data. Our intention was to provide optimized schemes, beyond any other standard scheme that had already been designed.
- And finally, to design an optimal data compression system. This field of work was still open, without any proposal for the Gaia science data. Thanks to our previous experience on this field (Portell 2000), we were able to provide useful ideas and designs, which should make possible to fulfill most of the data compression requirements. The refinement of its design was also part of our work.

1.3.2. Work plan

In order to reach the objectives listed in the previous section, we decided to split our work down into several “work packages”, each for a given field of interest. Each of these packages, which have been translated into this document as “parts”, contains sub-packages which are described in the chapters of each part:

- Definition of reference systems, also including the compilation of parameters and characteristics of Gaia and its instruments, as well as a proposal for the operation of the

astrometric instrument. The first part of the thesis is devoted to this *Operating Environment*.

- The *Payload Data Handling System* (PDHS), which is important enough to define a part just for this issue. The only chapter in this part will present its overview and system design, as well as the main data flux between its modules and their main operational guidelines. A detailed proposal for the operation of some subsystems corresponds to other work packages of our study –such as telemetry and data compression.
- Part III is devoted to the *telemetry and communications* of Gaia, including a telemetry model definition which was defined in coordination with the Gaia team at the University of Barcelona. Also, we propose a new, completely optimized scheme for the codification and transmission of Astro science data, fulfilling ESA telemetry standards. We were interested in studying the communications channel of Gaia as well, so we also recommend some improvements on its operating parameters. Finally, this part will describe a powerful simulation tool developed by the author, the *Telemetry CODEC*, which makes possible to do complex studies and transformations to Gaia simulated data and will be needed to test the data compression system.
- The last part of the thesis shall bring us to the final target of our work, which was to transmit all of the science data from Gaia to the ground station. The definition of an optimal *Data Compression* system is done in two chapters, starting with some preliminary definitions which, after being tested, bring us to the optimized data compression system.

It is important to note that our work should be considered as a set of proposals presented to ESA, rather than final designs of the mission. Nevertheless, our intention from the very beginning was to develop these proposals with enough reliability and realism to be considered as actually feasible within the mission, so that ESA and its industrial partners could take advantage of them for the final design. We can proudly state that this has been the case for many of them.

Part I:
Operating environment

Chapter 2

Reference Systems

This chapter contains a proposal for the set of reference systems and conventions for the instruments of Gaia, and describes their relationship with other standard reference systems like the ICRS (International Celestial Reference System). We will present both the theoretical definitions (reference systems) and practical implementations (reference frames), so origins, axes, units and ranges will be established for every system. Furthermore, the chapter describes the relations and conversions between the different systems.

It is important to note that here we deal mainly with a model-based set of reference systems and conversions. In practice, Gaia aims to be a "self-calibrating" instrument. This means that the real systems and conversions will be based mainly on calibrations rather than on models and, therefore, some of the definitions proposed here may become easier –even unnecessary.

The purpose of the work presented in this chapter was to develop a draft of the reference systems and conventions document, to be completed by the corresponding members of the GST and other groups related to Gaia. All the comments, corrections and completions have been included (specially those related to relativistic effects and formulation), in order to obtain a fully consistent definition. A final document has been prepared from this work that will be used as a reference guide for the whole scientific community of Gaia, as well as for the industrial development teams. Therefore, the aim of this work was to begin establishing some standards about the reference systems of Gaia, in order to unify efforts and, from here, to obtain consistent documents on Gaia.

Section 1 describes all the reference systems one by one and independently. Their relations and conversions will be described in section 2. Section 3 proposes a set of conventions and notations for the instrumentation and measurements of Gaia. Finally, we include a summary of the reference systems and a figure of the astrometric focal planes, which is needed to better understand some parts of this work.

2.1. REFERENCE SYSTEMS AND FRAMES

This section proposes a set of definitions of reference systems for Gaia. These Reference Systems (RSs) are described independently, starting with a simple description of the RS and finishing with the practical implementation or Reference Frame (RF), including its origin, fixing the axes, defining the units, the valid ranges and the recommended resolution when necessary, as well as other necessary considerations.

These definitions will be done in a uniform way, this is, always using the Cartesian x/y/z coordinates. This will make possible "standard" or uniform conversions between reference systems (mainly using simple matrix multiplication). However, correspondences with other standard coordinates (e.g., right ascension, declination and distance) will also be given.

It is very important to note that not only these definitions of RSs but also their relations (cf. section 4) will be based only on a Newtonian formulation. Relativistic formulation is not considered here (although some relativistic effects may filter in), while they are being included in other documents continuing this work. Finally, these definitions use the Gaia-2 design as defined in EADS/Astrium (2002).

2.1.1. Global view of Reference Systems

The Reference Systems for Gaia can be classified in a hierarchical structure, starting with the ICRS on the top (as "global" reference system, with its origin at the barycenter of the Solar System) and finishing with "local" and very specific RSs for every CCD or every measurement. Furthermore, generically speaking these RSs can be grouped into two broad classes: Payload Reference Systems and Astronomical Reference Systems. The second group stands for "global" reference systems (basically derived from the ICRS), based on equatorial coordinates but in a 3–D Cartesian representation (x/y/z), while the Payload group is used mainly to represent the measurements and the focal planes, expressed in a 2–D Cartesian system for an easier understanding.

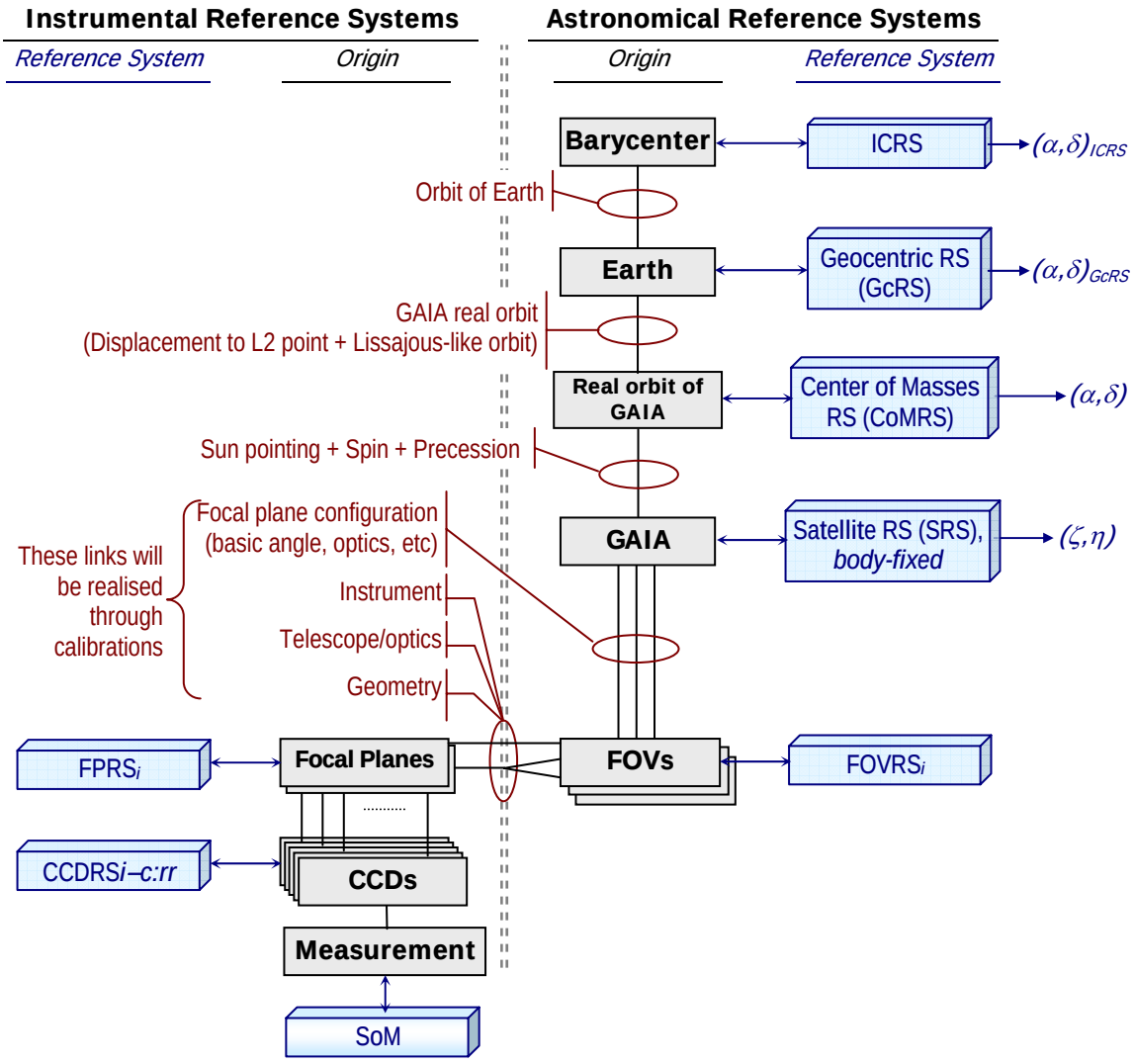


Figure 2.1: Full scheme of the hierarchy of Reference Systems

Figure 2.1 shows a global view of this Reference Systems structure, linking the two groups at a FOV/FP level –although they may be linked at a satellite/FP level using calibrations, as explained hereafter. These two self-consistent groups show how Astronomical RSs are linked from the ICRS to each FOV reference system (via a body-fixed SRS), while Payload RSs are linked from a measurement or a CCD to the whole focal plane. Gaia itself implements the FOVRS/FPRS link when projecting every field of view (i.e., the sky) over a 2–D focal plane. This projection is defined mainly by the instrument geometry and optics, but calibration methods will lead to a direct conversion avoiding complex models. Therefore, the Astronomical–Payload link will consist, in fact, of three different links (one for each

instrument: Astro-1, Astro-2 and Spectro), although 2 of them will be very similar: the two Astro instruments converge into the same focal plane.

This global scheme shows not only the several RSs as described, but also the objects or concepts (origins of the RSs) which the systems are based in, as well as the transformations needed to move from one system to another. It is important to note here that each and every one of these RSs includes (MUST include) the definition of a time coordinate. Also, all of the parameters and links between RSs depend on time: for example, the displacement from the barycenter to the center of the Earth (ICRS→GcRS) varies with time, as fixed by the orbit of the Earth. Another example is the Astronomical↔Payload link, where the calibration of the instrument can vary with time (e.g., deformations due to temperature variations).

In this figure, $(\alpha,\delta)_{ICRS}$ stand for the barycentric ICRS coordinates, while (α,δ) coordinates stand for the apparent equatorial coordinates seen from the nominal position (i.e., orbit) of Gaia, and $(\alpha,\delta)_{GcRS}$ stand for the geocentric equatorial coordinates as seen from the Earth. (ζ,η) stand for the *field angles* (described below) that are measured by the instrumentation. Finally, in the CCDRS, "*c*" and "*rr*" stand for the CCD column and row (1 and 2 digits, respectively).

In FOVRS, FPRS and CCDRS, "*i*" stands for the instrument identifier, but it can take different values depending on the system: In the FOVRS (and in the SoM) it can be "0" for the Spectro, "1" for Astro−1 and "2" for Astro−2; on the other hand, in the FPRS and CCDRS both Astro fields of view converge into a single focal plane, so "S" will be used for Spectro and "A" for Astro, in order to avoid confusions.

It is important to remind that this chapter is focused mainly on the model-based approach of reference systems, and this is the main reason of existence of the FOVRS. When dealing with calibrations (as in the real mission), this reference system will surely be bypassed.

2.1.2. Astronomical Reference Systems

The Reference Systems enclosed in this first group are used for astronomical purposes and hence, although the coordinates will be defined here in a Cartesian format (x/y/z), the most used coordinates are equatorial plus a parallax or a distance to the source. The ICRS will be taken as a starting point, and the hierarchy of RSs will finish with the FOVRS in this Astronomical group, linking to the next group with the projection of the sky onto each focal plane (obtained with very accurate models theoretically, and with calibrations in the real case).

As reminded by J. Kovalevsky, these celestial references will play essentially two roles:

1. To refer the observations (made within the kinematic frame centered on Gaia) to the ICRF, which implies referring Gaia reference system(s) to the ICRF.
2. To represent the path of Gaia in the Solar System in a barycentric reference whose axes are the same those of the ICRF.

Although these two problems are not necessarily similar, they are not treated separately in this chapter. When including the relativistic concepts and formulations in documents continuing this work they should be differenced.

2.1.2.1. International Celestial Reference System (ICRS)

This Reference System is, for sure, the basic reference used for astronomical coordinates and the basis for fundamental measurements of the positions of celestial bodies. It has a great importance within Gaia reference systems, even within the whole Gaia project: all of the ephemeris for the Solar System are nominally referred to this RS, as well as the final results of Gaia.

In a broad sense, the ICRS defines standard constants, data and algorithms for the processing and interpretation of precise astrometric observations. Because the ICRS is kinematically fixed to distant radio-sources, this reference system is entirely freed from any terrestrial reference and is considered as the best approximation available to a quasi-inertial reference system.

This RS is centered at the barycenter of the Solar System, this is, at its center of masses (which does not coincide with the center of the Sun, and has a complex multi-periodic motion when referred to the Sun). This RS is usually expressed on equatorial coordinates (Right Ascension, RA or α and Declination, DEC or δ, see below), although its axes are Cartesian. The time scale used in this RS is the Barycentric Time.

The following is a list of the main features of the ICRF, which fixes and implements the ICRS. This is obtained via a catalog of 608 extragalactic radio sources observed with VLBI. Further details can be found in [IR.5.].

- **Axes**: X, Y and Z.
  - Axis X: Nominally towards the J2000.0 spring equinox ($\alpha=0^h$, $\delta=0°$)[1].
  - Axis Y: Z×X. Therefore, in equatorial coordinates, Y-axis is defined as pointing towards $\alpha=6^h$, $\delta=0°$.
  - Axis Z: Nominally towards the celestial north pole ($\delta=90°$)[2].
- **Alternative coordinates**: α and δ, and eventually r.
  - α (Right Ascension): Range from 0^h to 24^h, defining the XY plane. Its origin (0^h) is in X=0 (spring equinox), and increases counterclockwise.
  - δ (Declination): Range from –90° to +90°. Its origin ($\delta=0$) is on the XY plane, and increases upwards.
  - r (Distance to the source): Range from 0 to ∞ (using linear units such as parsecs, meters or AU), defining the modulus of the vector towards the source once its direction is defined by (α,δ). Its origin ($r=0$) coincides with the barycenter. Sometimes the parallax (π) is used instead of r.
- **Time used**: Barycentric Time (TB)[3]. The discussion about the definition of this time scale could be several pages long, so we refer the reader to Kovalevsky et al. (1989), chapter 17, section 4 (p.436) for a thorough explanation.
- **Theories used**: Distant extragalactic radio sources used to define the ICRF are supposed to be at rest wrt the ICRF.
- **References used**:
  - Time reference: J2000.0 (12h TB on 1 January 2000).
  - List of extragalactic radio sources considered as "standards of rest" [IR.6.].
- **Undesired effects**: Uncertainty in the measurements. Rotation of the barycenter around the center of the Galaxy, which leads to a reference system which is locally inertial. Relativistic effects, light bending and other effects should be taken into account when measuring the reference objects.
- **Accessibility**: Nowadays the ICRF is defined in radio, in the S and X bands (wavelengths 13 and 3.6 cm, respectively) using VLBI. The obtained accuracy in the orientation of the axes is about 0.02 mas. Most of these radio sources have faint optical counterparts ($M_V>18$) and most of them are quasars, which are slightly extended

[1] Its real offset from the spring equinox at J2000.0 is 78±10 mas, so this would really be about $\alpha=0^h\ 0'\ 0.0007''$, $\delta=0°$ (Kovalevsky et al. 1989).

[2] The VLBI analysis shows that this pole at J2000.0 is shifted from the ICRS pole by 17.1 mas in the direction $\alpha=12h$, and by 5.1 mas in the direction $\alpha=18h$ (Kovalevsky et al. 1989).

[3] Barycentric Time is a time-like coordinate to be used when representing the motions of bodies in the Solar System and, in particular, Gaia. The current barycentric time is the Barycentric Coordinate Time (TCB). However, JPL still uses the Dynamical Barycentric Time (TDB), which has a different time unit; this use leads to difficulties.

objects and may have an emitting center depending on the wavelength, so a direct correspondence with the optical spectrum is not clear. The frame at optical wavelengths is defined by the Hipparcos catalogue, but the obtained accuracy is about 1 order of magnitude worse.

- **Data reduction techniques used**: The set of models and algorithms that are used to transform ICRS catalog data to observable quantities (and vice versa) are given in the IERS conventions (McCarthy 1996)[4]. We refer the reader to this reference, or to [IR.7.] for a thorough list.

2.1.2.2. Geocentric Reference System (GcRS)

In some cases this RS can become the most important one –even more than the ICRS. The reason is that most of the measurements of stars and other celestial objects (and, obviously, the measurements of elements of the Earth) are made from the Earth, so their coordinates are based in an on-ground reference system. Also, some of the ephemeris from ESOC will be given wrt this RS, so its definition within the Gaia project is very important.

The GcRS is very similar to the ICRS. The direction of their axes coincides, as well as many of the parameters of the frame: alternative axes, units, ranges, precision, theories, accessibility... in fact, the coordinates of very distant sources are similar in both the ICRS and the GcRS. However, its origin is instead placed in the barycenter of the Earth, in order to fit correctly within the Newtonian mechanics of the Solar System and, therefore, to link it correctly with the ICRS. The Time Scale is the Terrestrial Time (TT). The following are its main features:

- **Alternative coordinates**: equatorial coordinates could be used (taking into account the translation of the Earth around the barycenter), named $(\alpha,\delta)_{GcRS}$.
- **Time used**: Terrestrial Time (TT), which is defined as the proper time of a clock at the rotating geoid[5] (surface of the Earth) assuming that it does not suffer the gravitational field of the Earth. This clock would have an atomic nature, this is, it would give the time signals depending on an atomic law (a kind of integrated time, where an event happens repeatedly with a fixed and well-known period). For a thorough description of the TT we refer the reader to Kovalevsky et al. (1989), chapters 15 and 16.
- **Theories used**: Newtonian mechanics (relation with the Moon and the other objects of the Solar System, which affect the translation of the Earth). Other theories will be implicitly used because of the natural derivation of the GcRS from the ICRS.
- **References used**: The same used in the ICRS (in order to obtain correctly the axes of the GcRS), including the time reference (J2000.0).
- **Relativistic effects**: The rotation of the geocenter around the Sun leads to a non-inertial reference system. Light bending.
- **Accessibility**: Many measurements are made directly, so its accessibility is total in the whole spectrum.
- **Data reduction techniques used**: Same as in the ICRS. For further information we refer the reader to Kovalevsky et al. (1989).

[4] These Conventions are currently out of date in several respects, and a new version is in preparation. In any case, the reduction techniques described in this document concern only Earth-based observations, and may not be applicable to Gaia observations.

[5] The proper time at the geocenter is close to the TCG (Geocentric Coordinate Time) at that point, which is one of the four coordinates of the Geocentric Reference Frame (and is defined in the framework of general relativity).

2.1.2.3. Center of Masses Reference System (CoMRS)

In order to offer documents coherent with the GSR (ESA 2000, section 9.2.1), this reference system is introduced between the GcRS and the SRS. The Center of Masses Reference System is yet another displacement of the origin, from the geocenter to the best model of the actual orbit of Gaia. This is, the origin of the CoMRS is fixed to the center of masses of Gaia, but its axes are still parallel to the ICRS[6].

This reference system offers the possibility to describe observed directions and measurements in "ICRS-like" coordinates (i.e., using axes parallel to the ICRS), as seen from Gaia. Therefore, the axes of the CoMRS asymptotically coincide with the ICRS axes, but its origin is placed in the center of masses of Gaia. Also, the time used will be the Satellite Time, which will be calibrated periodically wrt the Terrestrial Time (TT). The following are the main features of the CoMRS:

- **Axes**: X_{CM}, Y_{CM} and Z_{CM}, parallel to the axes of the ICRS. The definition of "*parallel* axes" *must* be done (in documents deriving from this chapter) accordingly to general relativity.
- **Alternative coordinates**: Equatorial coordinates (α,δ) as seen from Gaia. Their definition (even their value) is similar to the right ascension and declination in the ICRS and in the GcRS, although aberration and other effects must be taken into account (due to the displacement and relative motion of the origin).
- **Time used**: Satellite Time (ST), which will be calibrated from the Earth. In principle, this time will be basically the same as TT but with a different phase and zero time (TBC). The "spatial origin" of this time scale (i.e., where the *clock* will be placed) is still TBD, and surely it will *not* coincide with the spatial center of coordinates –this main clock should be placed in the electronics and payload control module.
- **Theories used**: Newtonian mechanics, orbit of Gaia. Other theories will be implicitly used because of the natural derivation from the ICRS to the CoMRS.
- **References used**:
  - The same used in the ICRS and GcRS.
  - Ephemeris provided by the ESOC, before and during the mission.
  - The time reference will be J2010.0.
- **Accessibility**: Optical spectrum.
- **Data reduction techniques used**: The actual orbit of the satellite and, therefore, the actual behavior of this reference system, will be determined during the mission with daily measurements from the ground station (*ranging*). Also some techniques used for the ICRS and the GcRS may be included here.

2.1.2.4. Satellite Reference System (SRS)

In the previous subsection we have placed the origin of a reference system in the center of Gaia. Now we will introduce the Satellite Reference System in order to define an easier way to refer to the measurements made in Gaia. This RS includes a set of rotations (as seen from the CoMRS) in order to obtain a reference system *fixed to the body of the satellite*. This way, the

[6] The need of axes parallel to the ICRF implies that this system must be defined kinematically. But we are in an orbiting body, so the natural RS is rotating and that one must apply to observations the equivalent of the geodesic precession-nutation for the Earth. It is a rather large effect that amounts to about 20 mas per year. However, one may do what is done with VLBI observations on Earth: one determines the combination of this effect with that of the nutation and precession. So, similarly, when determining the attitude of Gaia using stars and quasars, this effect will be included in the variations of the direction of the rotation axis of Gaia.

coordinates given in this RS will be the coordinates "as seen *by* Gaia" (in the CoMRS we had coordinates "as seen *from* Gaia"). It is important to note that this RS is a definition *ab initio*, in order to offer a theoretical reference system for whichever necessary calculations or designs. In the practical case, this SRS will be fixed almost automatically, when executing the calibration process within the GIS.

The definition of this RS is, in fact, very simple: the coordinate system is centered within the body of Gaia, approximately in the geometric center of the payload (in the center of masses of Gaia). One of its axes points exactly towards the geometrical center of the Astro focal plane, thus bisecting the two astrometric FOVs, and another axis points towards the "north pole" of Gaia (on its rotation axis). In this way, the whole reference system will rotate and move the same way as Gaia does, wrt the ICRS or the GcRS. More details are given below. It is important to note that, again, the important displacement of the origin of coordinates (about $1.5 \cdot 10^6$ km from the GcRS) will lead to a different time coordinate. Furthermore, due to the –relatively– fast rotation of the system we may need to make some other corrections (*calibrations*), mainly in the time scale (TBC). There will be periodical calibrations of the instruments and the local clock wrt a ground reference clock (UTC-based clock).

- **Origin**: Center of masses of Gaia. It has to be determined[7], but it is *desirable* that the masses in the satellite would be distributed in a way that its natural rotation axis is approximately in the intersection of the optical axes of the Astro telescopes.
- **Axes**: X_S, Y_S and Z_S, fixed to the body of the satellite.
  - Axis X_S: Exactly towards the center of the astrometric focal plane (Astro), and more specifically towards its along-scan geometrical center (nominal mechanical center), so possible deformations in the structure of the payload may be included intrinsically. The technical characteristics described in Portell et al. (2003a) – already updated with the new Gaia specifications (EADS/Astrium 2002) – show that this along-scan center is in pixel 1849.0 of AF07 (approx. 41.11% of active surface from pixel 0 of the CCD).
  - Axis Y_S: $Z_S \times X_S$. Along the spin direction of Gaia, this axis will precede the X_S axis by 90°.
  - Axis Z_S: Coinciding with its rotation axis and pointing away from the Sun. This means that the Sun will have a negative Z_S coordinate.
- **Alternative coordinates**: spherical coordinates named *Field Angles* (η and ζ). These are the angles really observed by Gaia when scanning the sources. Their definition is similar to α and δ, corresponding the angle η to the along-scan direction, and angle ζ to the across-scan direction.
- **Time used**: Satellite Time (ST). See description in the previous section.
- **Theories used**: The SRS should be obtained from the CoMRS using the actual attitude of Gaia, represented by the best known attitude model at any given time (obtained from GIS). No special theory is needed for the definition of the SRS by itself.
- **References used**:
  - The same used in the ICRS and GcRS.
  - Ephemeris provided by the ESOC, before and during the mission.
  - The time reference will be J2010.0.

[7] This calculation should also take into account, for example, the mass distribution of the fuel and the emptying of its tanks during the mission. Surely this center of masses –and therefore the origin of coordinates– will be located below the payload, in the service module.

- **Relativistic effects**: Caused by the orbit model of Gaia (TBD).
- **Accessibility**: Optical spectrum.
- **Data reduction techniques used**: The real attitude of the satellite and, therefore, the real behavior of this reference system, will be determined during and after the mission with the Global Iterative Solution (GIS).

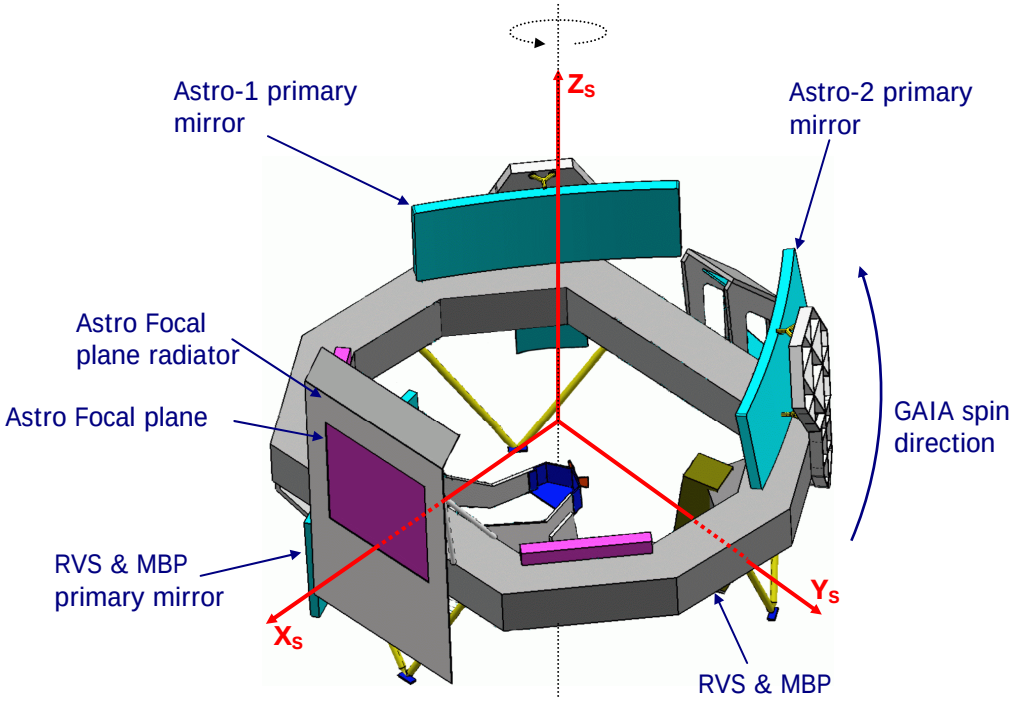


Figure 2.2: The SRS within Gaia

2.1.2.5. Field of View Reference System (FOVRS)

This Reference System is, in fact, composed by three different reference systems: one for each field of view (Astro-1, Astro-2 and Spectro). Therefore, a special notation will be used for these RSs (as well as in the FPRS):

- FOVRS1: FOVRS for the Astro-1 instrument.
- FOVRS2: FOVRS for the Astro-2 instrument.
- FOVRS0: FOVRS for the Spectro instrument.

These RS will be used to refer each observed object as *actually* seen by Gaia, this is, taking into account the features of every FOV: field observed, pointing and centering of the FOV, scanning law... Its definition, taking the SRS as the starting point, is not trivial: not only a translation and rotation of axes must be done, but also the optical paths should be taken into account. In a simple optical system (like a refractor telescope) it would be –more or less– simple, but in a complex 6-mirror structure like in Gaia the definition of a FOVRS, as seen from an external reference system like the SRS, is more complicated. Figures 2.3 and 2.4 show how this system is implemented. The conversion from the SRS is made in a fixed way –just taking into account possible deformations in the structure of the payload; in this way we will obtain again a body-fixed set of reference systems. The FOVRS is based on the optical systems of Gaia, and therefore it can include some concepts inherent to telescope systems.

Each RS will be centered in the corresponding focal plane, and more exactly in the corresponding optical center. The axes will be oriented in such a way that the projection of the sources will move towards negative values of one axis. Another axis will point towards the image of the Sun in the across-scan direction, and the third one will bisect the FOV following

the optical axis towards the light direction (with a sign corresponding to a right-handed coordinate system). These axes will make possible to place a source within this FOV, just as if it would be a "linear" FOV (ignoring the geometry of the reflections in the three mirrors). The following figure illustrates this generic definition.

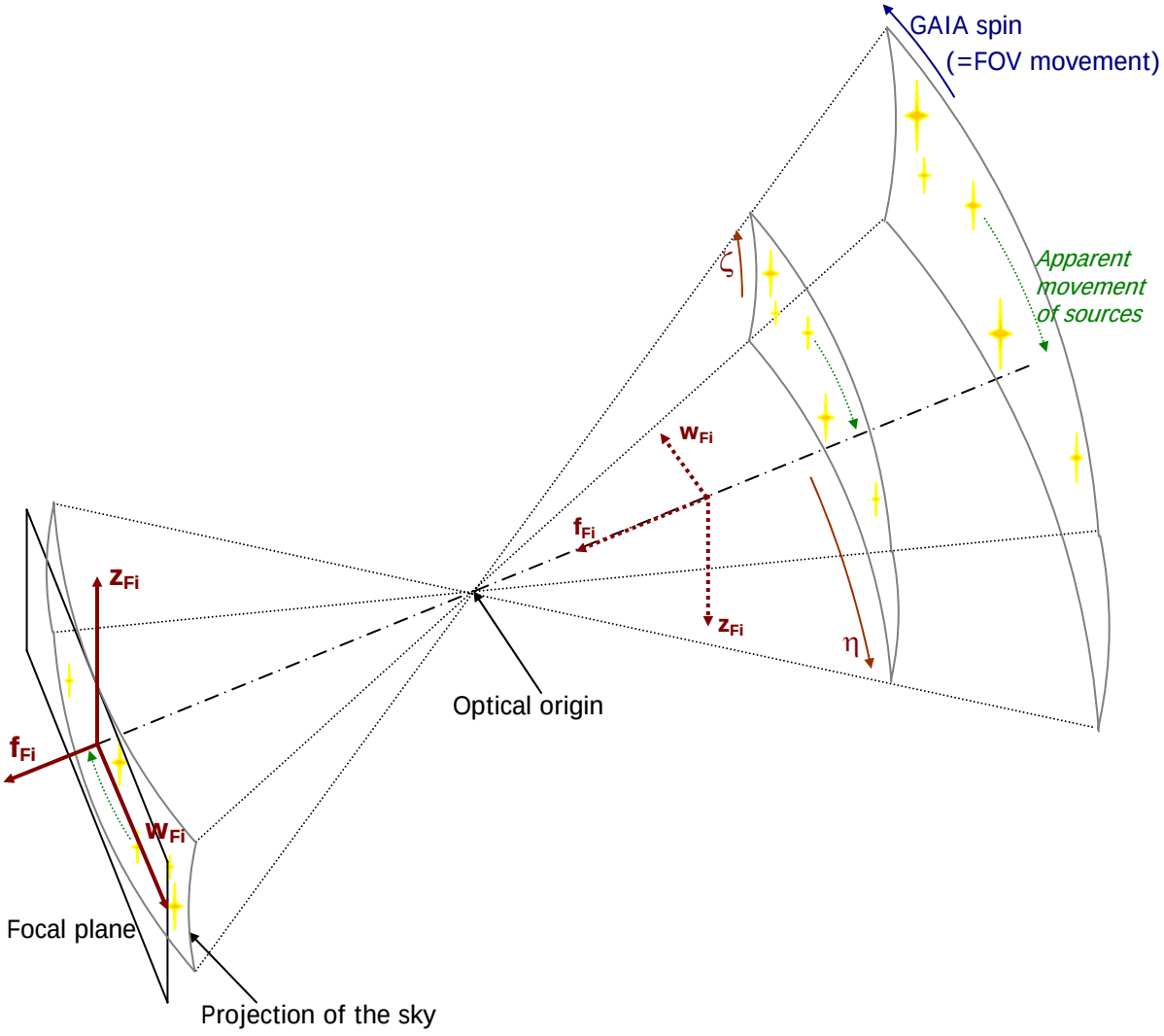


Figure 2.3: Coordinate system for a generic FOV of Gaia

More details on the optics of Gaia can be seen in EADS/Astrium (2002), although the exact definition of these optics is still TBD (and, therefore, so does the exact definition of the FOVRS, specially its conversion from the SRS). The time coordinate will be based on the Satellite Time, but each FOVRS will have its own time scale.

- **Origins:**
  - **FOVRS$_1$**: Nominal optical center for Astro−1 in the astrometric focal plane. Possible deformations on the structure may move this center wrt other parts of the satellite, but the BAMD will help tracking the variations in the basic angle due to these deformations and link the FOVRS with other reference systems. Taking the technical characteristics described in Portell et al. (2003a), this center would be exactly in the CCD column 5, row 08 (AF06), and more exactly in the pixel column 982.0, row 2249.0 (see details about this numbering in section 2.1.3.3). These coordinates could change depending on the optical and mechanical definition of the instruments.

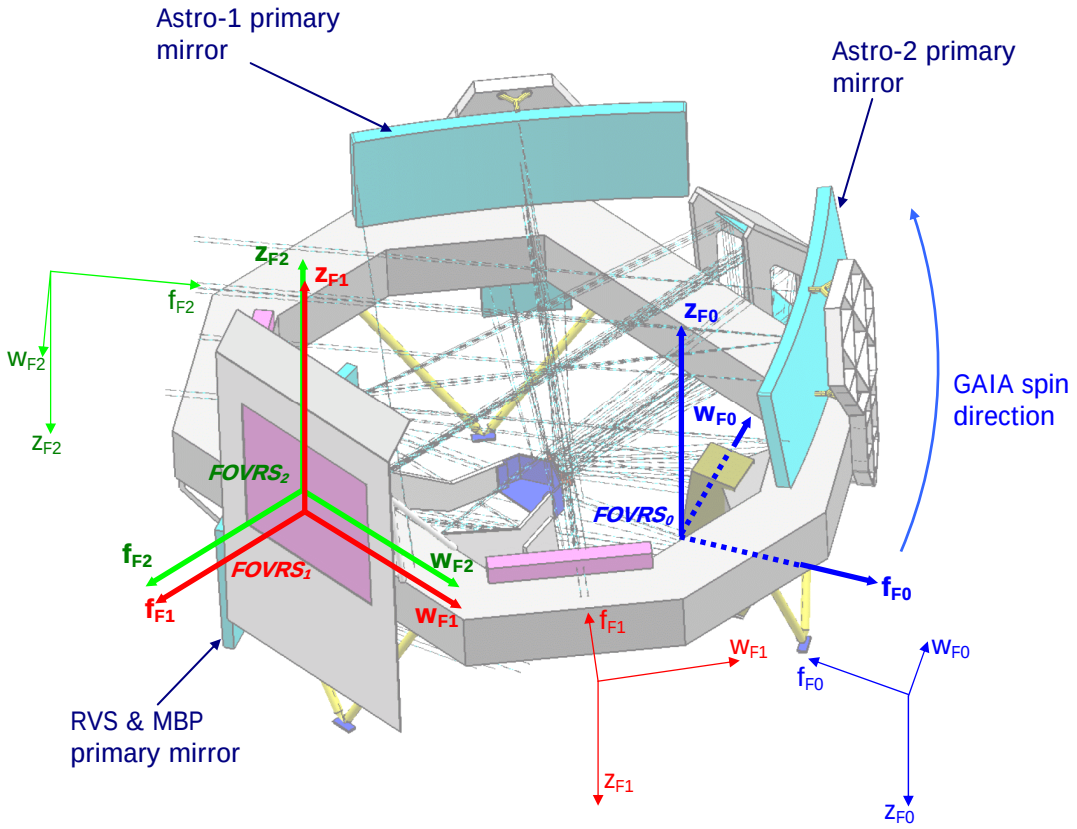
- **$FOVRS_2$**: Nominal optical center for Astro–2 in the astrometric focal plane. Same considerations about deformations. Coordinates: CCD column 4, row 08 (AF06), pixel column 982.0, row 2249.0.
- **$FOVRS_0$**: Nominal optical center of the Spectro FPA. Same considerations about deformations. Exact coordinates TBD.

- **Axes:**
  - **$FOVRS_1$**: f_{F1}, w_{F1} and z_{F1}.
  - **$FOVRS_2$**: f_{F2}, w_{F2} and z_{F2}.
  - **$FOVRS_0$**: f_{F0}, w_{F0} and z_{F0}.
    - Axes f_{F1}, f_{F2} and f_{F0}: following the optics path (i.e., the optical axis) with the same direction than light rays, this is, towards a hypothetical star the projection of which would coincide exactly with the origin of coordinates (but with the opposite sense). In other words, this axis is perpendicular to the focal plane and points away from the corresponding telescope.
    - Axes w_{F1}, w_{F2} and w_{F0}: opposite to the apparent direction of movement of the projected sources on the focal plane (i.e., from the BBP to the ASM for Astro, in figure 2.10). In other words, the images of the sources will move towards negative values of this axis. Because the focal planes could be slightly curved (or deformed), a thorough definition is needed: these axes will point towards an exact pixel coordinate. This coordinate, for w_{F1} and w_{F2}, is CCD column 5 or 4 (respectively) , row 0 (ASM1), pixel column 982.0, row 0.0 (just in the center of the pixel). This implies that possible deformations of the FPA could lead to slight rotations of this axis.
    - Axes z_{F1}, z_{F2} and z_{F0}: $f_{F1}\times w_{F1}$, $f_{F2}\times w_{F2}$ and $f_{F0}\times w_{F0}$, respectively. In figure 2.10, these axes will point from the bottom to the top, i.e., opposite to the Sun (so the Sun will have a negative z_{Fi} value). In the baseline these axes will point towards the image of the sun. In every focal plane, these axes will point towards the across-scan direction.

- **Alternative coordinates**: spherical coordinates, taking advantage of the intrinsic angular nature of this reference system. It is very important to note that, in order to use these angular coordinates in the right way, their origin must be on the optical origin of the FOV, this is, where the ray paths converge (i.e., exactly where the image gets inverted), as shown in figure 2.3. These alternative coordinates are η and ζ (field angles). These alternative coordinates will be possibly more used than the Cartesian ones because of the intrinsic angular nature of these RSs, although the Cartesian coordinates recommended will be used for the conversions between RSs. Different (η,ζ) ranges will correspond to different instruments (i.e., FOVs). Their definition is similar to α and δ, but their ranges are limited by the FOV of every instrument:
  - η: Full range within a FOV is 0.920° (nominal) for Astro-1 and Astro-2 (range in Spectro is TBD). Negative values will point towards the ASM, and positive values towards the BBP (i.e., the images of the sources will move towards positive values of η).
  - ζ: Full range within a FOV is 0.737° (nominal) for Astro-1 and Astro-2 (range in Spectro is TBD). Higher values will point towards the top of the focal plane shown in figure 2.10 (i.e., the image of the sun on the focal plane will have a positive ζ value).

  This definition of (η,ζ) is equivalent to the Right Ascension and Declination in the Earth and their use in equatorial telescopes: nominally, the tracking of a source as seen by Gaia can be done just increasing the value of η.

- **Ranges**: Valid ranges will be fixed by the "cone" of each FOV, this is, the area in figure 2.3 within the FOV. One can see that f_{Fx} must be negative, and that w_{Fx} and z_{Fx} valid ranges will increase as f_{Fx} decreases.
- **Precision**: continuous axes, so the resolution should be as high as possible, depending on the capability of the measurements. This obtained resolution will be basically the same than in the SRS, and the resolution obtained in the final results of the mission will depend on this FOVRS resolution.
- **Time used**: Every FOVRS will have its own time scale, which will be based on the Satellite Time (ST) but with a phase shift (or a delay) due to the transmission lines of the clocks. This effect is still TBD by the engineering teams. The precision required is high (about 10ns). The exact place for the clock is also TBD, but it could be placed very close to the origin of spatial coordinates.
- **Theories used**: Same as in the SRS. Also, optics and geometry of the instruments of Gaia should be considered.
- **References used**: Same as in the SRS.



Note: Thick axes and bold labels represent the reference systems and their origins. Thin axes show the way these reference systems are "seen" as in the ICRS or SRS, prior to the several reflections and rotations applied by the optics. Please note that thin f_{Fi} axes point towards the respective primary mirrors, while thick f_{Fi} axes point against the last mirror (M6 in Astro, M3 in Spectro).

Figure 2.4: Full set of the three FOVRS within the payload of Gaia

- **Undesired effects**: Same as in the SRS. Also, the optic paths of Gaia will depend on the wavelength and therefore the FOVRS set will also depend on it. We recommend to base all the FOVRS on a central wavelength (e.g., λ=700nm, TBC).
- **Accessibility**: Optical spectrum.
- **Data reduction techniques used**: Same as in the SRS.

2.1.3. Payload Reference Systems

In order to obtain unified documents from the several working groups of Gaia, not only a set of astronomical RSs has to be defined, but also a set of payload RSs for Gaia itself and its whole set of instruments. This section will propose their definitions and frames, based also in a Cartesian format (x/y/z). The FPRS will be used as starting point and, hence, this could be considered the "canonical" reference system for the payload group. However, the final reference system (the space of measurements, or SoM) will include all the data needed to place a measurement on the Gaia focal planes, so this could be considered as an "integrating" reference system –although it is really a way to express the measurements, not a reference system by itself.

It is important to note that these payload reference systems will take into account only the measurements made by the instruments of Gaia. These measurements will be done mainly with CCDs and other 2–dimensional instrumentation. Therefore, the intrinsic nature of the payload RSs will be also 2–D, except for the SoM, which will integrate all the features of the measurements and, therefore, could be considered as a multi-dimensional reference system (as described in section 2.1.3.3).

2.1.3.1. Focal Plane Reference System (FPRS)

The FPRS is composed of two Reference Systems, one for each instrument of Gaia (astrometric focal plane and RVS/MBP focal plane). Therefore, a sub-index will be included in order to determine which RS we are referring to: $FPRS_S$ (Spectro focal plane) and $FPRS_A$ (Astro focal plane). Please note that here we use the "A"/"S" sub-indexes, not "0"/"1"/"2", because the Astro focal plane includes data coming from both astrometric telescopes. These reference systems will be referred to a whole focal plane, this is, to the whole matrix of CCDs of every focal plane.

Each FPRS will be centered in the mechanical center of the Focal Plane Assembly (FPA), with one axis against the nominal apparent along-scan movement of the sources, and the other at 90° clockwise (as seen from the CCD side), pointing against the Sun (i.e., towards the image of the Sun on the focal plane). We remind that these reference systems are defined as 2–dimensional. The coordinates measured in this reference system (w_i and z_i, direction cosines in the FPRS) will be the primary coordinates of Gaia. Finally, the time scale will be based on the Satellite Time, although each FPRS will have its own time coordinate (the precision of which is required to be high, about 10ns).

- **Origins:**
  - **$FPRS_A$**: Nominal mechanical center of the Astro FPA (which, as shown in figure 2.10, does not coincide with none of the optical centers). Possible deformations on the structure may move this center wrt other parts of the satellite, but these changes will be integrated in the calibration obtained from GIS. The technical characteristics described in Portell et al. (2003a) show that this center is located between CCD column 4 and 5 (in the center of the dead zone, which is ~4mm), and in pixel row 1849.0 of CCD row 08 (AF07).
  - **$FPRS_S$**: Nominal mechanical center of the Spectro FPA. Same considerations as in $FPRS_A$, but the coordinates (CCDs, pixels, etc.) are still TBD.
- **Axes:**
  - **$FPRS_A$**: w_A and z_A.
  - **$FPRS_S$**: w_S and z_S.
    - Axes w_A and w_S: In the along-scan direction (*opposite* to the apparent movement of sources), and more exactly pointing towards the center of the dead zone between CCD column 4 and 5, in the CCD row 00, pixel row 0.0.

These axes model the focal planes as *true planes* (although imperfections or corrections within this plane, as CCDs tilt, will be considered). This implies that possible deformations of the FPA could lead to slight rotations of this axis. The projections of the sources will move towards negative values of these axes.

- Axes z_A and z_S: Rotated 90° clockwise from w_A and w_S (as seen from the CCD side). Looking at the focal plane as seen in figure 2.10, these axes point from the bottom to the top (i.e., against the Sun), following an across-scan direction. These axes point towards the (theoretical) image of the Sun.

- **Alternative coordinates**: w_i and z_i coordinates are defined in concordance with the FOVRS axes. However, the FPRS can be considered as an "engineering" reference system, while w_i and z_i axes do not coincide with the usual engineering senses. Therefore we can define these simple conversions:

  - Axes x_A and x_S: Equal to $-w_A$ and $-w_S$ respectively. This is, tracking the sources projected on the focal plane.

  - Axes y_A and y_S: Equal to $-z_A$ and $-z_S$ respectively. This is, opposite to the (theoretical) image of the sun.

  The origin of these alternative coordinates is the same than the standard coordinates.

- **Units**: Millimeters (mm) are recommended.

- **Ranges**: Full range (nominal, TBC) is 745mm along-scan (w_A) and 599mm across-scan (z_A). Full ranges for w_S and z_S are still TBD. These ranges do not represent an absolute limitation for the axes: possible imperfections, deformations or calibration of the final FPA could lead to slightly higher effective ranges, so a security margin (e.g., 5%, TBD) should be taken into account.

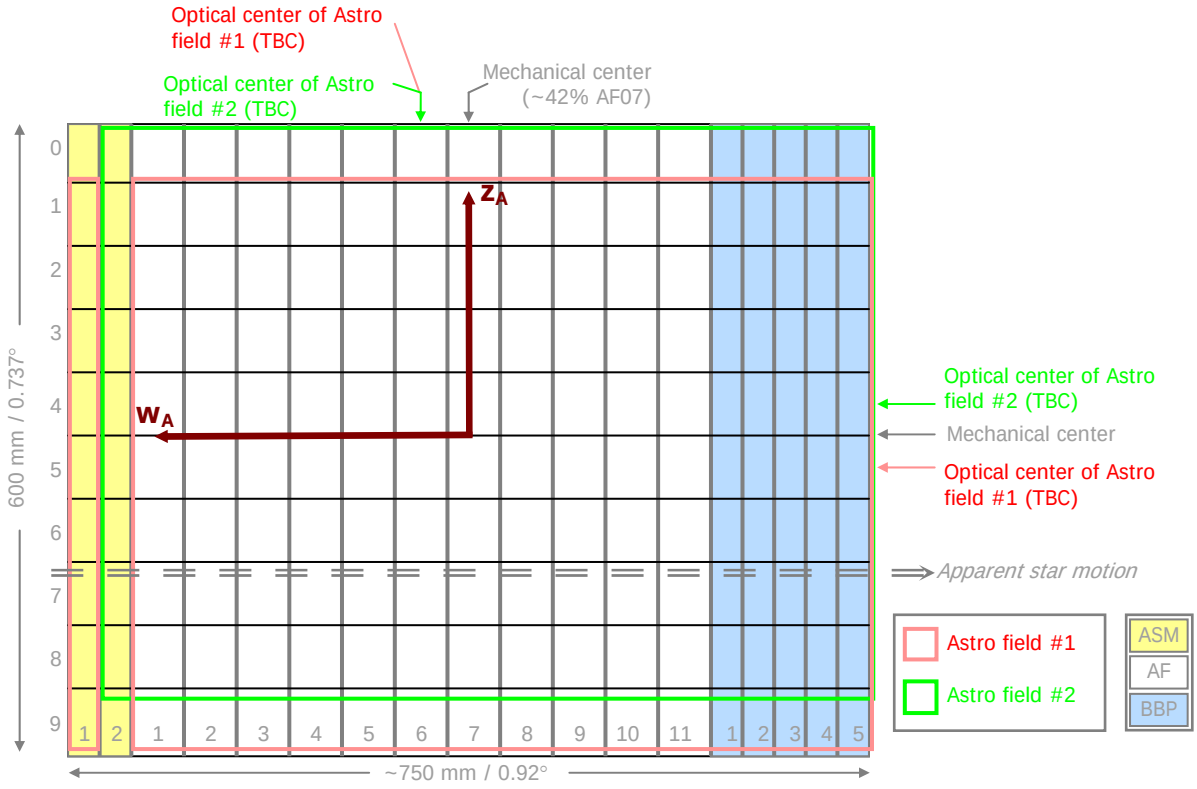


Figure 2.5: The $FPRS_A$ within Astro focal plane

- **Precision**: In principle the resolutions could be as high as required. However, because of the intrinsically discrete nature of the focal planes (they are composed of CCDs, which are composed of pixels) a limiting resolution is, more or less, well defined:

- When referring to payload elements or direct measurements, a resolution of 50nm should be enough (along-scan pixel size in Astro instruments is 10μm).
- When data reduction algorithms involve in, a source will be located with a resolution of microarcseconds. This implies a required resolution of 1pm on the focal plane, which will also be useful for referring to CCD tilts or other small corrections and calibrations on the focal plane.

- **Time used**: Satellite Time (ST). The displacement of the origin wrt the SRS is no relevant for the time scale. It is important to note that the focal planes need an extremely accurate time scale, as well as an almost perfect synchronization between focal planes (at the level of some 10ns). This may lead to some problems due to the distance between focal planes, so different time scales could be needed for each one of the FPRSs (TBC). If so, the two time scales should be named ST_A and ST_S.
- **Theories used**: Same as in the SRS. Also, optics and geometry of the instruments of Gaia should be considered, as well as the manufacturing and calibration details of the focal planes.
- References used, undesired effects, accessibility and data reduction techniques used: Same as in the SRS.

2.1.3.2. CCD Reference System (CCDRS)

We have defined a set of reference systems each one referred to a whole focal plane. Now, in order to offer a more detailed coordinate system, specially to characterize detections in a given CCD (or even problems or corrections in a given CCD), we will describe here the set of CCD reference systems. There will be one CCDRS per CCD, so there will be 180 CCDRSs for the astrometric focal plane and about 41 CCDRS for the spectrometric focal plane (9 RVS + 32 MBP, TBC), so the total number of CCDRSs will be 221. The nomenclature used to name these reference systems is the following:

- **$CCDRS_A$–c:rr**: CCDRSs in the Astro instrument, corresponding to the CCD of column *c* and row *rr*. "Columns" and "rows" nomenclature, as explained in section 5, is taken from the "industrial point of view", i.e., transposed wrt figure 2.10. Therefore, the columns are across-scan, and range from 0 (the top of figure 2.10, i.e., towards a theoretical image of the Sun) to 9 (the bottom, i.e., towards the Sun), while the rows are along-scan and range from 00 (corresponding to ASM1) to 17 (corresponding to BBP5), so a source image will move towards greater values. As an example, the optical center of Astro-1 FOV is within the $CCDRS_A$–5.07.
- **$CCDRS_S$–c:rr**: CCDRSs in the Spectro instrument. The columns/row convention will be the same used for Astro instruments, but the final number and distribution of CCDs (and therefore the ranges for *c* and *rr*) is still TBD.

The definition of a set of FPRSs makes possible the localization of a pixel, CCD, correction, or anything else on the focal planes. Now we will go further with the reference systems, and not to define a standard linear RS but a "discretized" one. This is, the CCDRSs will not be based on standard linear units (like millimeters), but on the measurements, i.e., on pixels. It is important to note that this definition makes the CCDRS independent of possible calibrations or deformations of the focal planes.

Every CCDRS will be centered on the left upper pixel (as shown in figure 2.6) of each CCD, leading to always-positive axes *only when using the alternative "engineering" axes*, and continuing the column/row convention. This is, one axis will give the pixel column, increasing (in absolute value) towards the last CCD column (i.e. towards the bottom, in figure 2.10), and the other axis will give the pixel row, increasing (in absolute value) towards the along-scan direction.

- **Axes**:
  - **CCDRS$_A$–c:rr**: $w_{A–c:rr}$ and $z_{A–c:rr}$, fixed to the corresponding Astro CCD.
  - **CCDRS$_S$–c:rr**: $w_{S–c:rr}$ and $z_{S–c:rr}$, fixed to the corresponding Spectro CCD.
    - Axes $w_{i–c:rr}$: In the along-scan direction, following the pixels of the CCD in the opposite direction. This is, from right to left in figure 2.10 (in order to maintain the *w* and *z* senses). The image of a source will move towards negative values.
    - Axes $z_{i–c:rr}$: In the across-scan direction, following the pixels of the CCD in an opposite direction. In figure 2.10 this is from the bottom to the top, i.e., towards the image of the sun.
- **Alternative coordinates**: As in the FPRS, the CCDRS can be considered an "engineering" reference system. In order to offer a more practical set of coordinates, the following simple conversions can be done:
  - Axes $x_{i–c:rr}$: Equal to $–w_{i–c:rr}$, this is, following the pixels of the CCD. This is the usual sense of an "x" coordinate in similar engineering systems. The image of a source will move towards greater values.
  - Axes $y_{i–c:rr}$: Equal to $–z_{i–c:rr}$, this is, following the pixels of the CCD in the across-scan direction. This is the usual sense of a "y" coordinate in similar engineering systems.

  The origins of these alternative axes will be the same than in "w" and "z" coordinates.
- **Units**: Pixels.

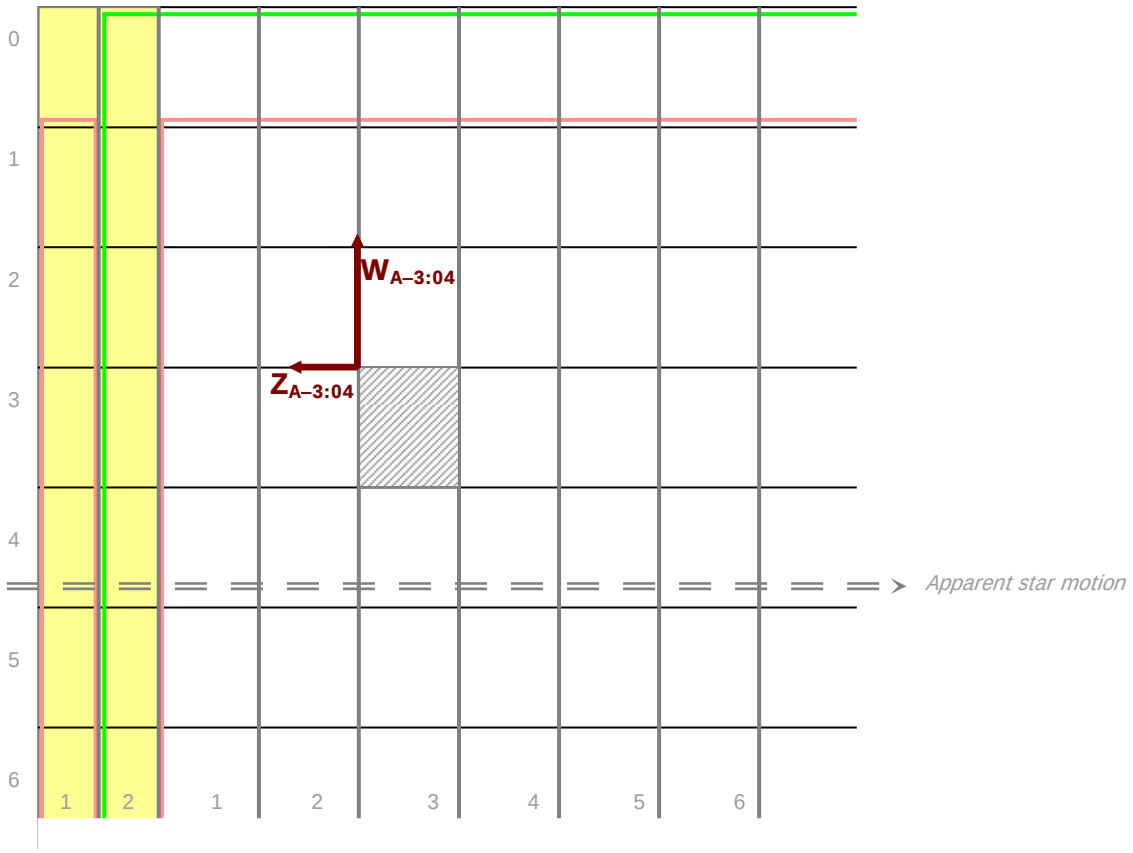


Figure 2.6: The CCDRS$_A$–3:04 within the astrometric focal plane

- **Ranges**: In the astrometric instruments, from –1965.5 to 0.5 pixels for $z_{i–c:rr}$ (across-scan), and from –2559.5 to 0.5 pixels (ASM/BBP), or from –4499.5 (AF) to 0.5 pixels,

for $w_{i-c:rr}$ (along-scan) and. In the Spectro still TBD. The center of a pixel will have a zero sub-pixel value (i.e., its coordinate will have the ".0" decimal).

- **Precision**: Taking into account the resolution obtained with the on-board algorithms (e.g., the detection algorithm) about 1/100 pixel should be enough. However, in order to provide a higher flexibility (even taking into account the final resolution of the mission), $1/10^7$ pixel is recommended here.
- **Time used**: Satellite Time (ST), with the same time scale and synchronization restrictions than the FPRS.
- **Undesired effects**: CCD tilt (when defining a CCDRS from the FPRS).

2.1.3.3. Space of Measurements (SoM)

The last reference system to establish for Gaia is not exactly so, but a definition of the measurements that Gaia can do. Its definition aims to offer a unified method to refer whichever measurement made by any instrument of the payload: Astro (coming from field 1 or 2), or Spectro. In the previous sections we have established reference systems that can be well interpreted from a geometric point of view, this is, they can be represented with a system of coordinates in 2−D or 3−D, and a stellar source or an object can be easily placed within them. Now we go further with the definition of an observations space which could be considered as a multidimensional reference system: a system that will have one dimension for every element of a measurement. This means that the SoM will be a space with 6 dimensions plus a time scale: instrument, CCD (2 coordinates), pixel (2 coordinates), flux read and time of measurement. Details are given below.

Despite of this high number of dimensions, thanks to the clear meaning of its coordinates a measurement expressed in SoM can also be easily represented: the charge of the measurement of each source shifts from one pixel to another, making successive measurements along the focal plane with several CCDs. Therefore, an "infinite" ribbon of pixels is obtained (in fact, for each CCD). This is represented in figure 2.7.

Here we must remind the difference between the several kinds of pixels:

- *Physical pixels* are the real pixels on the CCD (each covering an area of $10\times30\mu m^2$, TBC).
- *Effective pixels* are the *samples* readout from the video chain, this is, the total charge of several physical pixels (e.g., 12 pixels across scan for AF02-10 for G=12-20, TBC).
- *Data pixels* or *transmitted pixels* are the samples transmitted to ground, which will be the only ones used for data reduction. Usually they coincide with effective pixels (i.e., each sample readout is transmitted to ground), although sometimes a data pixel may be obtained from the addition of several effective pixels (e.g., 12 pixels across scan for AF01, obtained from 6 2-pixel samples, TBC).

It is important to note that in the SoM there exists a direct relation between two of its coordinates: the along-scan pixel coordinate and the time (even the CCD row is related to the time). This is caused by the TDI operation mode of the CCDs, which leads to readout data only in the last pixel line of each CCD and at fixed times. This leads to a "redundant" component in the SoM (this is, the pixel along-scan coordinate), although we will maintain this in order to offer a more intuitive system. Moreover, because of the different TDI periods in Gaia (e.g. the AF and the BBP fields in Astro may have different TDI periods) the pixels cannot be labeled only with their time, but also with their position (or with their real TDI period), and vice versa.

For the definition of the SoM, a measurement made by Gaia will contain the following items:

- **Instrument** where the measurement has been done (Astro-1, Astro-2 or Spectro), which will fix the "vector space of measurements". Please note that here the astrometric field (1 or 2) *does* matter, in order to obtain a completely defined

measurement. Finally, it is TBD whether or not the Spectro instrument is broken up in two parts: RVS and MBP.

- **CCD** (2 coordinates). This will identify the CCD coordinate where the measurement has been done, using the same convention proposed for the CCDRS.
- **Pixel coordinates** (2 coordinates), which will indicate where the data has been measured. In principle this should be noted with 1-pixel resolution (these are *real* measurements), but if necessary this coordinate could refer to the centroid of the source (obtained after processing the pixel data). The *really measured* data will correspond to the last pixel row (i.e., the readout register) of every CCD, and the rest of them will be calculated using the real TDI period of that CCD.
- **Flux**, which will indicate the flux measured in the given coordinates (the previous coordinates, i.e., instrument, CCD and pixel).
- **Time** when the data has been measured. Its resolution should be much higher than 1 TDI.

Each one of these items will be represented by a coordinate in the SoM. The following is a thorough definition for the implementation of the SoM:

- **Origin**: It cannot be fixed, because this is an observations space –not a reference system. The zero values of every coordinate will be indicated below, although a "space origin" could be placed at the left top of each focal plane (as seen in figure 2.10), i.e., at coordinates (0.0,0.0) of $CCDRS_i$-0:0.
- **Axes**: (i, c_w, c_z, p_w, p_z, fl, t_m).
  - i: Instrument where the measurement has been done.
  - c_w: CCD row (along-scan) where the measurement has been done.
  - c_z: CCD column (across-scan) where the measurement has been done.
  - p_w: Pixel row (along-scan) of the measurement. Effective measurements are done only for fixed values of p_w (the last pixel row or readout register of each CCD). The definition of the SoM for the rest of p_w will be defined based on the TDI rule for each CCD.
  - p_z: Pixel column (across-scan) of the measurement.
  - fl: Flux read at (i, c_w, c_z, p_w, p_z).
  - t_m: Readout time of the measurement at (i, c_w, c_z, p_w, p_z).
- **Alternative coordinates**: The pixel "engineering values":
  - p_x: Equal to $-p_w$.
  - p_y: Equal to $-p_z$.
- **Units**: Intrinsic to every axis:
  - i: None (i coordinate is just an identifier).
  - c_w: CCD.
  - c_z: CCD.
  - p_w: Pixels, although intrinsic units to the measurements will be the samples. The size of a sample is given in pixels. If samples are used instead, a simple conversion factor should be applied:

    $$pixel\_coord = sample\_coord \times sample\_size$$

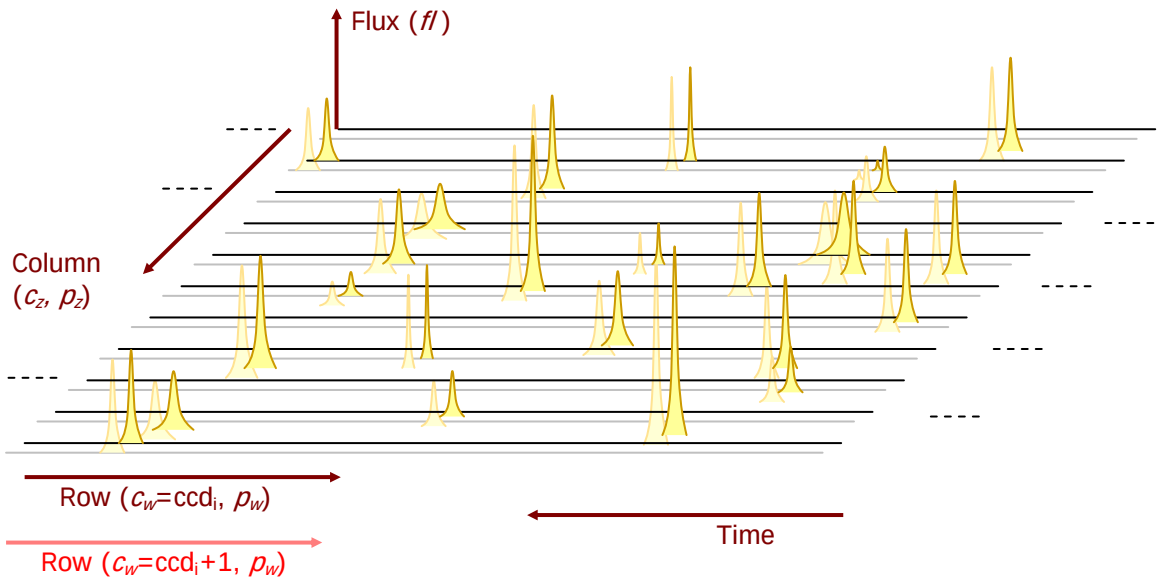
    (taking into account the possibly different sample size along– and across–scan).
  - p_z: Pixels, but same as p_w (i.e., if samples used instead).

- *fl*: electrons (e^-).
- t_m: Seconds.

- **Ranges:**
  - *i*: Valid values are 0 (Spectro), 1 (Astro-1) and 2 (Astro-2).
  - c_w: 0 to 17 CCDs in Astro-1 and Astro-2. TBD in the Spectro.
  - c_z: 0 to 9 CCDs in Astro-1 and Astro-2. TBD in the Spectro.
  - p_w: –2559.5 to 0.5 pixels (ASM/BBP), or –4499.5 to 0.5 pixels (AF), in Astro-1 and Astro-2. TBD in the Spectro. This numbering implies that the center of a pixel has a ".0" decimal in its coordinate, e.g., the coordinate of the center of the first pixel is "0.0".
  - p_z: –1965.5 to 0.5 pixels in Astro-1 and Astro-2. TBD in the Spectro. This numbering also implies that the center of a pixel has a ".0" in its coordinate.
  - *fl*: 0 to 189.999 e^- (TBC).
  - t_m: 0 to ∞ seconds (although the mission time will be limited to about 10^9 seconds).
- **Precision:**
  - *i*: Not applicable.
  - c_w: 1 CCD.
  - c_z: 1 CCD.
  - p_w: At least 1 pixel is needed, but $1/10^7$ pixel is recommended for generic coordinates on the SoM.
  - p_z: $1/10^7$ pixel (Same as p_w).
  - *fl*: 0.01 e^- (TBC), in order to be able to indicate the readout noise and other undesired effects.
  - t_m: 1 ns is recommended (TBC).
- **Time used**: Satellite Time (ST), with the same time scale and synchronization restrictions than the FPRS.
- **Theories used**: None special, just the dimensions of the focal planes. Also, the rules of the TDI operation mode are used in order to obtain the data corresponding to a generic p_x coordinate. These rules are the following:
  - Measurements are done (i.e., are *real*) only at p_w= –2559.0 pixels in the ASM or BBP, and at p_w= –4499.0 pixels in the AF.
  - For the rest of p_w (i.e. pixel rows), flux data can be found using the expression

    $$fl(i,c_w,c_z,p_w,p_z,t) = fl(i,c_w,c_z,p_{readout},p_z,t+TDI\times(p_{readout}+p_w))$$

    where *TDI* is the charge transfer period for 1 pixel in CCD c_w:c_z, and $p_{readout}$ is the corresponding value of p_w where the measurement is done (–2559.0 for ASM/BBP, –4499.0 for AF).
- **References used**: time reference (same as FPRS) and TDI clocks for the whole set of CCDs of each focal plane.
- **Undesired effects**: Measuring errors due to the readout noise and other effects. Also the central wavelength must be taken into account.
- **Accessibility**: Optical spectrum.



Note: A set of "infinite" pixel ribbons will be obtained for each CCD row (c_w), each set composed of several pixel ribbons (from across-scan CCDs). In the figure two sets (corresponding to consecutive CCD rows) are represented, each composed of 10 pixel ribbons.

Figure 2.7: A representation of the SoM

2.2. CONVERSIONS

This section describes how a given set of coordinates can be expressed in one or another reference system. These conversions will be formally based on matrices and vectors, although some hints will be given in order to understand each conversion step. This uniform way of conversion (matrices + vectors) will make possible an easy concatenation of several conversions.

It is very important to note that these conversions are based in a **Newtonian formulation**, without the inclusion of any relativistic effect. As noted by J. Kovalevsky (see introduction of section 2.1.2), this section (as is, in Newtonian formulation) may seem not really useful, but for visualizing goals. Relativistic formulation should be defined by the Gaia Relativity Working Group, and the difference between the two main goals of the celestial references (see introduction of section 2.1.2) should be clearly noted. Klioner (2001) and De Felice et al. (1997) may help in someway to include this formulation.

2.2.1. Algebraic conversions

In this first subsection only formulae for the conversions will be presented but with no data. Afterwards some useful data or approximations will be given in order to calculate model-based conversions between reference systems.

The scheme shown in figure 2.8 will be taken as a reference, considering here the initial reference system as O and the target reference system as O', this is, when we say ICRS-GcRS conversion, the ICRS is centered in O (with axes X, Y and Z) and the GcRS is centered in O' (with axes X', Y' and Z'). Nomenclature used in the conversions will include a rotation matrix **A** and a translation vector **d**, leading to the conversion:

$$\vec{v}' = \mathbf{A}(\vec{v} + \vec{d})$$

Rotation will be applied to the axes (not to the coordinates), so the following generic rotation matrices will be used:

- Rotation of angle α around X axis:

$$\mathbf{R}_X(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{pmatrix}$$

- Rotation of angle α around Y axis:

$$\mathbf{R}_Y(\alpha) = \begin{pmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{pmatrix}$$

- Rotation of angle α around Z axis:

$$\mathbf{R}_Z(\alpha) = \begin{pmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Also, other perturbations should be included in these conversions, whether they affect the rotation or the translation. Generically, these effects (noises, calibrations, CCD tilts, etc.) could be included as a "noise" matrix **N** plus a "noise" vector **n**, so the final formula should look like this:

$$\vec{v}' = \mathbf{NA}(\vec{v} + \vec{d} + \vec{n})$$

The time scale conversion is more difficult to formulate, but some kind of function should be found in order to approximate this conversion as t'=f(t).

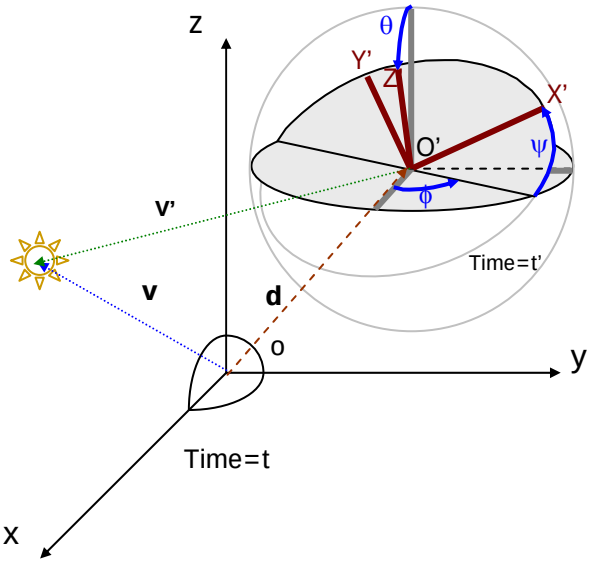


Figure 2.8: Transformations between Cartesian reference systems

2.2.1.2. ICRS-Geocentric RS

This first conversion is easy, taking into account that GcRS axes are parallel to ICRS axes and therefore only a translation is needed. It is important to note that this translation will depend on time, so a velocity conversion is also needed. This translation and velocity conversion will be given by the ephemeris of the *Bureau des Longitudes* (BDL). The summary of this conversion is the following:

- **Translation**: From the Barycenter of the Solar System to the Geocenter (Barycenter of the Earth). Translation vector:

$$\vec{d}_{ICRS\rightarrow GcRS}(t) = \left(-e_x(t), -e_y(t), -e_z(t)\right)$$

where ($e_x(t)$, $e_y(t)$, $e_z(t)$) are the (time-dependent) coordinates of the Geocenter in the ICRS.

- **Rotation**: None, therefore the rotation matrix is the Identity matrix:

$$\mathbf{A}_{\mathbf{ICRS\rightarrow GcRS}} = \mathbf{I} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- **Noise and imperfections**: Imprecision in the definition of the ephemeris.
- **Time scale conversion**: Based on measurements and ephemeris given by the corresponding organizations[8].

2.2.1.3. Geocentric RS – Satellite RS

This conversion, which aims to offer a body-fixed RS from the GcRS, could be very complex if models and NSL angles are used[9]. However, using ephemeris and other operational representations of Gaia attitude, this conversion will be very simple:

- **Translation** (and velocity change): From the Geocenter (Barycenter of the Earth) to the Center of Masses of Gaia, using the (time-dependent) satellite orbit wrt the geocenter, to be given by the ESOC. Translation vector:

$$\vec{d}_{GcRS\rightarrow SRS}(t) = \left(-g_x(t), -g_y(t), -g_z(t)\right)$$

where ($g_x(t)$, $g_y(t)$, $g_z(t)$) are the coordinates of the Gaia CoM in the GcRS, which equal to:

$$\vec{d}_{GcRS\rightarrow SRS}(t) = -\vec{d}_{ICRS\rightarrow SRS}(t) + \vec{d}_{ICRS\rightarrow GcRS}(t)$$

this is, barycentric satellite orbit minus barycentric Earth orbit[10].

- **Rotation**: It will be given by the attitude matrix. The attitude of Gaia may be expressed in the quaternion form, as a set of Euler angles or even as a simple rotation matrix. It will be self-calibrated during the GIS.
- **Noise and imperfections**: Only those implied by the measurement methods used in the definition of the satellite ephemeris.
- **Time scale conversion**: Based on measurements and ephemeris given by ESOC, and including periodical calibrations of the on-board clock.

---

[8] The reduction of Earth-based observations to the ICRS is a classical procedure in TT, using the precession-nutation data of IERS, which already contains the geodesic precession-nutation. The relations between TT and TCG, and between TCG and TCB, are well documented. The factor between the rates of the latter is 1.48×10^{-8} per day. The correction is not negligible for the computation of the aberration.

[9] This model-based conversion, used in previous versions of this chapter, was abandoned in benefit of a simpler GcRS-SRS conversion. However, this alternative approach included some aspects of the NSL that should be taken into account: For example, the Sun Aspect Angle should take into account the Lissajous effects, this is, the SAA should be fix. The difference wrt a non-inclusion of Lissajous effects is about 0.1 degrees for X_{SP} (an axis with origin in the CoM and always pointing to the Sun). This difference may be high enough for items like the thermal control of the spacecraft.

[10] The orbit of Gaia should be in the barycentric dynamical reference frame, based upon the observations from the ground and upon the barycentric ephemeris of the Earth. This is mandatory for a precise correction for aberration. Note that the stellar aberration correction varies by a quantity that may reach 4 μas/second. This means that the time reference on the orbit, and its equivalent on-board, should be accurate to 0.1 second.

2.2.1.4. Satellite RS – FOV RS

Once we have a body-fixed RS with the SRS, we can obtain each one of the three FOVRSs. Apart of the "one-to-three" conversion (to each one of the FOVRSs), this is one of the most complex RS conversions in Gaia because of the 6-mirror optics. This optics system leads to a "different" coordinate system whether it is "within Gaia" (i.e., after the mirror system) or "outside Gaia" (i.e., before the mirror system, see figure 2.9), so maybe the simplest way to solve this is to obtain two different conversions for each FOVRS:

- **Within Gaia** (after the mirror system): this conversion would lead to a FOVRS as it is in the origin (i.e., thick dashed coordinate systems in figure 2.9). Its valid ranges would be limited by the first optical inversion (where the coordinate system would also be inverted), although the ranges could be extended to refer whichever part of Gaia. This interpretation of the FOVRSs would be useful, for example, to refer some instrumental parts of Gaia (like the focal plane or the optics). However, this interpretation of the FOVRSs will not be used (it is better to use the SRS to refer instrumental parts of Gaia, or even a FPRS to refer a part of the focal plane).
- **Outside Gaia** (before the mirror system): this conversion is the most useful and simple, because it makes possible to refer the sources observed by the several FOVs (and therefore to convert their coordinates from the SRS). This interpretation of the FOVRSs leads to the thin coordinate systems shown in figure 2.9. Valid ranges of these coordinate systems are limited by the last optical inversion, although we recommend to use this interpretation outside Gaia.

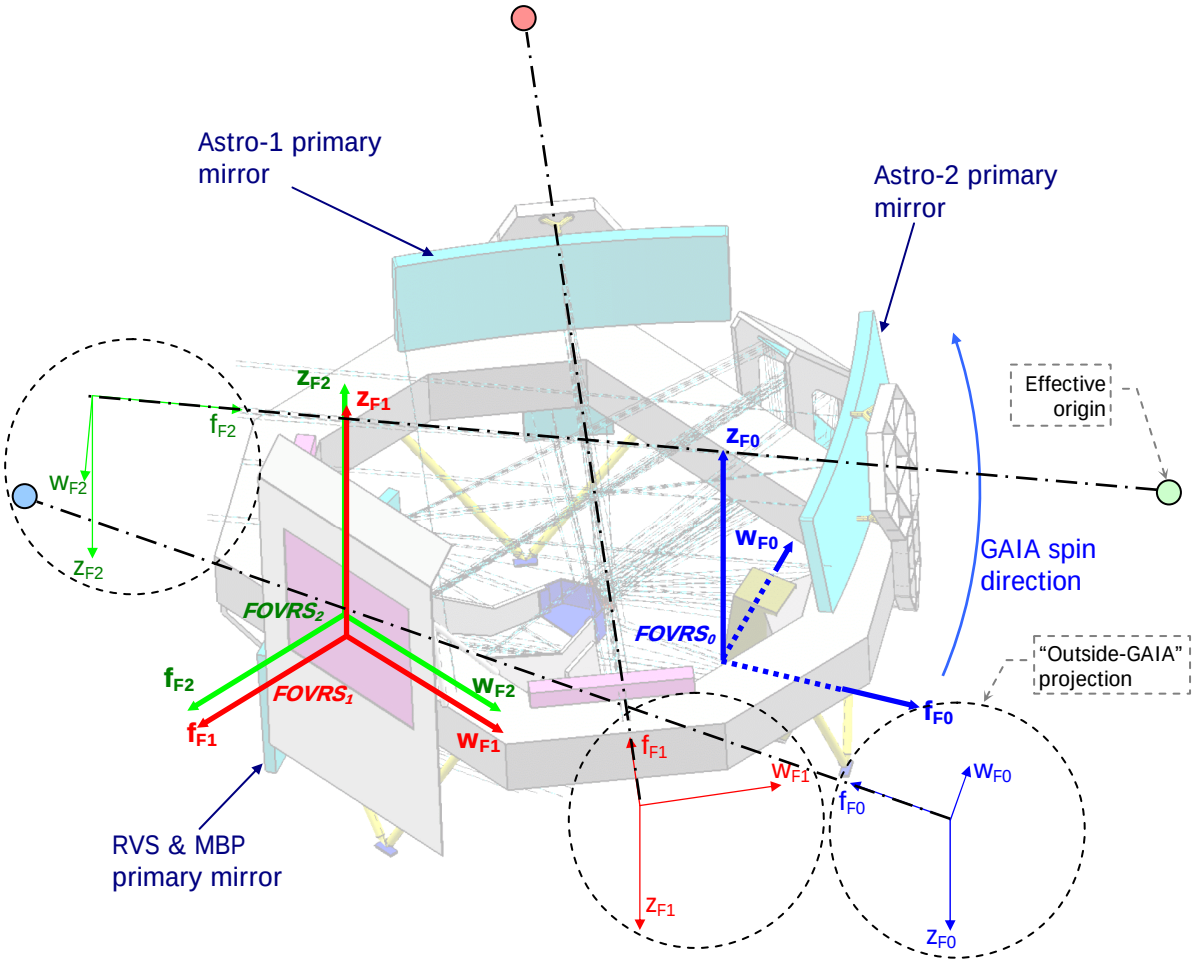


Figure 2.9: "Outside-Gaia" interpretation of the FOVRS

In order to perform these conversions from the SRS to the several "Outside-Gaia-FOVRSs", we will suppose a simple linear optics system like the one shown in figure 2.3 –which will not

affect the validity of the conversion, but will make it much easier. For this, an "imaginary" or effective origin for each FOVRS will be needed, obtained with the focal length of the telescopes. This is shown in figure 2.9. The SRS–FOVRS conversion for a generic instrument (Astro–1, Astro–2 or Spectro) is the following:

- **Translation**: From the center of masses of Gaia towards the corresponding effective origin. As seen in figure 2.9, this origin will be placed behind the primary mirror of the FOV, at a distance fixed by the optics and the focal length (not necessarily equal to this). Translation vector:

$$\vec{d}_{SRS\rightarrow FOVRS}(i) = \left(-o_x(i), -o_y(i), -o_z(i)\right)$$

  where $o_x(i)$, $o_y(i)$ and $o_z(i)$ are the coordinates of the effective origin of FOVRS of instrument i, given in the SRS, to be obtained by the optics engineering team.

- **Rotation**: a simple rotation, depending on the instrument (i), must be done in order to align the SRS axes with the FOVRS axes. The generic rotation matrix $\mathbf{A_{SRS\rightarrow FOVRS}(i)}$ will be *ideal*, while the deformations in the structure should be included by a *deformations noise* matrix $\mathbf{N_D(i)}$.

- **Noise and imperfections**: deformations of the structure, affecting both the translation (with a *deformations noise* vector $d_D(i)$) and the rotation (with the matrix $\mathbf{N_D(i)}$).

- **Time scale conversion**: Mainly a phase shift, due to the transmission delay from the main on-board clock (at SatelliteTime) to each instrument clock.

Finally, we must take into account the different FOV reference systems: one coordinate given in the SRS will be converted to $FOVRS_1$, $FOVRS_2$ or $FOVRS_0$ depending on its space–time coordinates (i.e., depending on the FOV "cone" where the SRS coordinate can be fit in). Also it is possible that, at a given time, an SRS coordinate could NOT be converted to any FOVRS coordinate (within its valid ranges, of course).

2.2.1.5. Focal Plane RS – CCD RS

This is the first conversion in the Payload group of reference systems. The conversion is basically a simple translation of the origin, from the FPRS origin (mechanical center of the focal plane) to the origin of a CCD (upper left pixel).

- **Translation**: From the mechanical center of the focal plane to the upper left pixel of the given CCD. Translation vector:

$$\vec{d}_{FPRS\rightarrow CCDRSi-c:rr} = \left(-CCD_w(i,c,rr), -CCD_z(i,c,rr)\right)$$

  where $CCD_w(i,c,rr)$ and $CCD_z(i,c,rr)$ are the coordinates of the upper left pixel of CCD column c, row rr in instrument i, given in the FPRS.

- **Rotation**: In this 2–D conversion, there is no rotation but a scaling of the axes.
  - Axis w: No scaling, except for the conversion from linear scale (e.g., millimeters) to pixels. This conversion will depend on the calibration of the focal plane, so it will not be as simple as the *ideal* conversion "10μm=1 pixel".
  - Axis z: Linear–pixel conversion (also complicated, depending on the calibration of the focal plane).
  - A "rotation matrix" can also be obtained for this conversion. This matrix, $\mathbf{A_{FPRS\rightarrow CCDRS}(i,c,rr)}$, will obviously be 2×2, and is TBD yet.

- **Noise and imperfections**: CCD tilts and other corrections on the focal plane, in order to calibrate and adapt it to the real optics of Gaia. Also, deformations of the focal plane can affect this conversion.

- **Time scale conversion**: The several CCDRSs will use the same time of the corresponding FPRS, although the delay in the transmission lines (from the Focal Plane clock to its several CCDs) should be taken into account (due to the high precision required).

Finally, we also must note that a coordinate given in the $FPRS_i$ will be converted to the corresponding $CCDRS_i$–*c:rr* depending on its space–time coordinates (i.e., depending on the CCD where this coordinate is located).

2.2.1.6. CCD RS – Space of Measurements

This last conversion is made in a "digital way": the several data given by the CCDRS will be converted, one to one, to the SoM set of data. It is important to note that the SoM includes a new coordinate, the flux, which is not included in none of the reference systems described. This new coordinate will be obtained directly from the measurements made by Gaia.

The following list describes how the several "coordinates" of the SoM are related to the CCDRS:

- *i* (instrument): equals to the instrument indicated by $CCDRS_{\boldsymbol{i}}$–*c:rr*.
- c_w (CCD row, along-scan): equals to the row indicated by $CCDRS_i$–*c:***rr**.
- c_z (CCD column, across-scan): equals to the column indicated by $CCDRS_i$–***c:****rr*.
- p_w (pixel row, along-scan): w coordinate of the corresponding CCDRS.
- p_z (pixel column, across-scan): z coordinate of the corresponding CCDRS.
- *fl* (flux read): this is an independent coordinate (SoM –only).
- t_m (readout time): basically equals to the *time* coordinate of the corresponding CCDRS. The displacement within a CCD seems to be not relevant (TBC).

2.2.1.7. Astronomical – Payload link based on calibrations: SRS–FPRS conversion

This conversion is a practical link between the two groups of reference systems: astronomical ones, and payload ones. The conversion could be based on the way the celestial sphere (or the Field Of View) is projected over the focal plane, and vice versa. However, in practice this conversion will be based on calibrations, as described in Lindegren (2001) and Figueras (2001). Also, the SRS–FPRS conversion is, in fact, three different conversions: one for every instrument (Astro–1, Astro–2 and Spectro), although two of these instruments (or FOVs) will converge onto a single focal plane (Astro). Finally, it is important to remark that this will be a 3–D↔2–D conversion (that is, a kind of projection).

Because of this calibration-based process we cannot obtain a well-defined set of translation vectors, rotation matrices and so on, but a set of calibration equations. This calibration will associate the field angles of Gaia (η,ζ) with positions on each focal plane (w,z) using the following equations:

$$\eta_l^{calc} = \eta_{(w,z)} + \Gamma_{(w,z)} (C_{i(l)} - C_0)$$

$$\Delta\zeta_{(w,z)} = \langle \zeta_l^{calc} - \zeta_l^{obs} \rangle$$

with:

$(\eta_l^{calc}, \zeta_l^{calc})$: calculated field angles of the elementary observation *l* (Lindegren 2001).

ζ_l^{obs} : observed field angle across-scan of the elementary observation *l*

C_i : color index of source *i* in the transit corresponding to the elementary observation *l*

C_0: reference color index

The noise and bias in this conversion will be basically due to non-modeled optical imperfections and to deformations of the structure and the focal plane. On the other hand, the

time scale conversion is mainly a phase shift due to the transmission delay from the main on-board clock (at SatelliteTime) to each instrument clock.

2.2.1.8. Astronomical – Payload link based on models: FOVRS-FPRS conversion

This conversion leads to an alternative link between the two groups of reference systems: astronomical ones, and payload ones. The conversion is based on the way the celestial sphere (FOV) is projected over the focal plane, and vice versa. It is important to remark that this conversion will also be a 3–D ↔ 2–D conversion (that is, a projection). Therefore, the translation vector and the rotation matrix must reflect this.

This model-based conversion will not be "one-to-one", because the $FOVRS_1$ and $FOVRS_2$ are both projected onto $FPRS_A$. Linking the FPRS with one or another FOVRS will depend not only on the FPRS coordinates (e.g., ASM2 can be related only to $FOVRS_2$), but also on SoM-related data (this is, on the instrument linked to the measurement). However, here we will tackle it in a simple way: 3 conversions will be defined ($FOVRS_1 \leftrightarrow FPRS_A$, $FOVRS_2 \leftrightarrow FPRS_A$ and $FOVRS_0 \leftrightarrow FPRS_S$), so that the use of one or another conversion for Astro is up to the user.

- **Translation**: From the optical center of the focal plane to its mechanical center (as seen in figure 2.10). This translation will be of about ½ CCD across-scan and about 1 CCD along-scan (at least for the Astro). The *z* axis will be ignored when converting towards the FPRS. Translation vector:

$$\vec{d}_{FOVRS\rightarrow FPRS}(i) = \left(optc_w(i), optc_z(i)\right)$$

  where $optc_w(i)$ and $optc_z(i)$ are the coordinates of the optical center of instrument *i* given in the FPRS. We should be careful with the use of identifier *i* for the instrument (FOVRS↔FPRS, respectively): '*0*' ↔ '*S*', '*1*' ↔ '*A*' and '*2*' ↔ '*A*'.

- **Rotation**: In a principle, there is no rotation in this conversion (*w* and *z* axes of FOVRS and FPRS are aligned). However, the 2–D projection over the focal plane could somehow be expressed as a 3×2 rotation matrix (better a *projection matrix*). Possibly this $\mathbf{A}_{\mathbf{FOVRS\rightarrow FPRS}}(i)$ matrix will not be linear, and is still TBD.

- **Noise and imperfections**: optical imperfections and calibrations, and deformations of the structure and the focal plane.

- **Time scale conversion**: None (the FOVRS and FPRS clocks are assumed to be the same for a given instrument)

2.2.1.9. SoM – ICRS direct conversion

A direct relation between the Space of Measurements and the ICRS will be very useful, because this is the main objective of Gaia: to obtain accurate measurements of the sources in order to, afterwards, obtain accurate ICRS coordinates for each observed source.

This direct conversion will be made considering calibrations (not model-based, so the FOVRS will be ignored) and in three steps: first of all, the ICRS–SRS conversion will be obtained, by concatenating all the matrices and vectors obtained before. Afterwards, the SRS–FPRS conversion will be obtained by remarking the *instrument* where the SRS coordinates are *located*. Finally, the "digital" conversion to (or from) the SoM will be made. Coordinates in the ICRS are named $\boldsymbol{v}$ and in the SoM are named $\boldsymbol{m}$, while coordinates in the SRS are named $\boldsymbol{s}$ and in the FPRS are named $\boldsymbol{p}$. Time scale conversion will not be treated here.

1. **ICRS–SRS:**

$$\vec{s} = \mathbf{A}_{\mathbf{ICRS\rightarrow SRS}}\left(\vec{v} + \vec{d}_{ICRS\rightarrow GcRS} + \vec{d}_{GcRS\rightarrow SRS}\right)$$

   and the inverse,

$$\vec{v} = \left(\mathbf{A}_{\mathbf{ICRS}\rightarrow\mathbf{SRS}}\right)^{-1}\vec{s} - \vec{d}_{ICRS\rightarrow GcRS} - \vec{d}_{GcRS\rightarrow SRS}$$

plus the corresponding corrections, noises, etc.

2. **SRS–FPRS$_i$:**

   These coordinates will be converted to $FPRS_A$ (if $\boldsymbol{s}$ values correspond to valid Astro ranges) or to $FPRS_0$ (if $\boldsymbol{s}$ values correspond to valid Spectro ranges). Once the "destination RS" is selected, we will have to apply the calibration-based conversion, as described in 4.1.5.

3. **FPRS$_i$– SoM:**

   This *digital* conversion will be based both on the values of $\boldsymbol{p}$ and $\boldsymbol{s}$. The different components of $\boldsymbol{m}$ will be obtained the following way:

   - i (instrument): the suffix i of $\boldsymbol{p_i}$ is not enough, so we will have to look at the values of $\boldsymbol{s}$.
   - c_w and c_z (CCD row and column): depending on the CCD where $\boldsymbol{p_i}$ is located. This could be calculated with an integer division (i.e., the corresponding coordinate divided by the dimensions of a CCD).
   - p_w and p_z (pixel row and column): depending on the pixel where $\boldsymbol{p_i}$ is located. This could be calculated with the modulus obtained when dividing the corresponding coordinate by the dimensions of a CCD.
   - *fl* (flux read): independent coordinate (SoM –only).

   The inverse transformation can also be done, by multiplying correctly each component of $\boldsymbol{m}$. In a sense, i would be the "most-weight" component, c_w and c_z the "middle-weight" components (to be multiplied by the size of a CCD, given in the FPRS), and p_w and p_z the "less-weight" components (to be multiplied by the size of a pixel). Therefore, this conversion could be made somehow with a formula like this:

$$\vec{p}_i = CCD\_size_w \cdot m.c_w + CCD\_size_z \cdot m.c_z + pixel\_size_w \cdot m.p_w + pixel\_size_z \cdot m.p_z$$

2.2.2. Some useful approximations

In the previous section we have introduced the several formulae to be used in order to implement the "full" conversions between reference systems, including imperfections, calibrations and noises. Here we will provide some useful data in order to be able to obtain a first approximation to the transformations.

2.2.2.1. Simplified Gaia orbit

From the barycenter, the distance to the center of masses of Gaia can be approximated to 1AU+1.5 million kilometers, this is, about 151.5 million kilometers. This will lead to simpler components of $\mathbf{d}_{\mathbf{ICRS}\rightarrow\mathbf{SRS}}$, which will be approximately sinusoidal.

2.2.2.2. SRS–FPRS conversion (astronomical–payload link)

The tilt for the CCDs can be ignored in a first approximation. They can be supposed to be placed uniformly all over the focal plane. Also, as a global approximation (not only for the SRS–FPRS case), the optical-linear conversion can be assumed to be fixed (see Portell et al. 2003a): 1 millimeter (in the focal plane) = 4.424 arcsec (in the FOVs).

2.2.2.3. FPRS–CCDRS conversion

CCD and pixel sizes, as well as the sizes of dead zones between CCDs, can be assumed to have the dimensions given in Portell et al. (2003a). This is, the along-scan dimension of 1 pixel can be assumed to be fixed and equal to 10μm, and the across-scan dimension equal to 30μm.

2.3. CONVENTIONS AND NOTATIONS

This section will present a set of conventions that we recommend to use in the forthcoming documents of Gaia, in order to homogenize them and prevent misunderstandings.

2.3.1. When to use every RS

Here we recommend when one reference system or another should be used:

- **ICRS**:
  - Ephemeris given by the ESOC, BDL, etc. in this standard RS.
  - Absolute coordinates for a stellar object.
- **GcRS**:
  - Ephemeris given by the ESOC, etc. in this standard RS.
  - Coordinates of an Earth object.
  - Coordinates of a stellar object as observed from the ground.
- **SRS**:
  - Coordinates of a source *as seen from* Gaia.
  - Coordinates of an element of Gaia (payload modules, focal planes, etc.). This includes the definition of the shape of some element (e.g., the curvature of mirrors or focal planes).
- **FOVRS**:
  - Coordinates of a stellar object *as seen by* Gaia.
- **FPRS**:
  - Coordinates of a stellar object *as measured by* Gaia.
  - Coordinates and sizes of CCDs, dead zones, pixels… on a focal plane.
- **CCDRS**:
  - Coordinates of particular sources, as seen by Gaia, on a given CCD.
  - Problems on a CCD (e.g., failures on the read of a pixel line, etc.).
  - Definition of windowing, sampling, etc. for the readout operation.
- **SoM**:
  - Final measurements made by Gaia, specially the final digital value.

2.3.2. Conventions

The conventions adopted in this work are the following:

- As seen in previous sections, the row/column convention adopted is the industrial one. This means that a row (CCD row or pixel row) stands for an across-scan line, while a

column (of CCDs or pixels) stands for an along-scan line. In other words, the TDI mode shifts charges from one pixel row to the next. Therefore an astrometric focal plane, as seen in figure 2.10, has 18 CCD rows and 10 CCD columns, and a CCD has 4500 pixel rows (in the AF) and 1966 pixel columns (as described in Portell et al. 2003a). If there is a possibility of misunderstandings, then some notation like AL and AC (ALong-scan and ACross-scan) should be used.

- The sampling considered for the astrometric focal planes is the described in Høg 2002b.
- The dates will be expressed as YYYYMMDD, so that it can be easily sorted. If a field separator is needed (e.g., to be easily human readable), it may be expressed as YYYY/MM/DD.

2.3.3. Notations and nomenclature

In order to refer the several parts of Gaia, we recommend using the following short names:

- **Fields of View**: if Astro−1 (preceding), Astro−2 (following) and Spectro identifiers are too long, we can name them as fields 1, 2 and 0 (respectively). These will be the indexes used when referring to the instruments in formulae, etc.
- **Instruments (focal planes)**: if Astro and Spectro identifiers are too long, "A" and "S" can be used. If RVS and MBP zones of the Spectro focal plane must be distinguished, then "R" (RVS) and "M" (MBP) could be used instead.
- **CCDs**: to refer a CCD on a focal plane we can use the same notation used in the FPRS. This is, *c:rr*, where *c* is the CCD column, from 0 (top of figure 2.10) to 9 (bottom of figure 2.10), and *rr* is the CCD row, ranging from 00 (ASM1) to 17 (BBP5). However, we can keep using the ASM/AF/BBP notation (e.g., CCD 2 of AF08, which could also be noted with CCD 2:09).
- **Pixels**: to refer a pixel on a CCD we can use a notation similar to that one used in the CCDRS. This is, *pc:pr*, where *pc* is the pixel column and *pr* the pixel row, ranging from pixel 0:0 (upper left pixel, in figure 2.10) to pixel 4499:1964 (lower right pixel, in AF of figure 2.10). The use of a colon (:) instead of a point makes possible the use of decimals, if needed (e.g., to refer the centroid of a source: pixel 241.7:1180.1).
- **Instrument+CCD+pixel**: In order to refer a pixel (or sub-pixel) in the whole set of Gaia, one can use the following nomenclature:

  i-c:rr-pc:pr

  using the same acronyms described before. This way, we can refer, for example, to:

  1-6:11-382.5:1342.7

  which is exactly between pixel columns 382 and 383, pixel row 1343 (not in the center, but 3/10 of pixel towards row 1342) of CCD column 6 of the AF10, in Astro (projected by the preceding field).

  Also, if we have to refer a pixel on a focal plane (and the projecting field does *not* matter), we can use the alternative acronyms for the instruments: "A" for the astrometric focal plane and "S" for the spectrometric focal plane (or even "R" for the RVS and "M" for the MBP). In the previous example we could use:

  A-6:11-382.5:1342.7

  This is a way to simplify the expressions in the SoM: we can refer to a pixel in this way, and afterwards we only have to say the flux detected there, and the measurement time.

2.4. SUMMARY OF REFERENCE SYSTEMS

Ref. System	Origin	Axes	Alternat. coords.	Units	Ranges	Precision	Time scale	Use
ICRS	Barycenter	3, determined by corresponding organizations	$(\alpha,\delta)_{ICRS}$ (or RA and DEC), and r (or π).	Linear (parsecs recomm.) or angular	Infinite	As high as possible	Barycentric Time	Ephemeris of Earth, Barycentric coordinates
GcRS	Geocenter	3, parallel to ICRS	$(\alpha,\delta)_{GcRS}$ (or RA and DEC), and r (or π).	Linear (pc recomm.) or angular	Infinite	As high as possible	Terrestrial Time	Ephemeris of satellite, coordinates from Earth
CoMRS	Center of masses of Gaia	3, parallel to ICRS	(α,δ) (or RA and DEC), and r (or π).	Linear / angular	Infinite	As high as possible	Satellite Time	Conversion step / Gaia apparent angles
SRS	Center of masses of Gaia	3, body-fixed (one towards Astro mech. center, another towards "Gaia north pole")	Field angles (η,ζ)	Linear / angular	Infinite	As high as possible	Satellite Time	Absolute coords. as seen by Gaia / element of Gaia (payload, etc.)
FOVRS$_i$	Optical center of focal plane	3, body-fixed (following ray trace, tracking sources, towards image of sun)	Field angles $(\eta,\zeta)_i$	Angular / linear	Fixed by FOV cone	1nm recomm.	Satellite Time + phase shift	Sources coords. as seen by Gaia instruments, telescope-like
FPRS$_i$	Mechanical center of focal plane	2, fixed to focal plane, one tracks sources	Engineering axes $(x,y)_i$	Linear (mm recomm.)	Fixed by focal plane size	1nm recomm.	Satellite Time + phase shift	Coords. of sources as measured by Gaia / Elements of focal plane
CCDRS*i–c:rr*	Center of upper left pixel (as seen in fig. 2.5)	2, fixed to CCD, continuing FPRS$_i$ directions.	Engineering axes $(x,y)_{i\text{-}c:rr}$	Pixels	Fixed by CCD size	$1/10^7$ pixel recomm.	Satellite time + phase shift	Elements on a CCD (sampling, windowing, failures…)
SoM	Not applicable	6, digital: i, c_w, c_z, p_w, p_z, fl	–	Intrinsic to axes (CCDs, pixels, e^-…)	Intrinsic to axes	Intrinsic to axes	Satellite Time + phase shift	Final measurements made by Gaia (digital value)

Table 2.1: Summary of reference systems for Gaia

2.5. ASTROMETRIC FOCAL PLANES OF GAIA

Here the structure of the astrometric focal plane (Astro) is shown in a global scheme, including the definition (and sizes) of the main samples and patches used in the readout of the CCDs as proposed in Høg 2002b. Figure 2.10 shows the focal plane as seen from the illuminated side (this is, from the last mirror, M6). The spectrometric instrument is not shown.

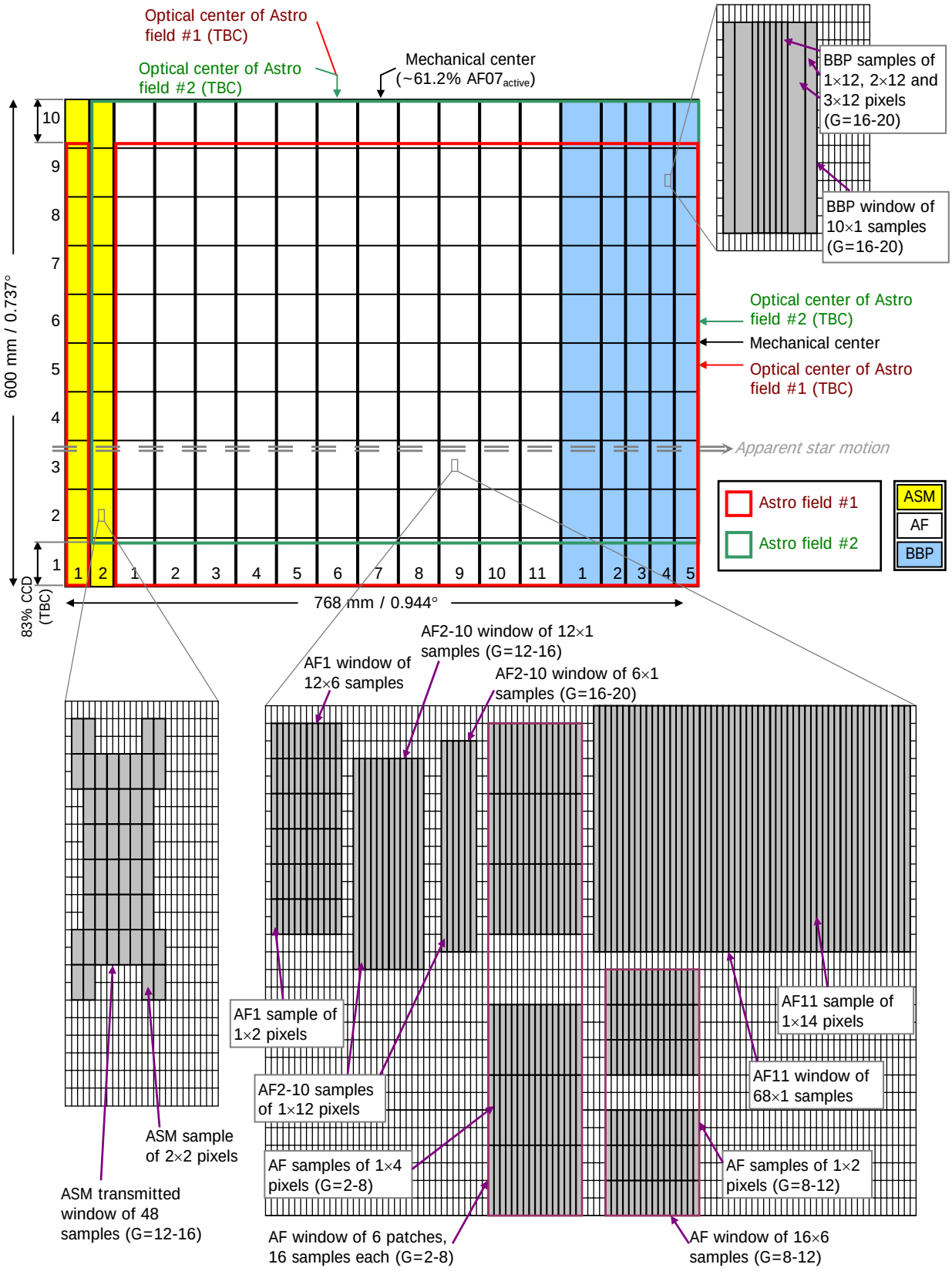


Figure 2.10: Astro focal plane layout and sampling

Chapter 3

Description of the instruments

This chapter contains a compilation of the main characteristics and specifications for the Gaia astrometric focal plane (as of April 2002, which is the reference we have used for our work), put all together for handy use. The spectrometric instrument is just mentioned, but it is not fully described here. Most of the issues treated here refer to the configuration of the focal plane (how the FOV is organized) and its sizes (in linear size, angular size and transit time). It includes the adaptation from the original design of Gaia to the new design presented in EADS/Astrium (2002). Also, the sampling scheme proposed in Høg (2002b) has been taken into account. Although some of these issues may have changed due to small redesigns in the mission and the spacecraft, we have considered them as the baseline for Gaia in our work.

Our purpose in this chapter was to prepare a small manual of the main characteristics of the focal plane, and to provide as well some general characteristics of the instrumentation and of the global concept of the mission in general. The characteristics described here have been used as the baseline in the GDAAS-II study.

In this chapter the detailed descriptions of the instruments can be found. Sections 1 and 2 present the main concepts of Gaia, its operation and its payload (specially the astrometric focal plane). Annex A, on the other hand, includes a tabulated summary of all the main features in order to find them more easily once they are well understood.

3.1. GENERAL CHARACTERISTICS

The purpose of the Gaia mission, which is to get an extremely precise three-dimensional map of our Galaxy and of the local group (including photometric, astrometric and spectrometric data), will be carried out with a payload module that includes two astrometric telescopes (converging onto a single focal plane), a medium-band photometer and a radial velocity spectroscope. An adequate orbit and scanning law will be needed as well. In this section the details of these main characteristics of the mission are described and summarized.

3.1.1. Scan Strategy

The observation mode will be a continuous sky scanning, that is, Gaia will scan systematically the whole sky from the beginning of the mission to its end. This will be done from an orbit around the L2 Lagrangian point, which is at about 1.5×10^6 km from the Earth opposite to the Sun. The Sun Aspect Angle (SAA), which is the angle between a vector pointing towards the Sun and the spin axis of Gaia, is 50 degrees (see figure A.1). The scan rate will be 60″/s (with an absolute scan rate error of 1.95mas/s). That is, Gaia will scan 60 arcseconds of the sky every second of time. It means a revolution every 6 hours:

$$\frac{1\,revolution}{scan\,rate} = \frac{360°}{60'' / s} \cdot \frac{3600''}{°} \cdot \frac{1^m}{60^s} = 360\,minutes$$

This result is equivalent to $24^h/6^h = 4$ revolutions every day and, consequently, 4 *great circles* scanned per day. However, Gaia will not only turn just around its own axis, but it also will do a precession motion (ESA 2000, p. 151), completing one revolution in 70 days. This precession (always maintaining the SAA at 50°, ±0.1°) leads to an apparent motion of the stars over the focal plane (in addition to the longitudinal motion due to the spin motion). The maximum across-scan speed due to precession is approximately 171mas/s, that is, 0.171°/h (1° = 3600″,

$1^h = 3600^s$). It means that the nominal spin axis will move about 1.03° every revolution (1 revolution needs 6^h to be completed). The absolute precession error rate required is 0.1arcsec/s.

This selected scanning strategy leads to a non-uniform sky coverage. As noted in ESA (2000), p. 268, every object in the sky will be scanned by Gaia a different number of times, depending on its ecliptic latitude. With the current design, the average number of focal-plane passages per astrometric telescope will be about 41, so about 82 observations (in average) will be performed for each star (taking into account the two astrometric telescopes). This non-uniform sky coverage also means a non-uniform data generation, because the galactic plane will not be scanned uniformly. Figure 3.1 shows two typical scanning periods of 10 days in galactic coordinates, one of them (left) with low data generation (because of the almost-perpendicular passages through the galactic plane), while the other one (right) will generate a lot of data (because of the almost-tangential passages through the galactic plane, and therefore lots of stars will be scanned during a long time). These figures were obtained using the old design (ESA 2000), but they would look similar with the new one (EADS/Astrium 2002).

Finally, we have to take into account the attitude adjustment processes in this scanning law. It will include continuous attitude corrections (thanks to the measurements made in the ASM and in the Wide Field Star Tracker), as well as large attitude adjustments made eventually, which may lead to discontinuities in this scanning law. However, these processes are not yet taken into account in the simulations (which includes GDAAS).

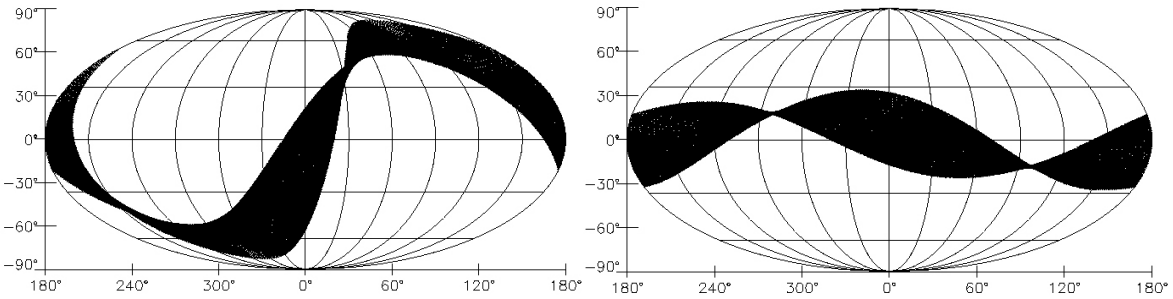


Figure 3.1: Area scanned by Gaia in a 10 days interval time

3.1.2. Payload Summary

The Gaia payload module contains one astrometric instrument (named Astro), and one radial velocity spectrometer and medium band photometer (named Spectro). The astrometric instrument consists of two telescopes (Astro-1 and Astro-2) converging into a single focal plane. The angular separation between the axes of the two astrometric telescopes (named *basic angle*) was 106° in the baseline, although newer designs point towards a value of about 99.4°. Since the final design of Gaia may eventually change this value again, in our study we have taken the initial value of 106° for our calculations.

The other instrument, Spectro, consists of a three-mirror anastigmatic (TMA) telescope, which directs the central part of the field to the radial velocity spectrometer and the outer parts to the medium-band photometer (ESA 2000, p. 168). The FOV of this instrument is not centered between the FOVs of the two astrometric instruments: Astro-2 points 74° (about 1.23 hours) before Spectro, and Astro-1 points 180° (3 hours) after Spectro (wrt the scanning direction).

Figure A.2 shows the operation of this payload set. The nominal operation is the following: first, the source being observed enters the FOV of Astro-1 and is measured; 106 minutes after, the same source enters the FOV of Astro-2 and is measured. Finally, 74 minutes after, it enters the FOV of the Spectro instrument and it is measured by this instrument. Whether or not this object is observed again by Astro-1 (3 hours after entering Spectro), and even its complete scanning in one, two or the three instruments in a row, depends on its across-scan position in the FOV. This is the way Gaia scans the sky, but an object may not follow this sequence due to the precession motion: a star could enter the Astro-1 FOV, then the Astro-2 FOV and 74

minutes after be out of the Spectro FOV (or get out while being measured by Astro-2). The time needed to point one FOV after another has been calculated with the following expression:

$$Time = \frac{Angle}{Angular\,velocity} = \frac{Angle(^\circ)}{60''/s}\frac{3600''}{^\circ}\frac{1^m}{60^s}$$

Therefore, Time(minutes)=Angle(degrees), or Time(seconds)=Angle(arcmin). Please note that all these angles and times in the scan sequence of Gaia are relative to the *centers* of a FOV, not from the *end* of a FOV to the *beginning* of the next FOV.

3.2. ASTROMETRIC FOCAL PLANE

All dimensions shown here, as well as in the summary, are noted as width × height. Here "width" stands for "along scan", that is, the direction of the apparent motion of the objects on the focal plane due to the spin motion (which is from left to right in figure 2.10), and "height" stands for "across scan" (perpendicular to scan direction).

3.2.1. General

As shown before, the astrometric measurements are obtained from the combination of two astrometric telescopes, Astro-1 and Astro-2, converging into a unique astrometric focal plane (Astro). The sections (or parts) of the focal plane, whose structure has already been shown in figure 2.10, are the following:

1. The Astrometric Sky Mapper (ASM), in charge of detecting the objects to be measured and determining the way it should be done.
2. The Astrometric Field (AF), which is the astrometric instrument itself. It measures several times and with high resolution the passage of every object along its surface. AF01 (the first column of the AF) is a special case: it confirms the sources entering the focal plane (thus rejecting false ASM detections), but at the same time offers a high-resolution measurement.
3. The Broad Band Photometer (BBP), which is in charge of measuring the photometric features of every object. These measures will be done in a broad-band system, just measuring the flux (and shape) received from every object in 5 spectral bands (with one different filter on every column of the BBP).

The focal plane technology is CCD-based (18×10 array), operated in TDI mode synchronized with spacecraft rotation. That is, the sky is scanned with the CCDs as it passes over the focal plane: the charge (flux from sources) is shifted from one pixel to another within the same CCD matching the stars speed, thus accumulating the same charge obtained with a standard CCD mode at ~1 second exposure time, but with a continuous scanning. The problem of this technique is that a perfect synchronization with spacecraft rotation is needed: If not, blurring would appear on the measurements.

The focal plane size, including the three sections (ASM+AF+BBP), is 768 mm wide (along-scan) and 600 mm high (across-scan). The telescopes lead to an angle/linear scale of 226.26 μm per arcsec (in average). With this relation, the angular size of the FOVs can be found: 0.944°×0.737°, taking into account both FOVs (i.e., the angular size of the whole focal plane).

3.2.2. Optics

The telescopes used in the astrometric instrument use a primary mirror (one for every FOV) of 1.4×0.5 m^2. Both FOVs are then combined using a 6-mirror system, 3 of which are common to the two telescopes. This structure leads to a focal length of 46.67 m and an optical

transmission higher than 0.86. An overview of the payload from two different angles, including optical ray traces, is shown in figure A.3 (Appendix A).

3.2.3. Timing Requirements

The high precision required in the Gaia mission implies also an extremely high precision in the timing and clocks on the spacecraft: a simple delay or de-synchronization between two instruments would imply high precision losses in the measurements. Therefore, the relative time accuracy should be about 1 μs over 10 s, and as a consequence the maximum drift of the clock in 10s has to be lower than 1μs. In absolute terms, the maximum clock drift accepted is 5 μs. Also, the temporal stability of the sequencing over 100 s must be less than 10ns. These requirements have been extracted from ESA (2000), p. 184, and they should be re-validated for the new design (EADS/Astrium 2002). Further details on the timing requirements (specially on the master clock) can be found in De Bruijne 2003b.

3.2.4. CCD Specifications

3.2.4.1. Sizes and dead zones

The pixel size of the CCDs is 10×30 μm^2 (or 44.2×132.6 mas^2). Its ratio (1:3) has been selected in order to match the PSF aspect ratio (which is also 1:3). The CCD size in pixels is 2600×1966 for a "narrow chip" (ASM and BBP2-5), or 4500×1966 pixels for a "wide chip" (AF and BBP1). Taking into account the size of a pixel, then the size of one CCD is 26×59 mm^2 (just for the active area or pixel area) for narrow chips, or 45×59 mm^2 for a wide chip. The angular size can be easily found with the relation given in 3.2.1.

For an array like the one used in the astrometric focal plane, it is important to consider the dead zones (the space between one CCD and the next one). Their size are <4 mm along-scan and <1 mm across-scan. Taking into account the size of a pixel, these dead zones are equivalent to <400 inactive pixels between one CCD and the next one (in the along-scan direction), and <33.3 inactive pixels across-scan. It leads to fill factors (ratio between the active area and the total area) higher than 86.6% and 91.8% along scan, respectively (for narrow and wide chips), and 98.3% across-scan. Taken as a whole, it leads to a global fill factor of >85.2% for narrow CCDs, and >90.3% for wide CCDs.

3.2.4.2. TDI rates

All the CCDs in the focal plane will operate with a single TDI rate. In principle, all of their clocks will operate as a single one (TBC), that is, the pixels of all the CCDs will transfer their charges at the same time. This TDI rate (or TDI period) must be perfectly synchronized with the spin rate (to avoid a blurring effect on the acquired images): its nominal value is 736.6μs, which is the time between two successive charge transfers (from one pixel column to the following). With this rate, the equivalent integration period corresponding to one CCD is 1.92 seconds (narrow CCDs) or 3.31 seconds (wide CCDs).

3.2.4.3. Effect of the precession motion

The precession motion of the spacecraft leads to an additional "apparent motion vector" on the focal plane, the direction of which will depend on the spin phase. The nominal maximum of the apparent across-scan motion induced by it is 171 mas/s, which equals to 1.285 pixels per second on the focal plane.

3.2.4.4. Saturation levels

A CCD device does not have an unlimited capacity: when it is exposed to a bright enough source its pixels will saturate –and thus they will not continue accumulating charge. This is due to the limited well capacity of the pixels, which is the number of electrons that one pixel is able to store. Also, one can use some kind of binning (i.e., integration of the charge of several pixels into another one, usually with a higher capacity); if this is the case, there will be another limitation due to the capacity of this "bigger" pixel: in some cases it is not able to store the addition of the full well capacity for each and every one of the pixels binned, because the CCDs to be used in Gaia have a built-in across-scan binning of up to 12 pixels.

The CCDs used in the Gaia focal plane have a full well capacity of 190000 e^-. However, binning is always used in the whole focal plane (although with different binning sizes), so the restriction will come from the capacity of the *big pixel* (named *sample*), which is 350000 e^- (although output FET saturation will be reached at about 300000 e^-). The following are two security systems to be implemented in the CCDs:

1. **Anti-blooming drains** are implemented at pixel level in all the CCDs. These drains, which can be seen as vertical (across-scan) cables between each pair of pixels, are connected to ground in order to avoid contamination of neighbor pixels due to the saturation of a pixel. Therefore, if saturation occurs, it will just imply a non-linear measure in the saturated pixels, and will not affect pixels next to them.
2. **Selectable gate phases**, although their use is not confirmed yet, they use the same principle than anti-blooming drains, but with a wider separation between them (anti-blooming drains will be separated just one pixel width) and draining all of the charge of the pixel (i.e., resetting it). In addition, they are not permanently connected to ground: their connection is controlled by the video chains, thanks to the flux information given by the ASM. If one bright star is detected in the ASM, an adequate drain will be connected to ground some time later causing a reset of the charges accumulated at a given row of the CCD, and therefore restarting the measurement at that point. These selectable gate phases can be seen as a selectable TDI scan time; the only problem is that this time cannot be selected only for the bright object: it will also affect the other objects on its same vertical.

3.2.5. Parts of the Focal Plane

First of all, in order to fully understand all the features of every part of the focal plane and their operation, a few concepts have to be defined here:

1. **Sample size**: Not all the pixels of all the CCDs are read independently, because of the high number of resources that it would imply as well as the faster speed needed to read all the data (and therefore the higher readout noise, see below). Instead of this, pixels are binned (or grouped) and readout together as a single *sample*. This binning is made internally by the CCD. As a counterpart to the lower resources needed and the lower readout noise, this binning implies a loss in the resolution, which will depend on the sample size (e.g., a sample size of 2×2 pixels leads to a quarter of the real CCD resolution).
2. **Patch size** and **window**: Not all the pixels of every CCD are read, but just some relevant groups of them (except on ASM, where all the –binned– pixels are read). Following the instructions given by the Selection Algorithm, the focal plane sequencers will read only the pixels around a detection. This does not mean that the TDI process will be done only on that group of pixels: the TDI operation (charge shift from one pixel to its neighbor) is done in the whole focal plane. The patch or window is selected at the end of the CCD (when all the TDI is done for those pixels), in the readout register. There, the pixels to be discarded are flushed at a higher speed than the readout speed, and the "useable" pixels are binned and read.

3. These "useable" pixels, or "acquisition window", are then sent to the preprocessing electronics, to the on-board processing system and later to ground (if possible). It is important to note that the window/patch size is given in *samples*, not in *pixels*: a patch size of 6×1 means that a group of 6 *samples* (along-scan) are read, not that 6 pixels are read. The total number of pixels read ("acquisition window") can be found multiplying the sample size and the patch size (but taking into account that individual pixels cannot be extracted from the sample).
4. The difference between *patch* and *window* is mainly the following: a *patch* is a one-dimensional set of samples (usually in the along-scan direction), while a *window* is a group of *patches* (usually across-scan). Thus, a patch (combined with its samples size) will contain data related to a "connex readout area", while a window may contain non-contiguous patches (like `WYn`-style windows, see Annex A or Høg 2002b) and thus *may* be related to a "non-connex readout area".
5. **Read-Out Noise (RON)**: The electronic noise (uncertainty) associated to the measurement of a sample. This noise increases with the read-out speed of the pixels: if all the pixels of a CCD have to be read within a fixed time (e.g., the TDI time) the RON will be much higher. On the other hand, if pixels are binned and packed into patches, RON will be lower. Therefore, this noise depends on the number of pixels read, so if sample and patch sizes are fixed the noise will depend on the number of patches read at a time.

The sample and patch sizes given in the following subsections, as well as the RON data, are obtained from Høg (2002b) and Høg et al. (2003).

3.2.5.1. ASM (Astrometric Sky Mapper)

The ASM, which is in charge of detecting the objects entering the FOV of Gaia, is 2 CCDs wide per 10 CCDs tall. The first column (across-scan strip) of CCDs detects sources entering Astro-1 FOV, while the second column detects objects entering Astro-2 FOV. The sample size used in the whole ASM is 2×2 pixels, and all the samples are readout (although just a selection of confirmed detections may be transmitted). Further details on their operation, as well as a redundancy scheme, can be found in Høg et al. (2003). The RON is 8.7 e^- per sample.

3.2.5.2. AF (Astrometric Field)

The AF is the central part of the astrometric focal plane: it is in charge of measuring 11 times the object image when it crosses over its FOV, in order to obtain a high precision shape measurement. Therefore, it has to be 11 CCDs wide (and 10 CCDs high). It is important to note that its last CCD column (AF11) is intended to perform an "extra" measurement (background of the source, etc.), so only 10 columns will be used for "high-quality" measurements. Also, the first column (AF01), apart from measuring the source with a high quality, performs a "confirmation" task as well. The total integration time for these 10 CCDs is 33.1 seconds per passage. The sampling scheme used in the AF can be found in Høg et al. (2003). The total RMS read-out noise (RON) obtained in AF02 to AF10 is 3.7 e^- per sample, while in AF11 it is 5.1 e^- per sample due to the larger acquisition window. AF01 RON is also higher (because more windows must be acquired, some of which will be rejected as false detections): 9.7 e^- per sample.

3.2.5.3. BBP (Broad Band Photometer)

This last part of the focal plane, which is in charge of measuring the flux of every object in 5 different wavelength bands, is 5 CCDs wide (one per band) per 10 CCDs high. BBP1 uses wide CCDs, while BBP2-5 use narrow CCDs. Details on their sampling scheme can be found in Høg et al. (2003). Their RON is 4.5 e^- per sample.

Chapter 4

Operation of the astrometric instrument

On-board detection and selection rules in the Gaia mission, as well as object tracking on its focal plane, will be critical in order to obtain correct information for every object to derive high precision astrometry and to provide a complete and unbiased catalogue. This chapter contains the main requirements needed for a correct detection and tracking operation of every object. Therefore, the Astrometric Sky Mapper (ASM) and its outputs will be analyzed in depth here, as well as the selection algorithm. Some details on the Astrometric Field (AF) and the Broad Band Photometer (BBP) will also be given. On the contrary, the Spectro instrument is not discussed here. Generally, some hints and proposals on the detailed operation of these systems will be given here, as well as some additional ideas and new concepts. These definitions have been partly used for the Gaia simulator, as well as in the GDAAS prototype and other studies.

Section 1 of this chapter reviews several concepts that must be clearly defined and understood before entering in more details and designs. In sections 2 and 3, the detection (ASM) and selection systems are analyzed and some of their expected outputs are presented. The AF and BBP are also analyzed –specially the tracking of sources along them. Section 4 gives some hints on the types of simulations that may be required in order to cover as many situations as possible. Finally, section 5 draws our conclusions on these issues.

4.1. GENERAL CONSIDERATIONS

4.1.1. Objects to be observed

The primary objective of Gaia is to obtain accurate astrometric data of many galactic stars. Gaia will do astrometry only in point-like or almost point-like objects, and maybe in slightly extended objects. On the other hand, one should think about what to do with the rest of the objects detected by Gaia, so an on-board object classification algorithm is needed:

- **Single point-like objects:** Stars, quasars and other objects with an extension of a few pixels (i.e., 2×2 for the brighter part of the PSF) when projected on the focal plane. These objects, whose image spatial extension will be due only to the PSF, will be measured accurately by all the sections in the astrometric focal plane (ASM, AF and BBP), in order to get their position, magnitude, apparent motion on the focal plane, and broad band photometry.
- **Multiple point-like objects.** It includes not only the real binary stars, but also *optical* binaries (i.e., two distant stars very close in the line of sight), fitting in a 6×12 acquisition window when projected on the AF. Also, multiple objects can be considered here (more than two stars or point-like objects very close in the line of sight) if this 6×12 restriction is fulfilled. The observations to do with these objects are, more or less, the same that with point-like objects, but taking into account the visual proximity between them. Therefore, the ASM will have to detect them and instruct the AF on how to perform their tracking (most probably with a bigger acquisition window).
- **Slightly extended objects:** Non-point-like objects, but fitting in a 6×12 pixels window when projected in the focal plane (e.g., distant galaxies or solar system bodies). It could be interesting to use an extended tracking window (12×12 pixel) for these

objects, as well as special tracking operations, but this is not currently considered in the baseline.

- **Very extended objects.** These objects do not fit in a 6×12 pixels window in the AF (e.g., planetary nebulae, diffuse objects, etc.), so there is not much study to do with them. At most, some of them could be studied in the last AF (AF11) and in the BBP, but in the GDAAS prototype they will not be considered at all (i.e., an ASM detection of one of these objects will be ignored).

Besides this classification of objects, the observations (or operation modes) in Gaia can also be classified in one of the following groups:

- **Low and medium density fields**. This will be the normal operation of Gaia, where the instruments will be able to process almost all of the objects, as well as sending their data to the ground station.
- **Crowded fields.** When a FOV of an instrument of Gaia scans a high-density area (i.e., the Galactic Centre or the Galactic Plane), the number of objects to be processed will be too high for the processing capability of the systems, specially for the telemetry capability. Therefore, the kind of job that can be done with these very dense fields is, simply, trying to obtain partial information of their main constituents. That is, to measure only the brighter stars (this criterion is TBC). However, sometimes this could be impossible due to saturation on the detection level (hardware level), and therefore some very dense fields may not be measured at all by Gaia.

4.1.2. Crowded field limitations

All the systems in Gaia should be correctly sized in order to do not lose relevant information. The number of stars entering the focal plane can vary in several orders of magnitude from one area of the sky to another. We can identify here three *layers* of star data processing, each one to be sized in a different way:

1. Electronics, DSPs, hardwired logic, etc., that manage and read the data directly from the CCDs and the video chains.
2. On-board data memory, which is in charge of storing all the data before transmitting them to ground.
3. Telemetry. This will be the most restrictive layer because its capacity cannot be easily increased – a faster telemetry link would imply a more expensive antenna and higher power consumption.

4.1.2.1. Hardware-level sizing: number of windows to be read in the AF

The first layer is the most important one to size correctly: we need to detect all of the stars in order to determine what do we send to ground and what we do not without introducing any selection bias, as well as what do we store on-board and what we do not. As shown in ESA (2000), p. 176, an average star density of about 25000 stars per square degree is expected, but with a maximum of about 3 million stars per square degree; consequently, the hardware of Gaia should be able to manage this density range. Also, in order to avoid potential fluctuations in the power dissipation of the astrometric CCDs (due to the large fluctuations in the star density), it is mandatory to read always the same number of acquisition windows in the AF (and discard the useless windows in low density areas). This should be taken into account in order to simulate more accurately the real satellite behavior.

An issue also noted in ESA (2000) is a recommended constant capacity of 12 simultaneous windows read per CCD. Actually, this capacity was not enough in the original design when considering a maximum star density of 3 million stars per square degree. Even assuming the best case (see figure 4.1), the maximum star density that could be scanned was about 2.9

million stars per square degree –which would imply acceptable losses anyway. But since in the new design Gaia scans the sky at half the original spin rate (i.e., 60 arcsec/s instead of 120 arcsec/s), and furthermore the number of simultaneous windows acquired in the CCDs has increased, the hardware problems have disappeared under almost any condition.

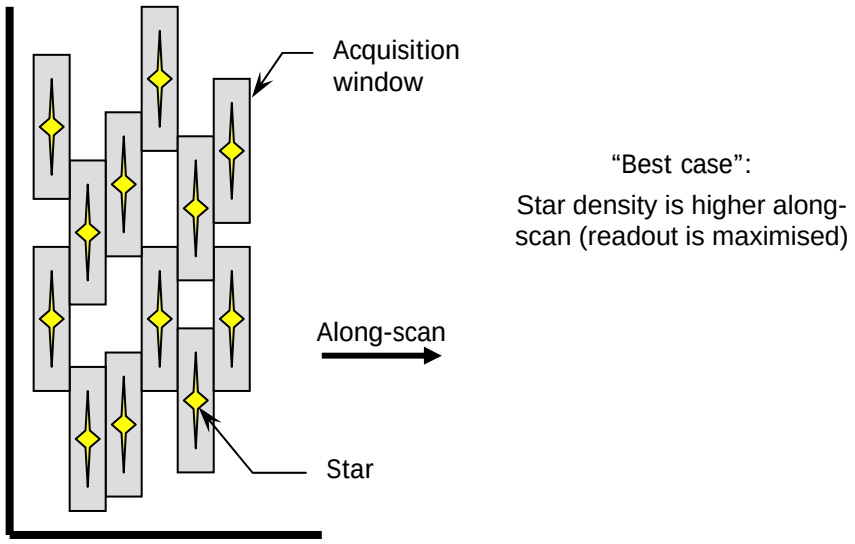


Figure 4.1: Star distribution on the focal plane (best case)

4.1.2.2. Memory sizing

The next question to address is the on-board storage. If a complete simulation has to be implemented (including possible limitations due to on-board storage), a first approximation can be obtained as follows: as we have seen, it is clear that we will need to store the data of, at least, the average measurements made between two contacts with the ground station (which is 16 hours). With 25000 stars per square degree on average, we will have about 300 stars entering the two astrometric FOVs every second. Assuming that during the contact time the new data generated will need no on-board storage (or that it will reuse the memory of the data already sent), we will have to store the data of about 17 million stars. Adding a 10% security margin we should have an on-board storage for at least 19 million stars. This is the minimum amount of scientific data storage needed on-board for both astrometric instruments. The real size of the storage needed can be easily obtained when the size of the information of a star (in bits) is known. Obviously, this is a very simplistic approximation, since it assumes a constant star density.

4.1.2.3. Telemetry sizing

Now we turn our attention to the next problem, which is how to transmit all this data to ground. On average, it is clear that we will have to transmit the data of about 300 stars every second, so if we take into account a daily contact time of 8 hours we will need a real link capability of 900 stars/s (considering only Astro data). The real data rate needed can be easily calculated when the size in bits for every star is known. It is important to note that this capacity is the one needed only for astrometric data: the Spectro contribution (as well as housekeeping data and attitude telemetry) have to be added to this value, as well as a contingency margin –specially for the large variations in the star density. Lammers (2004) offers a very interesting study on telemetry occupation.

4.1.3. Concepts and conventions

Some concepts and conventions on the operation of the Astro instrument should be well defined in order to allow uniform documentation for the whole scientific community involved in Gaia. In chapter 2 we have introduced some conventions at a reference systems level, but this should also be done at a payload (and data handling) level. For example, what do we call the *extension* of a source, which are the criteria for determining when a source is *point-like*, *extended* or *very extended*; the criteria to note a source as *fast moving*, or when a source is *binary* or *multiple*. Figure 4.2 illustrates an example of a criterion (based on maximum separation) that could be used to determine the multiplicity condition of a source or set of sources. Although these issues are beyond the scope of this work, we found interesting to raise this question. This was done during the first stages of our study, and other groups as the Gaia OBD have undertaken this work.

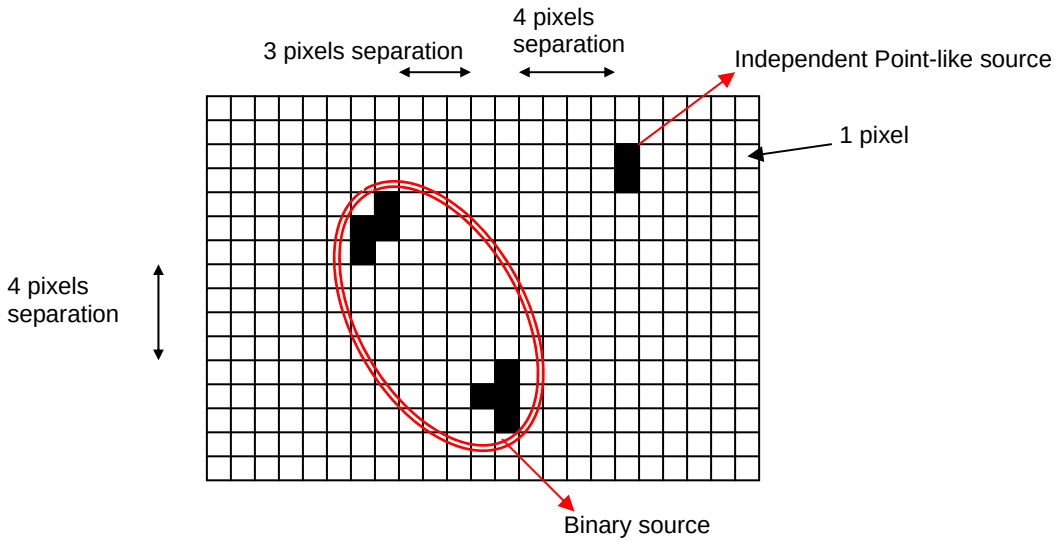


Figure 4.2: Determination of binary sources

4.2. DETECTION SYSTEM

Sources entering the focal plane will be detected by the ASM and then confirmed (or rejected) by the AF01. The requirements for the detection system are the following:

1. It must be able to detect the objects entering the astrometric Field Of View (FOV), with completeness up to a G magnitude of 20.
2. It must perform an approximate calculation of the magnitude of every object detected
3. It must measure the background around every object detected
4. It must also detect extended and multiple objects, and measure their shape and size (including multiple object information).

However, the main task of the ASM is the first one (detection of objects), because it will be the one responsible of the tracking of every object along the Astrometric Field (AF). This detection has to be done with the highest probability of success, from the brightest objects (about magnitude 3) to the faintest ones detectable by Gaia (which has to be complete at magnitude 20), and the probability of a false detection has to be as low as possible. Therefore, a good detection algorithm has to be selected and adapted for the ASM, specially taking into account the following additional requirements:

- Continuous operation is needed. Therefore, the algorithm should be designed to work with a continuous data flow and, thus, giving detection results working with very little blocks of data.

- Fast operation is also needed, in order to obtain results before the star exits the next CCD.

This last requirement is maybe the most restrictive one because of the limited scan time between CCDs: all of the processing on the data obtained from an object in the ASM should be done before the acquisition window corresponding to that object has to be read in the AF. This time includes not only the TDI time of a CCD, but also the transit time through the along-scan dead zone. On the other hand, some margins have to be included in order to take into account the size of the acquisition window, the block size used in the algorithm, and a possible along-scan motion of the object.

4.2.1. Instrumentation

4.2.1.1. ASM1/2

These strips of CCDs will have the most important task: they have to analyze the scanned sky area in a continuous way, looking for the objects that must be measured. The pixels will be binned in 2×2 samples in order to get a better read-out noise (RON). As already noted in ESA (2000), a large amount of cosmic ray events per second is expected, and the read-out rate will be high (despite of the binning), so the RON will be high as well. Therefore, a confirmation for every detection will be needed.

The ASM is the only CCD strip that will examine the whole scanned sky area, so the detection of extended objects (connectivity test) will have to be done here. Therefore, the outputs to be offered by the ASM will be the following:

- Position within the FOV reference frame. At this point it is not yet clear what will be the exact format of the along-scan position data (just the sub-pixel position, or also the CCD pixel column, etc.), but our recommendation is to offer the along-scan position as time data, and CCD row plus CCD coordinate (pixel + sub-pixel) for across-scan coordinate.
- Time of detection (absolute), with an adequate resolution.
- Magnitude estimation (flux).
- Background estimation, with at least the same resolution as object flux.
- SNR obtained in this detection (to be confirmed).
- Flags: point-like or extended object, size and shape (width, height and angle, only useful for extended objects).

The samples read around the detected object will also be output, usually as 5×5 samples of 2×2 pixels each.

4.2.1.2. AF01

This CCD strip is responsible of the confirmation of each detection made in the ASM. In fact, it will have to repeat (more or less) the job done by the ASM, but with a new threshold and simply confirming or rejecting the detection done there.

The operation of AF01 is based on the readout of 1×2 samples patched in 12×6 windows (as seen in figure 2.10 and Annex B). Despite of the fact that the samples are even smaller than in the ASM, the readout noise will be smaller since not all the focal plane shall be read. The patches will be centered on the positions of the detections made by the ASM. Because of this reduced CCD area for every object, the background calculation cannot be done in AF01 so it will need the background level detected at the ASM in order to execute the detection algorithm.

Also, extended objects will not be easily detected in AF01 (only slightly extended objects). However, an alternative algorithm for these objects could be implemented on AF01: when an extended object is detected in the ASM, the later could notify it to AF01 in order to analyze that zone (that patch) in another way, and to verify if there is a detection or not. This could be very useful for quasi-tangential cosmic rays: they could be wrongly detected as extended objects. Therefore, the inputs that AF01 will need from the ASM are the following:

- Background level, in order to execute the detection algorithm.
- Extended object flag and, maybe, also its size, in order to confirm that object with another algorithm if necessary.

On the other hand, the outputs are the following:

- Position of detection in AF01 (same components and resolution as ASM).
- Time of detection (relative: difference since the detection was made in ASM).
- Magnitude estimate (flux), with the same resolution as in ASM.
- SNR of detection (TBC).
- Flags: binary object (the AF01 will have less RON than the ASM, so it will better detect if that object is a binary system or not).
- Patch read: 12×6 samples (i.e., 72 flux values).

4.2.1.3. Outputs of the detection system

The general outputs to be offered by the ASM+AF01 system (from the processing by the *detection algorithm*) are the following:

- Position of the object wrt the focal plane reference frame (with about 1/5 pixel precision, TBC). The across-scan position will be the CCD row and the coordinate in that CCD, and the along-scan position will be just the sub-sample coordinate (because the pixel column will be known with the time field). This position will be the one measured in the ASM, as well as the one measured in AF01 (if re-detected). Position in AF01 could be codified in a differential way if desired (i.e., displacement from the expected position).
- Time of detection (absolute) measured in the ASM, and the one measured in AF01 (if re-detected). Time in AF01 could be relative to ASM detection time if desired.
- Flux estimates measured in the ASM and AF01. Flux in AF01 could be coded in a differential way if desired.
- Background measurement, made in the ASM.
- SNR obtained both in the ASM and in AF01.
- Flags: extended object, size and shape (ASM), binary (AF01).
- Patch read in the ASM and in AF01.

These outputs will be needed for the selection algorithm in order to select the objects to be observed and to track them along the AF and BBP. Some of them will be sent to ground, specially in order to reconstruct the acquisition window selection and thus to obtain the exact tracking that the AF and BBP do for every object. Also, some outputs (like mean object motion, obtained with positions and time) will be needed for attitude control.

4.2.2. Operation hints

4.2.2.1. Connectivity tests

A connectivity test has to be performed in the ASM in order to determine whether a pixel belongs to a source or not, and in order to determine the size and shape of the source. This connectivity test will likely work with some kind of threshold (which must be selected taking into account the typical PSF of a source), but also the *real* values of the flux should be taken into account in some situations. For example, we could have the situation illustrated in figure 4.3. In this situation, one of the pixels could be lost because it is not connected "physically" due to peculiar charge distributions over the pixels.

4.2.2.2. Detection margins

If a source is detected very close to a CCD across-scan border, the AF may not track these sources. Otherwise, partial patches would be obtained (e.g., 6×5 pixels rather than 6×12) in some of the AF CCDs, and the inclusion of this information within the computation could be difficult (special centroiding algorithms, etc). Therefore, a recommendation like the following one may be appropriate: sources detected and confirmed in the AF01 closer than 5 pixels (TBC) to the across-scan border of the CCD should be discarded. Figure 4.4 illustrates an example of this.

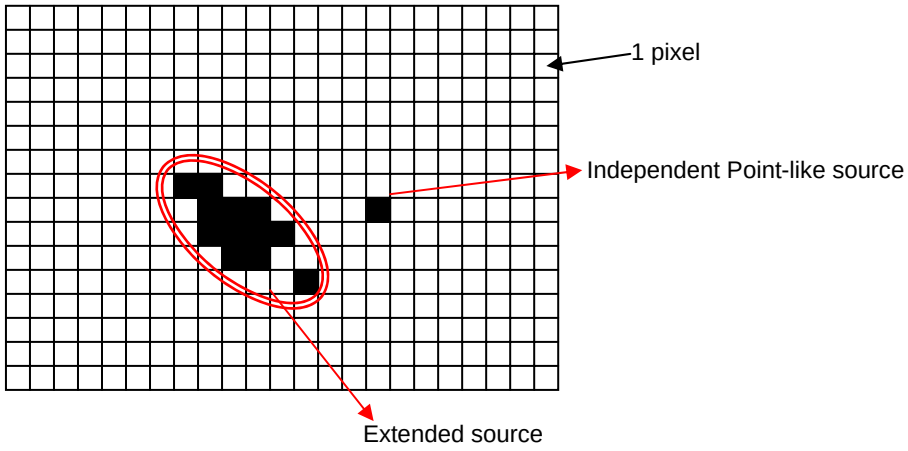


Figure 4.3: Example of a connectivity test

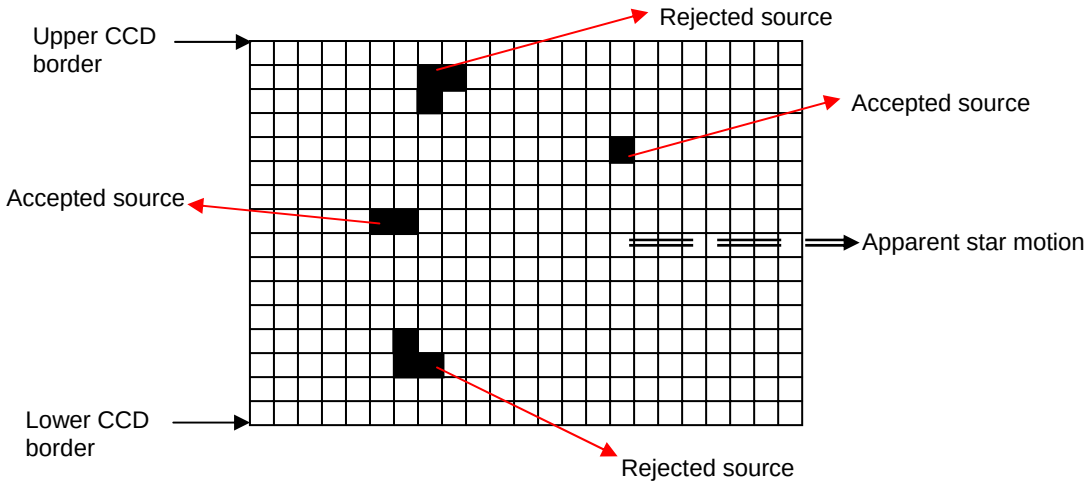


Figure 4.4: Detection margins in the AF01

4.3. SELECTION ALGORITHM

Once the detection task has been completed by the ASM, all of its outputs must be used in order to decide what do we observe and what not, as well as the way every object will be tracked along the AF and BBP. These decisions will be taken by the Selection Algorithm, which could be implemented within the ASM system although it will be treated separately here in order to define a more modular design. The main objectives of the selection algorithm are the following:

1. Patch control (positioning) for AF01. This task could be implemented in a separate module, named ASM sequencer –see figure 3.23 of ESA (2000), p. 173.
2. Discrimination of false detections (due to cosmic rays and noise), this is, confirmation of the ASM detection.
3. Motion measurement, specially important for fast moving objects (mainly solar system objects). For almost any other object the proper motion will be too small to be measured in the ASM.
4. Average displacements of objects along the ASM (for attitude control).
5. Patch control for the AF and the BBP:
   a) Centering
   b) Tracking
   c) Sizing

In order to accomplish these objectives, the selection algorithm may internally define an index that could reflect the current operating conditions. This will significally ease the decisions to be taken. This index, which we call the *field density index*, will be defined hereafter.

4.3.1. Patch control for AF01

4.3.1.1. Nominal patch centering due to spin motion

First of all, we should analyze at the nominal values for a static object: taking into account only the spin motion of the satellite (60 arcseconds per second), the time difference between the scanning of a point-like object in the ASM and its scanning in AF01 will be the following:

- For Astro-1 objects (ASM1→AF01): 79 mm, scanned in 5.82s
- For Astro-2 objects (ASM2→AF01): 49 mm, scanned in 3.61s

Therefore, when an object is detected in ASM1 (or ASM2), it should be read in AF01 exactly 5.82 seconds (or 3.61 seconds) after its transit time over the last pixel of the corresponding ASM CCD, and in the same CCD and pixel row. This is the nominal motion, but precession motion should also be taken into account.

4.3.1.2. Corrections on centering due to precession motion

These corrections to the nominal values will be done by the AOCS (attitude subsystem), which will indicate the exact deviation due to precession motion. Here we show a first estimate of the effect of the precession motion on the apparent motion of objects on the focal plane, although it may change due to modifications in the mission design. In the worst of the cases, the maximum correction along or across scan will correspond to a precession motion of about 0.2143 arcsec/s, applied *only* in the along-scan or in the across-scan direction. Table 4.1 shows these worst-case values.

	Astro-1 objects (ASM1→AF01)	**Astro-2 objects (ASM2→AF01)**
Maximum along-scan deviation	± 20.79 ms	± 12.89 ms
Maximum across-scan deviation	± 9.41 pixels	±5.83 pixels

Table 4.1: Corrections to the nominal projections on the focal plane due to precession

The time shown as maximum along-scan deviation corresponds to the maximum correction needed when reading out the data (i.e., the time at which AF01 data must be read). These are the values for a point-like object, but now we should also take into account the size of the acquisition window in AF01 (which is 12×12 pixels).

4.3.1.3. Patch centering constraints

The patch of 12×6 samples in AF01 leads to a scan area of 12×12 pixels, which should be centered on the position (and time) obtained before (taking into account the spin motion and the precession motion). To achieve this, i.e., to re-observe in AF01 an object detected in the ASM, the former must start reading out the CCD window 6 preceding pixels before this position obtained, and then 6 following pixels. This means that readout should start exactly 4.42 ms before the time obtained before (with spin and precession motions), and this readout should last 8.84 ms in total (12 pixels). During this time, 12 pixels (6 samples) across-scan have to be readout, centered on the position obtained.

On the other hand, we must remember the pixel binning used in AF01 (1×2 pixels). This means that a centering at the pixel level will usually not be possible to obtain, because binning can be considered as a uniform partition of the CCD surface. An example is shown in figure 4.5, where the "ideal" centering obtained cannot be achieved due to pixel binning.

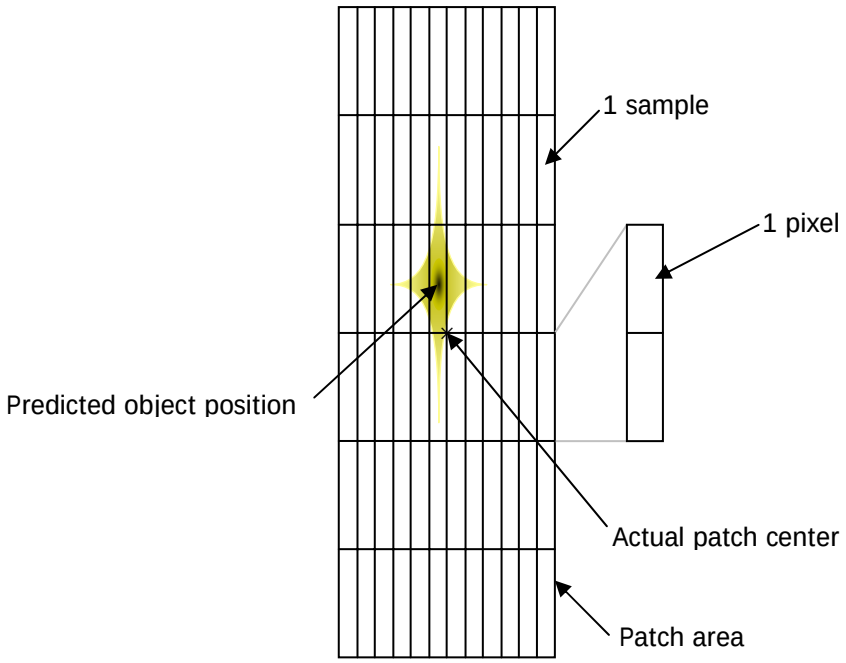


Figure 4.5: AF01 patch miscentering due to discrete focal plane readout

The worst case, shown in this figure, leads to a displacement of 1 pixel across-scan. The worst along-scan error will be the minimum one, i.e., half a pixel, since there will be no along-scan binning in the whole AF. These centering errors obviously lead to reduced margins in the

centering operation. The way to control this (i.e., to control the ideal centering combined with "discrete" data availability) is very simple and almost automatic: the Selection Algorithm (or the ASM sequencer, if a more modular design is desired) has to request an AF01 readout at the ideal time (minus 4.42 ms, in order to take into account the patch area). Then, AF01 will start reading out when possible (i.e., when the next pixel has to be read), so miscenterings will always appear as a "delay" (this is, the predicted position of the object will be always at the left of the actual center of the patch). Across-scan centering is more flexible, and decision leans on AF01 sequencing criteria.

4.3.2. Discrimination of false detections

Once an object has been detected in the ASM and re-scanned in AF01, false cross-matchings between these two detection systems must be rejected. To do this, both position and magnitude margins will have to be defined in order to determine whether a detection in the ASM and AF01 is the same or not.

4.3.2.1. Position margins

If the patch centering in AF01 has been correctly done, a static object will be exactly on its predicted position on this CCD. However, fast-moving objects could show a slight displacement, corresponding to their motion during the crossing time of 1 narrow CCD (or 1 narrow plus 1 wide CCDs). This is, the displacement from ASM1/2 to AF01. Fast-moving objects will usually be faint, which may lead to further complications –as we will see hereafter. Solar system bodies, and specially Near-Earth Objects (NEOs), are considered the fastest-moving objects detectable by Gaia, so the position margins in the AF01 patch will be calculated from their motion. The maximum speed of a NEO is assumed to be 300 arcsec/hr (Høg 2000) or 83.3 mas/s, so in the 5.82s (±20.79ms due to precession) between ASM1 and AF01 (nominal motion) the maximum displacement of a NEO will be 485 ±1.73 mas (along- or across-scan). This leads to a maximum displacement in AF01 of 11 pixels along-scan, or 3.7 pixels across-scan. In the case of a source observed by Astro-2 (and hence detected by ASM2), the 3.61s (±12.89ms) between ASM2 and AF01 will lead to a maximum displacement of 300.83 ±1.07 mas (along- or across-scan), which is equivalent to 6.8 pixels along-scan or 2.3 pixels across-scan. As it can be seen in figure 4.6, an extremely fast moving NEO can fall out the standard acquisition window in AF01. These extreme cases should be further studied in detail in order to use smaller margins (and, hence, to obtain more reliable cross-matchings).

Therefore, if an object detected in the ASM is re-detected in AF01 with its centroid within these margins, the position cross-matching can be accepted (i.e., the detection is accepted). Furthermore, we recommend applying margins that could take into account small centroiding errors due to position resolution (i.e., due to GD resolution). Finally, an important thing to take into account is the possible error in the attitude control: the AOCS should give this data to the Selection Algorithm, and the later should recalculate AF01 patch centering (and margins) according to this error. At the same time, the ASM and the Selection Algorithm will calculate average object displacements on the ASM and will offer these results to the AOCS, so there will be a feedback between these two systems.

4.3.2.2. Magnitude margins

An object detected in the ASM and verified in AF01 will have a very high probability of being a "real" object (i.e., a valid detection). However, not only position constraints have to be taken into account, but also magnitude constraints: a detection in the ASM and another detection in AF01 in the predicted position could not correspond to a same object (even to a "real" object) if magnitudes detected in the two CCDs are very different: this difference could for instance appear in the case of cosmic rays hits. Therefore, some margins (TBD) should also be applied here.

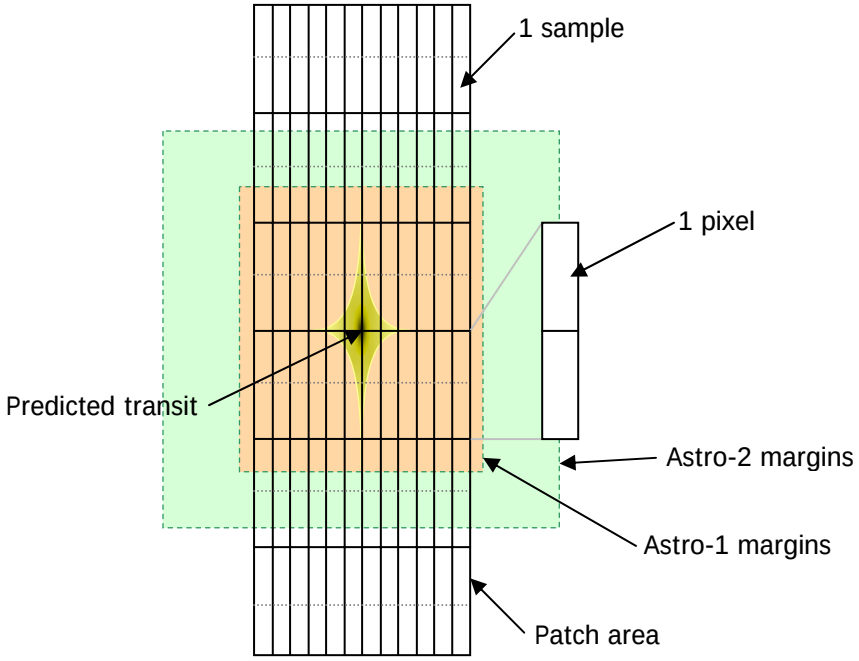


Figure 4.6: Position margins in AF01

4.3.2.3. Other verifications

The detection and selection algorithms should be as accurate as possible, offering the highest percentage of successful detections (and a minimum percentage of incorrect or missed detections). An additional verification would be to take into account the size of the object: a point-like object detected in the ASM must also be point-like in AF01, and an extended object should also be so (and with similar dimensions) in both of them. Also, in order to avoid wrong cross-matchings or interpretations with very faint objects (close to magnitude 20), the pixels around a detection in the ASM could be temporally stored and re-verified if some unusual detection is done in AF01. Figure 4.7 shows an example.

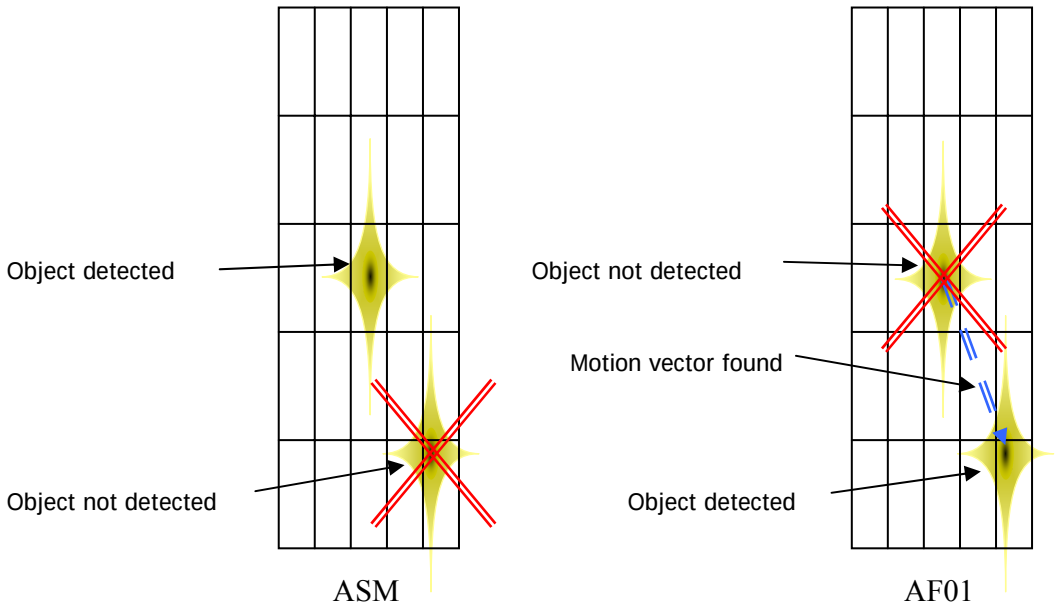


Figure 4.7: Wrong cross-matching with faint and close objects

In this figure both objects have a magnitude close to 20 (completeness limit of Gaia). In the ASM, one of the objects is detected and the other one not, while in AF01 the other object is

detected and the first one not. This leads to a false "fast moving object" detection, so the selection algorithm could try to follow that "fast" object with an appropriate AF tracking operation while, actually, both objects would not be observed at all. This error can be corrected by temporally storing the pixels around a detection in the ASM (only when the detection is close to the limit capability of the algorithm, i.e., with very faint objects), and verifying them if a fast motion is detected in AF01. Therefore, if pixels corresponding to the original position in the ASM are very close to the detection threshold, and so do the pixels corresponding to the new detected position in AF01, then the selection algorithm should determine that actually there is not one fast-moving object, but two steady objects. The counterpart of this kind of verification is the on-board resources required to implement this, which may be prohibitive.

4.3.3. Source motion measurement

Once detection and cross-matching are completed, the selection algorithm has to take both detected positions in the ASM and in AF01 in order to calculate the displacement wrt nominal motion (due to spin and precession motions). This displacement, divided by the time between both detections, will offer the motion of the object. It is important to note that attitude deviations have to be subtracted to the previous value in order to not include possible errors in attitude control. The calculation of these attitude deviations is explained in the following section.

4.3.4. Attitude control

The AOCS is the subsystem in charge of controlling and monitoring the attitude of the satellite. This subsystem receives data not only from the detection and selection algorithms, but also from other subsystems like the Wide Field Star Tracker (WFST). The monitoring task of AOCS is fairly simple: it has to accumulate all the displacements of the sources along the ASM and obtain afterwards the average motion of the sources over the ASM. Extreme cases (i.e., fast moving objects) may be discarded in this computation. This will provide a good estimate of the attitude of the satellite.

Some improvements could be applied in order to obtain a better estimation of the attitude. For example, this mean motion could be calculated in a continuous way and objects with a high deviation could be ignored (in this calculation). Also, only brighter objects could be used for this (since they will have a higher probability of being static than fainter ones). Another solution could be a combination of both: fainter objects (e.g., mag>16) could be discarded from the calculation, as well as those objects with a high deviation in their motion from the average one.

4.3.5. Patch control for the AF and BBP

Centering of point-like objects is a trivial process, since it basically reproduces the same process as in AF01 patch centering. For extended or multiple objects, not only brightness criteria should be used, but also position criteria: the acquisition window should be centered on the "brightness center" (a kind of "center of masses" taking into account the flux of the pixels), rather than on the brightest part. In this way, the patch will be able to contain the entire object (or all the objects, if multiple) or, at least, the most relevant part in the centroiding calculation. Also, some "random" criteria could be included here, in order to do not lose always the same sources (e.g., in a crowded field with faint objects near bright ones). Details of this patch centering process for extended or multiple objects are still being developed.

The tracking of sources along the AF and the BBP will be, more or less, the same operation used to control AF01: taking the nominal motion (spin and precession), attitude corrections (from the AOCS) have to be applied (at the CCD level), and finally proper motion corrections have to be applied on every object (i.e., their deviation from the average displacement of the objects along the ASM). An easy way to do this is, actually, to extend the displacement

detected along the ASM for every object (i.e., to extend the motion vector obtained in the ASM), which will include the real attitude as well as the object motion. Individual CCD corrections should be treated in the AOCS description, although TDI corrections will be avoided in order to obtain a simpler operation.

4.3.6. Density selection (field density index)

In the previous sections we have proposed some operational guidelines for the selection algorithm. As it can be seen, its operation is fairly complex and many details should be taken into account in order to correctly select the most interesting sources. In order to accomplish these requirements, we introduce here a new concept which will not only help in this objective, but will also ease the implementation and analysis.

We propose to use an internal index, which we name *Field Density Index* (FDI), in order to describe the kind of field being observed by each instrument of Gaia at a given time (crowded field, low density, medium density, etc.). This index (or indexes, one per field of view) would express the source density of each field currently being scanned. This would be indicated as the number of sources detected (and confirmed) every second –and hence it would be updated every second. Another possibility would be to code these indexes as relative to the average source density predicted for the mission, or to the mean "processing capability" of Gaia. If this would be the case, an FDI lower or equal to 1 would indicate that Gaia can perfectly cope with the current measurements, while an index larger than 1 would reveal processing capability problems due to high densities.

Another proposal is that an *Integrated FDI* (IFDI) could be implemented as the average of the last FDI values (e.g., the last 60 values), but also computed every second. This would help in determining the trend of the field being observed: at a given moment, if FDI is higher than 1 (or higher than the average density) but IFDI is lower than this, it would mean that the density is increasing. The IFDI is specially useful when taking into account the large density variations that the instruments of Gaia will have to deal with. Hence, a low IFDI will reveal that there are still enough resources available (e.g., on-board storage or telemetry) for the current field – despite of an eventually high FDI.

This index-based operation is much more useful than a simple flag-based operation (e.g., enough resources available or not): the higher the indexes, the most restrictive must be the selection –and vice versa. Furthermore, two IFDI indexes could be calculated, using short and long integration periods. For example, a *short term IFDI* could use an integration time of 1 minute in order to indicate hardware limitations, while a *long term IFDI* (with an integration time of 1 great circle, this is, 6 hours), would indicate on-board storage and telemetry shortages.

4.3.6.1. Selection by hardware limitations

Our idea of using field density indexes can be extended and improved in order to also take into account the *scientific interest* of a source. We must remind that the on-board hardware resources are also limited, so a first selection process must be done even at a hardware level. This is, determining which sources detected in the ASM should be rejected beforehand, in order to save AF01 resources. This selection must also be repeated afterwards, in order to determine which must be measured in the full AF and BBP.

Such a kind of index could be a *Source Priority Flag*, describing both the scientific interest and the detection quality of a source. For example, this index could be higher for brighter point-like sources with a high detection quality. Discrimination criteria should be selected, in order to define what should be observed when there are more detected sources than available resources. Random-like criteria could be included in the evaluation process in order to obtain a more uniform completeness.

4.3.6.2. Selection by storage limitations

Some considerations should also be done regarding the on-board storage system. First of all, the stage where data compression is performed must be determined –i.e., compressing the data *before* or *after* being stored on-board. We recommend compressing the science data *before* being stored, thus requiring about a third of the storage capacity.

Data selection before transmission should also be done, and hence, some mechanism for selecting one or another set of data (already compressed) should be designed. We propose a bank-based implementation, this is, a kind of multi-priority on-board storage. Then, data should be selectively stored beforehand (depending on, e.g., the current IFDI), and the banks should be transmitted afterwards depending on the downlink availability. We envisage at least 3 memory banks:

- *Basic* data bank. Its size should be enough to fill 8 hours of transmission to the ground station, and hence all the sources scanned in an *average* field should be stored here. Taking into account different field densities (FDI and IFDI indexes), all of the stars observed in low and medium density fields shall be stored here.
- *Extended* data bank. Its size is TBD, and would store, for example, the stars observed in fields with a density up to twice the Gaia capability. In this way, we would assure that the data link is always full (i.e., we assure that we always transmit data), even if a very low density field is scanned during a long period. Moreover, the data link would be filled with relevant data, since it is not expected that Gaia will be able to scan very dense fields –so any information on these fields will be very interesting.
- *Emergencies* or *contingences* bank. If an error in a daily contact of Gaia occurs, all of the data measured during one day could not be transmitted until the next contact. Then, all of these data could be moved to this bank and the standard bank would then be free for new data. It would obviously be difficult to transmit all of these data during the next contact, so the second bank (extended data) should not be filled meanwhile.

An *overwrite algorithm* should also be considered, this is, which data should we overwritten (and therefore lost) when receiving new data and the old data has not been transmitted yet. We propose here to make use of the Source Priority flag in order to overwrite the least important sources. If this is done source by source (or in small blocks of sources), then we should take care of reordering the data for a better (and easier) transmission. Finally, we must also take care of clearly identifying every source or set of sources: as we will see in forthcoming chapters, this reordering process may lead to some problems if not done correctly.

4.3.6.3. Selection by TM limitations

The last step will be to transmit all of the data stored on-board when a contact with the ground station is available. This process must obviously start with the highest priority bank (the *basic* data bank). The rest of data, if possible, may be transmitted depending on their independent (or group) *source priority* flags.

As a last suggestion, the low priority data may be transmitted in a different way, e.g., using lower resolutions (8-bit fluxes rather than 16-bit), or even discarding part of the measurements (i.e., transmitting half of the AF measurements). These low-priority data may be false detections (i.e., ASM data of AF01-rejected detections), or detections of exceeding data (data that cannot be scanned or stored or transmitted, due to hardware or memory or telemetry limitations). An additional flag may be included in these cases (false detection, hardware-rejected, memory-rejected, telemetry-rejected, etc.). These data would be stored in the "extra" memory banks, and transmitted only if possible. For example, Gaia could scan a small but very dense area of the sky that could saturate its electronics (i.e., too much simultaneous windows may be required, or even overlapped windows would be generated); but if the average status (i.e., the IFDI) of the on-board storage and telemetry is acceptable, then the transmission of partial ASM data could be interesting.

4.4. REFERENCE FIELDS FOR SIMULATIONS

It is clear that the instruments of Gaia will have to operate under very different and extreme conditions. In order to test as much as possible these situations, a set of *reference fields* should be defined. We propose here the following "standard" fields which may help in testing the simulation systems under the most typical situations:

- Low density field: simple point-like stars with a low density (e.g., 5.000 stars/deg^2).
- Standard field: simple point-like stars with an average density (about 25.000 stars/deg^2).
- High density field: simple point-like stars with a high density (e.g., 100.000 stars/deg^2).
- Crowded field: simple point-like stars with a very high density (e.g. 1.000.000 stars/deg^2).
- Binary field: standard field with a high percentage of binary stars (e.g., 30%). Also *low density binary field* and *high density binary field* could be considered.
- Galaxy field: standard field with a high percentage of galaxies, e.g., 30%. Also low and high densities could be considered.
- NEOs field: standard field with a high percentage (e.g., 30%) of fast moving objects with different proper motions (up to 300arcsec/hr).
- Bright background field: standard field with a brighter background, e.g., $M_V \cong 21$ or 22.
- Cosmic rays field: field with about 30% of false detections in the ASM due to cosmic rays (distributed randomly in space and time).
- Variable stars field: standard field with about 30% of rapidly variable stars, with different amplitudes and waveforms.

4.5. CHAPTER CONCLUSIONS

In this chapter we have described the envisaged operation of the astrometric instrument, specially focusing on its source detection and selection systems. We have also proposed some parts of their operation, specially for the selection algorithm, introducing new concepts (as the field density indexes) that will help in understanding and implementing the system. The high complexities of these systems require not only a text description of their operation, but also illustrative schemes with a first approximation to their implementation. This will be covered in the next chapter.

Part II:
Payload Data Handling System

Chapter 5

An optimized and pipelined PDHS for Gaia

Gaia will be a technological challenge in all its aspects, and the on-board data handling is a clear example of this. In this chapter we describe our proposal for the Payload Data Handling System (PDHS) of Gaia, designed as a high-performance, real-time, concurrent and pipelined data system. This proposal includes the front-end systems for the instrumentation, the data acquisition and management modules, the star data processing modules, and the Payload Data Handling Unit (PDHU), including some hints on a supervisor module. We also review other payload and service module elements and we illustrate an overall data flux proposal. All these modules and sub-modules, as well as their input and output signals, are intended to be consistent with the designs and simulators currently available for Gaia, such as GASS and GIBIS.

The work described in this chapter was started on the request of the On-Board Detection working group (OBD) of Gaia, which asked for some ideas for the data flux within the payload of Gaia and a possible supervisor module in charge of controlling the whole PDHS of the spacecraft. Also, the OBD was interested in presenting the results of this work as an input for an industrial contract for the Payload Data Handling Electronics (PDHE). Beyond this, the project team has used it in order to further define the details of the scientific requirements for the payload. It is very important to note that this work aims to offer a high-level description of the payload modules and data fluxes. Although some items may be described in detail, even using terms of "bits", electronic components, or other low-level terms, it is done just in order to avoid confusions; at the end, the systems will be implemented as the industrial teams decide. As in the rest of chapters, we focus on the Astro instrument, as well as on the data handling unit. The Spectro instrument will also be treated here but with not many details.

The chapter is organized as follows: section 1 is an introduction to the problem of the data handling in the payload of Gaia. In section 2 an overview of this payload is given. In section 3 the astrometric instrument is described, while the PDHU is described in section 4. Section 5 lists the other elements in the service module of Gaia, and finally section 6 summarizes our conclusions.

5.1. INTRODUCTION

Measuring more than 1 billion objects several times with the highest resolution implies a technological challenge, not only for the predicted 75 TB data base of compressed raw data (on ground), but also –and specially– for the on board data handling. A full reading of the whole set of CCDs would imply a video output of more than 7 Gbps (taking only the astrometric focal plane), so a selective sampling method is mandatory. As seen in the previous chapters, this will be done by detecting and selecting the relevant sources to measure, reading only windows around these selected sources. The sampling method is fully described in Høg et al. (2003) and summarized in Annex B. For a typical case of single faint stars, the astrometric instrument bins 12 pixels in the across-scan direction for obtaining a sample, reading afterwards six contiguous samples in the along-scan direction. In this way, it mostly covers the Point Spread Function (PSF) of the image.

Using this sampling method and the baseline technical features, the video data output rate of the astrometric focal plane may be reduced to less than 400 Mbps in the worst case, namely, when a very dense stellar field is observed. Typical stellar fields should imply an output of about 200 Mbps. Although this value may still appear large, typical communication standards like Spacewire can implement this payload data bus bandwidth comfortably.

As noted in chapters 2 and 3, Gaia will orbit around the L2 lagrangian point, 1.5 million kilometers from the Earth opposite to the Sun. Therefore, the satellite will be seen from the Cebreros ground station for about 8 hours per day. This, presumably, will be the only radio station used for receiving the data, so a permanent link between Gaia and the ground station will not be available. This implies that the 4 Mbps data link will be reduced to an effective downlink of about 1.3 Mbps. Therefore, all the data acquired by the astronomical instruments must be compressed (with lossless methods), packetized and stored on a mass storage system, waiting for being transmitted to ground during the next contact.

In this chapter we elaborate on the Payload Data Handling System (PDHS) of Gaia, from the observation instruments to the mass storage. We intend to propose a complete system design, not limiting ourselves to a list of scientific requirements for the PDHS, in order to offer a much richer input to the industrial teams that will be in charge of manufacturing the system. The main elements proposed here are the front-end modules of the astrometric instrument (Astro), the Payload Data Handling Unit (PDHU), and the data exchanges between all these modules. Some suggestions about a Supervisor module (SPV) coordinating the PDHU and the instruments are also given. All these modules and sub-modules, as well as their input and output signals, are intended to be consistent with the simulators and designs currently available for Gaia. Overall, the main interest of this proposal is the need of being consistent with the intrinsic operation of Gaia. It implies the use of a high-performance data handling system, operating in real time and capable of being implemented as a pipeline, since in average about 300 sources per second will enter the astrometric focal plane and will be measured during about 1 minute. Furthermore, the actual number of stars entering the focal plane will vary in a large scale, from very few stars per second up to some thousands stars per second, depending on the area of the sky surveyed.

The latest design of Gaia is again taken into account (EADS/Astrium 2002), as well as the different sampling options depending on the brightness or kind of sources (Høg et al. 2003). However, an option of selectable gate phases will be kept, in order to allow for a reduction of the integration time for a given source. We also describe here some third-party elements and designs such as the CCDs and parts of the service module for the sake of completeness.

5.2. OVERVIEW OF THE PAYLOAD OF GAIA

5.2.1. Summary of modules and sub-modules

In figure 2.10 we can see 10 CCD trails, each following the apparent motion of the stars on the focal plane (i.e., in the along-scan direction). A CCD trail is a physical part of the focal plane assembly (FPA), containing both the CCDs and the proximity electronics needed for basic star acquisition. We define an *Astro Trail Unit* (ATU) as a functional unit, including a set of video processing modules, which converts a CCD trail in a self-operated module. Each ATU is intended to operate independently, which leads to an intrinsic parallelism in the design and implementation of the proximity electronics and data handling. Other options for the design included a distribution of this focal plane in 10 Video Processing Units (VPUs), each linked to a trail of 18 CCDs and their proximity electronics. In this work we propose a different distribution based on ATUs, which is more intuitive, efficient and easy to implement. Each one of these 10 ATUs contains the following elements:

1. One CCD trail, composed of 18 CCD chips, 18 video chains and 18 local sequencers. As described in the previous chapter, the first two CCDs (the ASM) will not only detect the sources entering the focal plane but will also classify them –whether they are being observed by Astro-1 or Astro-2 telescope. The next eleven CCDs (the AF) will perform the high-quality astrometric measurement, while the last five CCDs will perform Broad Band Photometry (BBP).

2. One Window Acquisition Manager (WAM), which will make possible the measurement of *only* the relevant areas of the sky.
3. Two Star Data Processing Modules (SDPM), implementing the Detection Algorithm (DA) and the Selection Algorithm (SA), see section 5.3.2 below. The DA will detect and confirm stellar sources entering the focal plane, also discarding false detections due to cosmic rays. The SA will select the sources to be measured and the quality of this measurement. Also, the SA will compute the real spin rate of the satellite using the measured star data, and will assign a *priority flag* to each source depending on its scientific interest.
4. A Star Data Collection Module (SDCM), compiling the data from all the CCD chips and the star data processing modules.

The Spectro instrument is composed of a single telescope. The center of its field of view will be projected over a spectrometer designed to measure radial velocities through Doppler shifts. Thereof its name, *Radial Velocity Spectrometer* or RVS. A *Medium Band Photometer* (MBP) will measure the outer parts of the field of the telescope. The technical implementation of Spectro in CCDs, video chains, sequencers and other video data processing modules will be similar to the Astro implementation. It will also contain several functional units operating in parallel, named *RVS Trail Units* (RTUs) and *MBP Trail Units* (MTUs). Figure 5.1 illustrates only 2 MTUs and 1 single RTU, which is the design envisaged for those instruments.

The Payload Data Handling Unit (PDHU), linked to both the instruments and the High Rate Telemetry Formatter (HRTF), will be composed of the following elements:

1. Supervisor (SPV), which will control the overall operation of the instruments and data handling.
2. Science Data Selection (SDS), which will be in charge of selecting the data to be transmitted to ground taking into account storage and downlink limitations. Also an eventual lossy codification will be determined by this element, although this would be applied only to extra data. All the basic data generated by Gaia, related to astrometry up to a G magnitude of 20, will be transmitted with no losses. Finally, the transmission priority will also be assigned by the SDS.
3. Data Handling and Compression (DH&C), compressing and packeting the data according to CCSDS standards.
4. On-Board Storage (OBS), also known as mass storage, which may be implemented as a solid-state recorder.

The attitude subsystem will contain an Attitude Data Collection Module (ADCM), compiling data from the Attitude and Orbit Control System (AOCS), the Wide Field Star Tracker (WFST), and several gyroscopes. It is important to note here that, because the CCDs data allow a high precision, the scan rate will be measured from the sky mappers and plays a major role in maintaining the required attitude accuracy. The housekeeping (HK) subsystem is composed of a Housekeeping Data Collection Module (HKDCM) and several sensors placed both in the payload module (PLM) and in the service module (SVM). The clock subsystem is composed of a master clock, a Clock Distribution Module (CDM), and several local counters distributed among the instruments and modules.

5.2.2. Global scheme and data flux

Figure 5.1 shows a global view of the payload of Gaia, including the main data fluxes between them. Some of the signals have been intentionally omitted or simplified for the sake of clarity, whereas some links to the SVM have been included for illustrative purposes. This figure clearly shows how we can consider the Astro instrument as 10 independent sub-instruments, which we name ATUs. This independency is considered a requirement, and taking advantage of this parallelism when implementing the system in DSPs or microprocessors is strongly

recommended. It is envisaged that the MBP and RVS instruments will operate with a similar parallelism, although these instruments are still under design.

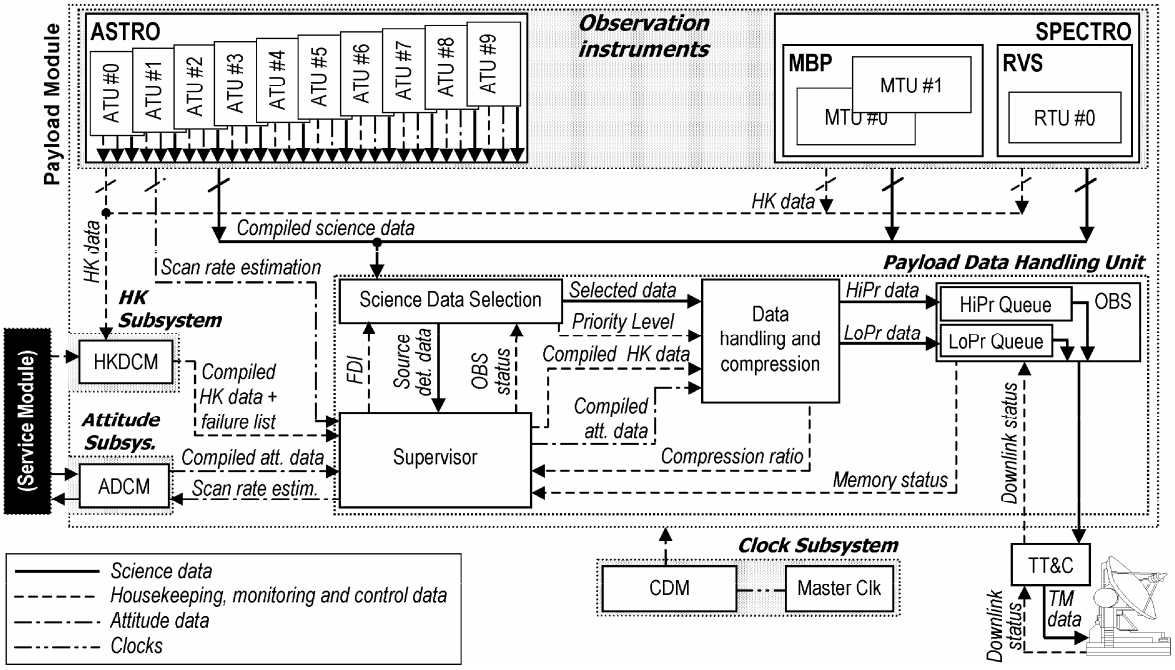


Figure 5.1: Global view of the Gaia payload, including the main data interfaces and internal modules of the PDHU.

5.3. ASTROMETRIC INSTRUMENT (ASTRO)

In this section the main interfaces, sub-modules and operation overview of the Astro instrument of Gaia are presented. We focus on the astrometric focal plane, which is the main operative element of the Astro instrument. We recall that a field identifier (ASM, AF or BBP) is used for referencing its several parts, followed by a CCD strip number: 1…2 for the ASM, 01…11 for the AF and 1…5 for the BBP. The MBP and RVS instruments, still under design, will be based on a similar scheme. All these instruments can be considered as self-controlled systems, measuring the sources and sending the data to the appropriate module or sub-module as they enter the focal plane. A few parameters are controlled by the SPV, as seen in the interface list below. Some inputs to the Astro instrument may not feed directly into the focal plane or into its proximity electronics, but may be filtered by the star data processing modules. A diagram of this system is given in figure 5.2 where some interfaces, as well as a part of the CCD trail, have been omitted for the sake of clarity. The Astro instrument is composed of 10 of these units, operating independently and in parallel. The following are the main inputs to Astro:

1. *Attitude data* from the ADCM, filtered by the SPV, needed for the positioning of the acquisition windows.
2. *Operate signals* from the SPV indicate if a given module or sub-module is being operated or deactivated. It may be needed due to hardware problems, or during the transfer phase to the final orbit around L2.
3. *Clock signals* from the CDM must offer the adequate time resolution with the highest possible precision.

There will also be other inputs (which shall always be fixed to a nominal configuration) for opening the way to eventual in-orbit reprogramming. For instance, the SPV offers the *Maximum simultaneous samples* value to every CCD. This parameter indicates the maximum

number of samples that can be readout at a given time from a CCD chip. This value affects the resulting readout noise (RON), because the higher this value, the faster the CCD will have to read the data. Actually, this value indicates the number of samples that are *always* (at any TDI period) read from the CCD. It is required for maintaining constant the power dissipation. Useless samples will be discarded.

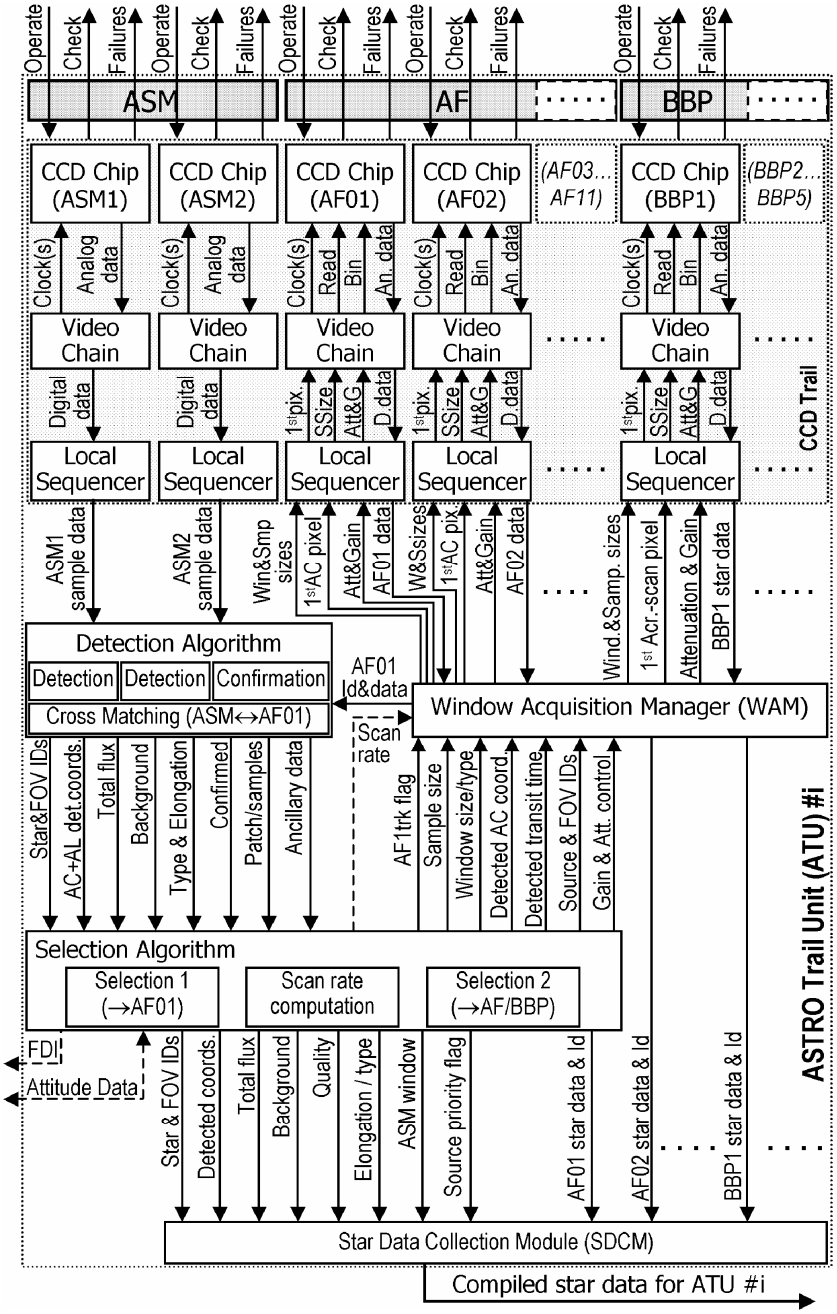


Figure 5.2: Modules and data flux within an Astro Trail Unit (ATU).

The outputs of Astro may also be from other sub-modules, rather than from the focal plane itself:

1. *Compiled source data*, towards the Science Data Selection module, is the result of the observations made in this instrument.
2. *Housekeeping data*, towards the HKDCM, contains a set of global-checks and a failure list.
3. *Scan rate estimate,* towards the ADCM, offers an estimate of the scan rate. This output is filtered by the SPV before reaching the ADCM.

4. *Field density index* or FDI, towards the SPV, is an estimate of the star density of the field currently being observed. It will be useful for data selection and other issues (see chapter 4).

These interfaces are applicable to each one of the ATUs. As previously mentioned, each one of these units contains a CCD trail, with 18 CCD chips aligned in the along-scan direction. Each CCD has its corresponding video chain and local sequencer for commanding the image acquisition at a higher level. Figure 5.2 illustrates this distribution, together with the internal and external CCD trail interfaces. The remaining video processing and management modules of an ATU are also illustrated there.

5.3.1. Focal plane and proximity electronics

We define a CCD trail as a set of 18 CCD chips in the along-scan direction, with their corresponding video chains and local sequencers. The full set of 10 CCD trails conform the focal plane and proximity electronics. This concept can be easily identified as a physical element –the CCD focal plane with the electronics attached to it. On the other hand, an ATU is identified as an operative and standalone element. Here we describe the contents of a CCD trail, while the rest of elements within an ATU –the video processing and management modules– are described in section 5.3.2 below.

5.3.1.1. CCD chips

The commanding method and interfaces of a CCD chip are highly industrial dependent, so the features described here will be rather general and based on the requirements listed in ESA (2000). Their final implementation will be done under industrial contracts with ESA, but they must always operate in TDI mode and follow the main guidelines illustrated in figure 5.3 (adapted from ESA 2000). Although several hundreds of stars will be projected on a CCD during its real operation, only two stars are shown in this figure for the sake of simplicity. The transfer of charges in the CCDs must be perfectly synchronized with the apparent motion of the star images on the focal plane. In this way, the image of a star will enter a CCD trail and will be tracked all along it, from CCD to CCD. Several sources will be tracked simultaneously, both across-scan and along-scan. In the current design, up to some 30 sources may be tracked simultaneously, across-scan, in AF02-10. The along-scan quantity of sources will depend on the stellar field, which ranges from a maximum of $3 \cdot 10^6$ stars/deg^2 to a typical value of $2.5 \cdot 10^4$ stars/deg^2. These stellar field densities imply a typical rate of about 175 stars/s per telescope, up to a maximum of 32000 stars/s.

As seen in figure 2.10, the astrometric instrument contains narrow and wide CCDs, with 2600 or 4500 pixel rows, respectively. All of them have 1966 pixel columns. According to chapter 2, we name *row* a pixel line in the across-scan direction, and *column* a pixel line in the along-scan direction. The active surface of the CCD can be considered as an array of parallel shift registers using the same TDI clock, as illustrated in figure 5.3. This figure shows also the readout register, which is a row of not-illuminated pixels with a higher capacity. It can be considered as a serial shift register, outputting the pixels or samples one by one. The clock frequency used for this operation will depend on the kind of pixels being read. In this way, useless pixels will be flushed at high speeds, while useful pixels or samples will be readout more slowly. This selective readout method lowers significantly the resulting read-out noise of the measurements, typically below $10e^-$. Therefore the useful dynamic range is improved, since the maximum capacity of a pixel well is about $200.000e^-$. The video chain linked to each CCD is the only element commanding it, except for the *operate* signal which comes from the supervisor module:

1. *Operate* is an "on/off" binary switch for deactivating a CCD if it fails.
2. *Read* is a serial input indicating whether the pixel being shifted out from the readout register has to be read or flushed.

3. *Bin* is another serial input indicating whether the pixels being read must be binned (summed) or not. This input only has sense if read is active. A notch in this signal will cause a reset in the accumulator, thus making possible the readout of two consecutive samples. Another possibility could be to include a reset_sum_register input.
4. *Gates*, a multiple input, controls each one of the 12 selectable gate phases of the CCD. It activates them or not depending on when a pixel row has to be reset or not. Thus, bright stars may be measured this way, using a shorter integration time in order to avoid pixel saturation.
5. *Clocks* include a parallel clock, at about 1.36 kHz, distributed to all the pixel rows in order to perform the parallel charge shift. Also a serial clock, up to a few MHz, is used in the readout register for serial pixel readout or flush. This readout clock will be different for flushed and useful pixels. Its valid values must fulfill a set of conditions for making possible the readout and flush of the entire readout register during a single TDI period.

The main output of a CCD chip is the analog data, which sends the accumulated flux of each pixel towards the corresponding Video Chain. Also, the HKDCM is reported with the global check and list of failures of the CCD chip.

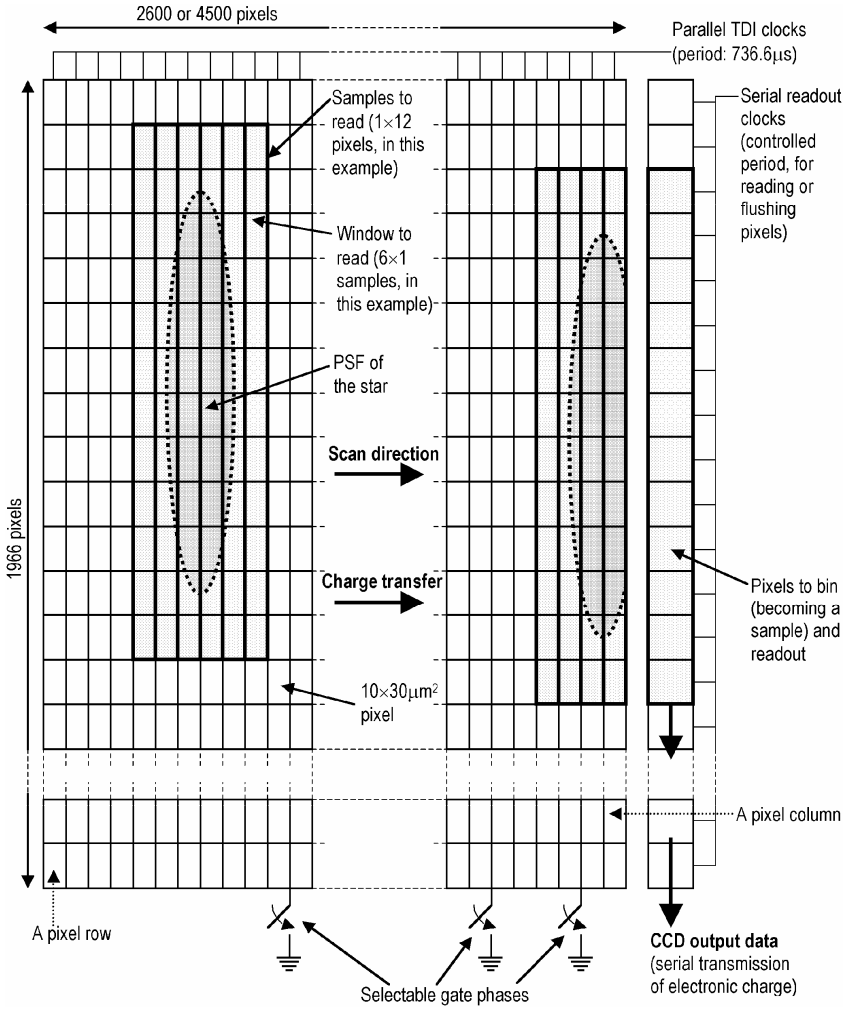


Figure 5.3: Operation principle of the CCDs to be used in the astrometric focal plane

5.3.1.2. Video Chains

A CCD chip is the most elementary module that measures the data, treating its pixels one by one. A video chain (or video controller) must be included for dealing with more useable data. It pre-processes the data, converting it from electronic charge to electronic voltages, and

afterwards converting it to digital 16-bit values. Also, this module converts mid-level commands to low-level commands for the CCD. Therefore, every CCD chip must have its own video chain (VC), so an ATU will also contain 18 video chains.

A video chain manages commands related to a single pixel row. For this reason a strict instruction protocol must be defined, which will be based on the TDI period (736.6μs in the Astro instrument). During this time, all of the pixels in the readout register of the CCD must be read or flushed. We define the transit time as the time when a given set of pixels of a star is loaded in the readout register. In other words, a star image will be over the readout register at its transit time –although the readout register is actually not illuminated. With this in mind we define the commanding protocol for measuring a set of samples from a given pixel row. Assuming that their transit time is at TDI cycle #n:

1. At the TDI cycle #n-1 the VC receives a sequence of commands indicating the samples to be read.
2. During the TDI cycle #n the commands are executed. This will imply an output from the CCD chip. These analog data will be amplified, digitized and sent to the local sequencer.
3. At the end of the TDI cycle #n plus the A/D conversion time, all the digital data will be available in the local sequencer.

This commanding protocol is further explained in section 5.3.2.1 below. Figure 5.4 shows a possible implementation of a video chain, also including the gate phases and variable gain control. The FIFO queue stores the commands until their correct release time. Its size should be about 2 times the number of simultaneous samples that can be readout in a CCD. In this way, the commands for the current and the next pixel row can be stored.

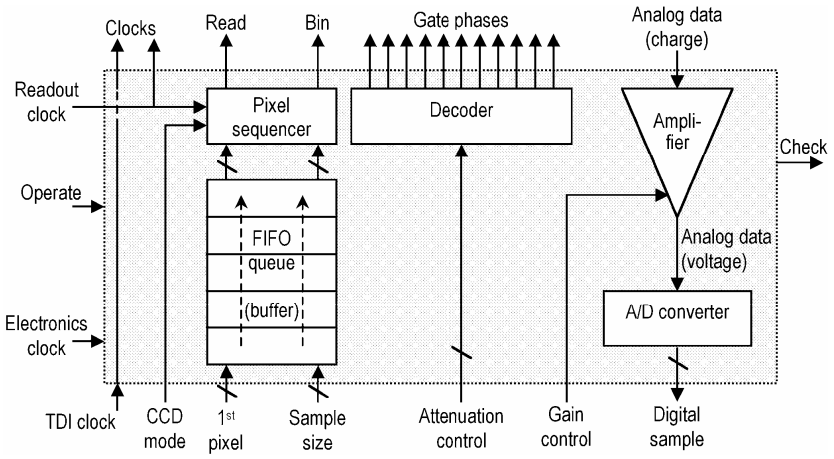


Figure 5.4: Internal layout of a video chain.

This figure also shows the required inputs and outputs of a VC. Interfaces in the top of the figure link with the CCD chip, and they have been already described in section 5.3.1.1. *Operate* and *check* have the same meaning as in the CCD chip. An easier readout control for the SPV, as well as a higher flexibility in case of failure, is provided by other inputs. *CCD mode* is an example of this, which will mostly be fixed during the entire mission because of the pre-fixed behavior of the several parts of the focal plane. It could be changed if necessary, thus forcing a *windowed mode* of a *continuous mode*. A CCD operating in windowed mode only reads pixels around each star. It is the typical mode for the AF and BBP. On the contrary, the continuous mode implies reading the whole CCD, thus ignoring some of the VC inputs. It is the typical mode for the ASM.

The other inputs, all of them sent by the local sequencer, are required for commanding the windowed readout mode. *1st pixel* indicates the first pixel (across-scan) to be read for a given sample, following the convention of pixel numbering as described in chapter 2. *Sample size* indicates the number of across-scan pixels to be read and binned for the current sample.

Attenuation control indicates the gate phases to activate for reducing the integration time, while *gain* control selects a low ($3\mu V/e^-$) or high ($6\mu V/e^-$) amplification for the VC amplifier. The data readout from the CCD chip and converted to a digital value is finally sent in 16-bit format, offering a high enough resolution. A logarithmic-like A/D converter offering higher resolutions at lower signal levels could be an interesting alternative, and may be studied in the future.

5.3.1.3. Local Sequencers

Video chains operate only with data measured during one TDI cycle, acquiring them from the readout register of the CCD. On the other hand, local sequencers are able to manage two-dimensional data, acquired during several TDI cycles. Therefore, they deal with *patches* and *windows* (i.e., sets of samples both along and across scan), while video chains deal only with samples. The use and basic requirements of both video chains and local sequencers were already introduced in ESA (2000), but we propose here a detailed operation scheme for these modules.

A CCD local sequencer receives readout commands related to a given star with a fixed anticipation with respect to its transit time. In this way, unnecessary signaling and electronics near the focal plane are avoided, such as the transmission of source identification. Two TDI cycles of anticipation are recommended, thus leaving 1 cycle for sequencing tasks and another one for video chain tasks. Combining several *source read* commands into *sample read* command sequences is the main task of a sequencer, which will release these commands towards the video chain at their adequate time. We will come back to this issue when discussing figure 5.7 in section 5.3.2.1.

A local sequencer is always linked to a video chain and its corresponding CCD chip. It can be considered as the main external interface of the focal plane. The *Window Acquisition Manager* (WAM) controls the local sequencers in order to obtain the measurements during the passage of a given star along the focal plane. As explained before, the WAM will send commands to the sequencers 2 TDI cycles before the transit time of each star. These commands include, as shown in figure 5.5, the window and sample sizes to be used for a given star. As a requirement, all the samples within a window must have the same size. Also, only commands for contiguous windows will be accepted. This implies that some measurements, mainly those of bright stars, will have to be requested as two different measurements, both for the same star.

The *1st AC pixel* is another required input for acquiring star data. It indicates the first across-scan pixel of the window to be acquired, which corresponds to the uppermost pixel to be read as seen in figure 2.10. It follows the two conventions of pixel numbering described in chapter 2. *Max simult. samples*, introduced in section 5.3, is an example of those inputs required for an eventual in-orbit reprogramming. It tells the sequencer the number of samples that must be read in the CCD chip during a TDI cycle. If the requested samples are not enough for reading this amount of samples, the sequencer will complete this number requesting *dummy* samples. It is important for maintaining a constant power consumption in the focal plane. Fluctuations around the nominal value may lead to thermal variations, which must be avoided due to the mission requirements. About 32 samples must be continuously read in most of the CCDs in the current design.

Star data is the main output of a local sequencer. The local sequencer sends to the WAM the set of samples acquired for a given stellar object. Also, a *Read OK* flag indicates if this source has been successfully read. If not, only a *Partial Observation* (PO) will be available for this star. A reason for this may be a sample overlapping, or a window falling out of the CCD bounds. It must be noted that samples from different stars cannot overlap on the CCDs, although windows can do it if necessary (Høg et al. 2003).

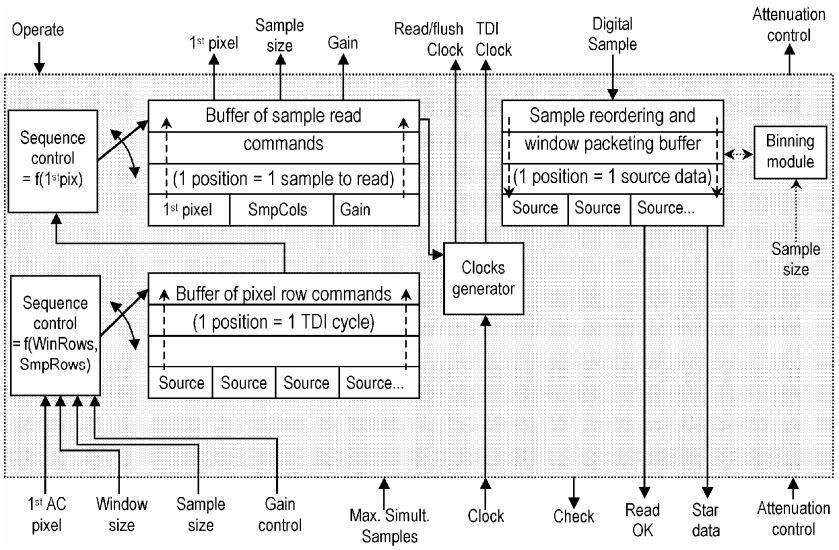


Figure 5.5: Possible implementation of a CCD local sequencer.

Figure 5.5 illustrates these and other interfaces. We recommend implementing the local sequencers on a DSP or similar. Interfaces in the top of the layout have been already described in section 5.3.1.2 before, as well as *check* and *attenuation control*. A single *clock* input will be required at the master clock frequency, namely 6.4 MHz. Other clocks for TDI and readout operation will be generated by using, e.g., frequency dividers.

The acquisition requests for a local sequencer will be partitioned in sample commands and fed into a queue of pixel row commands. These will be sorted depending on several issues, such as the window width or the sample width. Sample commands for several sources will be combined in this buffer. All the commands related to a star must fit in this queue, so its size must be larger or equal to the widest of the windows that can be acquired. According to Høg (2002a) the number of registers must be at least 68. One of these registers will be served every TDI cycle, feeding them into another buffer. This second queue will contain as many registers as simultaneous samples to be acquired in the CCD. Each register will contain sample requests commands, ordered using the *1st AC pixel* value. These commands will be served at the beginning of a new TDI cycle, being sent to the video chain at their corresponding time (see figure 5.7 in section 5.3.2.1 below).

Digital samples received from the video chain will be recombined for each source. This recombination is possible because a fixed commanding protocol will be used, and because sample data will be received from the video chain during the TDI cycle next to its request. Also, useless samples –for maintaining a constant readout rate– will be discarded during the recombination process. Finally, remaining binning operations will be performed, and sample sets from each star will be combined in order to offer star data blocks. Ultimately, all these data will be sent to the WAM at the beginning of the next TDI cycle.

5.3.2. Video processing and management modules

The focal plane and proximity electronics are composed of 10 CCD trails, which will be able to acquire star data as requested. Our proposal for the modules in charge of selecting the stars to be acquired, command the CCD trail and process the results are described in the next subsections.

5.3.2.1. Window Acquisition and Management (WAM)

The aim of this module is to accurately control the several sequencers of the CCD trail for acquiring the star data, and to forward these data towards the PDHU. It is commanded mainly by the Selection Algorithm (SA), which sends the basic coordinates for each star to be tracked.

From these, the WAM obtains the sequence of transit times and transmits the individual acquisition coordinates to each local sequencer. We note here that the CCDs of AF01 measure the data in a different way than the rest of the CCDs of the AF, being its main task the confirmation of the source. Because of this, the SA inputs a special *AF1trk* flag into the WAM. With a value of *true*, it indicates that the detection (ASM) coordinates are input, and that AF01 confirmation coordinates must be computed. Otherwise, it indicates that confirmation (AF01) coordinates are input, and that AF02-11 and BBP measurement coordinates must be computed. This is illustrated in figure 5.6, where *AF1trk = false* propagates the acquisition coordinates all along the CCD trail. The controls for AF03 to AF11 and BBP2 to BBP4 have been omitted in this figure for the sake of clarity.

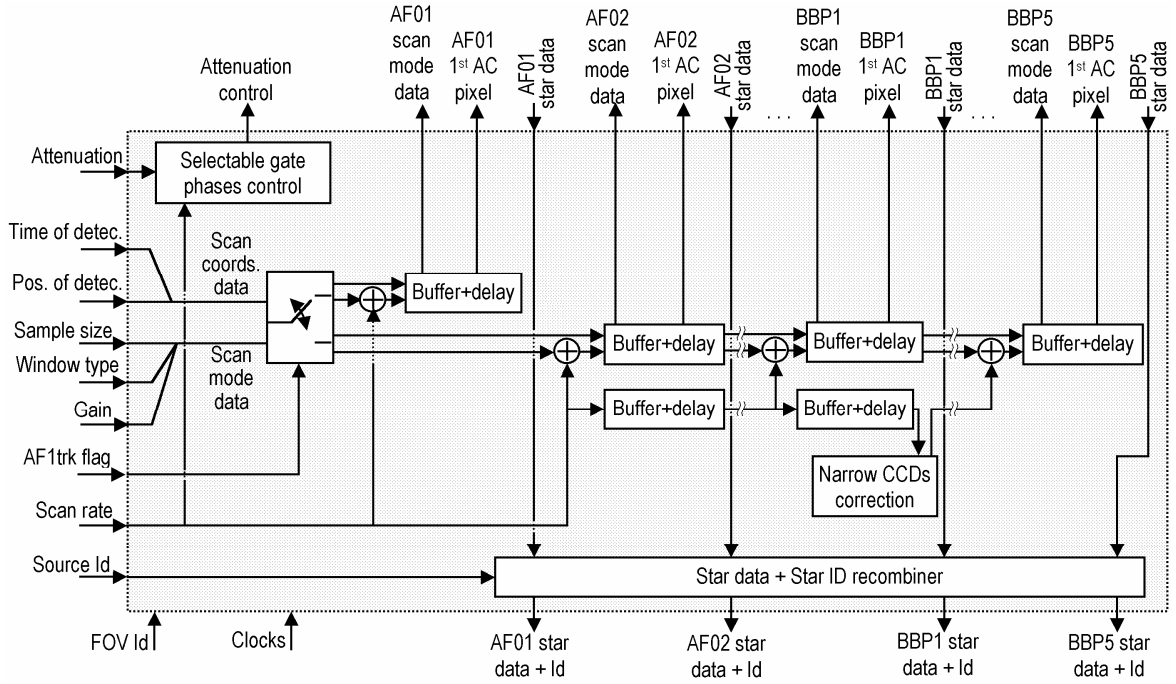


Figure 5.6: Internal layout of a Window Acquisition Manager.

Figure 5.6 also illustrates the several interfaces of the WAM. One of these is the *scan rate*, which is an input from the attitude subsystem indicating the apparent motion of the sources on the focal plane, in arcseconds per second. Additional data will be included to this scalar value in order to report the "slope" of this apparent motion over the focal plane, thus composing a kind of *scan rate vector*. Also the coordinates (time and across-scan position) where the source has been detected or confirmed are provided. The WAM will take these *origin* coordinates and add the *scan rate vector* to them several times. Each time this addition is performed, the acquisition coordinates for a new CCD strip will be obtained. These transit predictions will always be calculated taking into account the attitude at the detection or the confirmation time, not at the time when the star is measured in the AF or in the BBP. We note the *narrow CCDs correction* element in the layout, which corrects the calculation of the transit times for being adequate to CCDs BBP2-5, which are narrower than the rest of CCDs.

The sampling method, which includes the sample size and window type, is also provided by the SA, as well as the gain and attenuation values. These parameters will be different for AF1-10, AF11 and BBP fields, so multiple inputs are required. Also a *Source Id* input is necessary for uniquely identifying the source being measured and, thus, making possible the compilation of its data at the end of the measurements. This identifier can be as simple as a counter, with a recycling time large enough. We note that the WAM outputs each CCD measurement one by one, while the SDCM will compile all of them (see section 5.3.2.4 below). Neither this identifier nor the transit times will be transmitted to the local sequencers, thus avoiding unnecessary interfaces. The acquisition protocol shown in figure 5.7 makes it feasible. The results of the star measurement are sent to the WAM exactly *win_width* + 3 TDI cycles after

requesting the measurement, taking *win_width* as the width of the acquisition window in pixels. Data from source measurements with the same *win_width* and requested during the same cycle will be processed on a *first requested – first output* basis. Sources projected by one or another telescope (Astro-1 or Astro-2) are detected by ASM1 or ASM2, respectively. It implies different calculations for the transit times. Therefore, a *field id* flag must also be served. For the example shown in figure 5.7 we have taken a typical sampling case of 6 samples with 12 pixels each. The readout process will operate as a pipeline, so the 3 extra TDI cycles are not strictly "dead times".

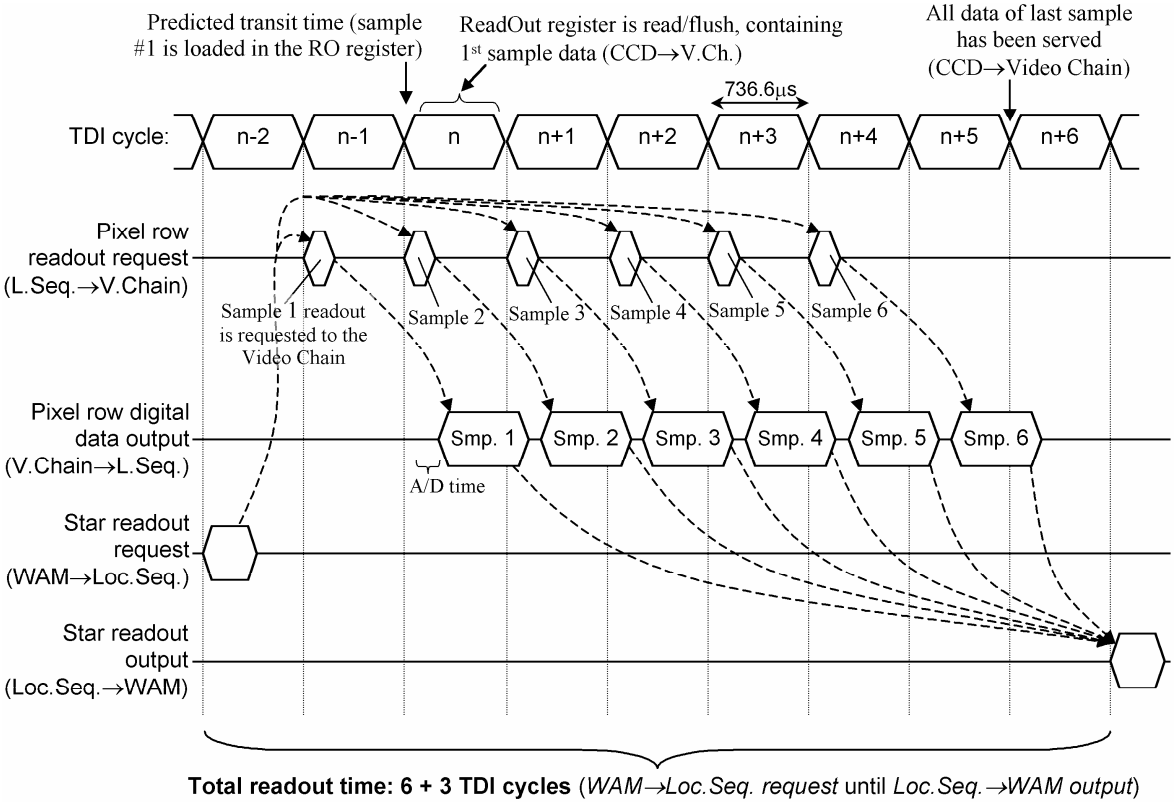


Figure 5.7: Timing protocol for the acquisition of star data.

5.3.2.2. Detection Algorithm (DA)

The way in which Gaia measures the stars has already been described previously. However, we still have to show how Gaia "*knows*" what (and where) to measure. The Detection Algorithm (DA) is the key for this.

The CCDs of the ASM are the first ones where the images of the stars are projected. These CCDs are fully read and the result is a continuous image strip, a kind of panoramic picture of the sky. The DA receives this strip and looks for sources in it. Point-like stars are the most typical sources, but there can also be binary stars, galaxies, Near-Earth Objects (NEOs) or other solar system bodies, etc., all of them with a wide range of brightness, up to magnitude 21. Every source will also have different background conditions, motions and shapes. Overall it makes the design of an image detection algorithm a very complex task (Chéreau 2002).

The DA should be implemented in a set of microprocessors or DSPs, taking advantage of the intrinsic parallelism of Gaia. It is the video processing module which has the most extreme requirements, both in timing restrictions and processing load. Therefore, it is imperative to implement the DA as a concurrent and pipelined set of parallel processes, whichever physical implementation is used. Its processing capability must also be correctly sized, so that very dense stellar fields can also be analyzed.

From the operational point of view, the DA is composed of four sub-processes. Two of them are identical, although operating with different inputs –ASM1 and ASM2. They receive the

continuous image strips and look for sources in these images. Not only the coordinates and features of the detected sources will be obtained, but also a background estimate will be computed and a detection quality flag will be issued. A third sub-process is in charge of confirming the detections, using AF01 measurements. Although these images are not in a continuous strip form but as windows around the detections, the operation of this sub-process is also similar to the previous ones. The main difference is that the background estimate is taken from the preliminary detections, and that a better image quality is used –because only a fraction of the CCD is read. Finally, a fourth sub-process is in charge of *cross-matching* the detections and the confirmations. It will not only verify that detection occurs both in ASM (first detection) and in AF01 (confirmation), but it will also verify the main features of the source. In this way, false detections due to cosmic rays or readout noise will be minimized.

The main outputs of this module will include the coordinates of the detection with an adequate resolution. A pixel-based resolution is recommended: about 1/250 of pixel will be enough for avoiding important quantization errors. Also the validity and quality of the detection will be an output, as well as the source features. It includes the elongation, the type of acquisition windows to use and the total flux measured, as well as a unique identifier. The field (ASM1/2) where it has been detected must also be output, thus indicating if the source is projected by the Astro-1 or the Astro-2 telescope. Finally a set of samples surrounding the ASM detection, as well as the AF01 window, will be sent for completing the measurement data block.

5.3.2.3. Selection Algorithm (SA)

Once the several sources entering the focal plane have been detected and confirmed, Gaia must select the most prioritary of them. The current status and capability of the PDHS, as well as the detection features, must be taken into account. The most prioritary sources will always be measured, while the remaining sources will be measured depending on the current system load and on the stellar field density. Selecting the sources to measure –and the way it has to be done– is the aim of the Selection Algorithm. It will depend on the available resources. Data selection, in order to decide what to transmit –even what to store on-board– will be done in a second stage. This will be performed by the Science Data Selection module in the PDHU.

Acquisition modes for every confirmed source will be selected by the SA. It includes the gain, attenuation, and sampling mode. Also a small shift of the acquisition window may be selected, which will be notified to the WAM outputting *bogus* detection coordinates. A reason for doing this could be the presence of a nearby source.

One of the main features of the SA is the capability of calculating the *source priority flag* (SPF). Each source will be classified in this way, numerically indicating its scientific interest. Key factors for its calculation include the multiplicity or shape of the source, its brightness and the quality of its detection. This flag will be useful for both selection stages in Gaia, so that measurement, storage and transmission of a source will be an easier decision. The SPF is calculated during the first SA stage, when processing ASM detection data. Taking AF01 hardware limitations, first-detected sources will be selected for trying to confirm them in AF01. It is named *Selection 1* in Høg et al. (2003) and in figure 5.2. Confirmed sources will correct their SPF value, because of the higher quality of AF01 measurements.

The SA will continuously compute the *Field Density Index* (FDI), indicating the density of the stellar field currently being scanned. This value, as well as the SPF, will be used for on-board control purposes. Therefore none of them will be included in the science telemetry –at most in the housekeeping telemetry. The long-term index, the *Integrated FDI* (IFDI), will also be calculated by integrating the FDI evolution during a few seconds. Both values, FDI and IFDI, will be computed taking into account all the confirmed sources, even those not selected for being measured.

ASM detection and AF01 confirmation are separated in time by 3.6 or 5.8 seconds, respectively, depending on whether the sources are projected onto the ASM1 or ASM2. This value and the resolution of the measurements are high enough for obtaining a good estimate of the scan rate. This is a third task to be done by the SA, which will make possible an accurate

tracking of the sources on the focal plane. Moreover, the attitude determination will be obtained with a higher resolution than in the AOCS. The next stage of the SA is to determine the samples and acquisition windows to be used when measuring the source in the rest of the CCDs (AF02-11 and BBP). This stage includes the overlapping test, checking if a nearby source would overlap its window with the window of the current source. If this is the case, the algorithm will try to accommodate the windows by shifting their centers. If this is not possible, one of the sources will be discarded. Another possibility is that the windows overlap only in some CCDs. It can be possible with one source from every telescope, because the apparent across-scan motion will be opposite. In this case, a *Partial Observation* (PO) may be selected (Høg et al. 2003). The last stage, selecting the most interesting sources to be measured all along the focal plane (AF02 until BBP5), is named *Selection 2* in Høg et al. (2003) and in figure 5.2. All these measurements will be requested to the WAM.

5.3.2.4. Science Data Collection Module (SDCM)

This module simply collects all the detection and measurement data of a source, sending it afterwards to the PDHU. These data are inputs from the SA and the WAM, including not only the data to be transmitted to ground but also possible extra flags. An example is the *source priority flag*, to be used when assigning the transmission priority to the data. A flag indicating the ATU (0 - 9) is also included by the SDCM itself. The cross-matching of all the data related to the same source is possible thanks to the *source id*, embedded in the SA and WAM data.

5.3.3. Astro operational diagram

The diagram in figure 5.8 shows the process of detection, confirmation, selection and measurement of a star entering the astrometric focal plane. Some names of the operations are taken from Høg et al. (2003). In this figure the three main tasks in the measurement of a source are displayed as three columns: detection, selection, and high-quality measurement. Solid lines indicate the flux diagram, while dashed lines give a hint on the data path. It includes the samples from the ASM detection, the AF01 confirmation and measurement, and AF02-11 and BBP measurements.

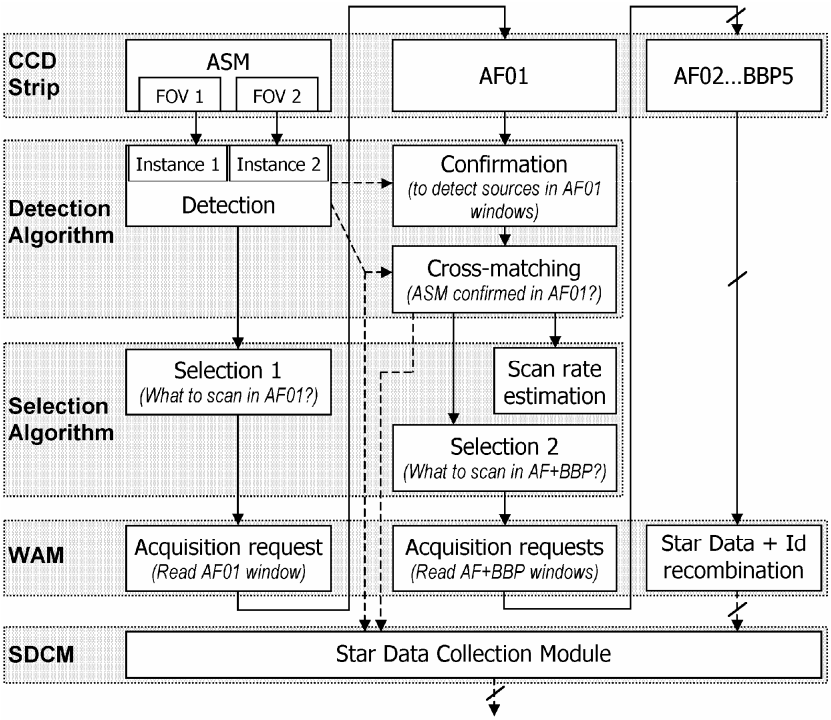


Figure 5.8: Operational diagram of the Astro instrument.

5.4. PAYLOAD DATA HANDLING UNIT (PDHU)

Perhaps, one of the most critical issues of Gaia is transmitting the huge amount of data generated by the instruments and, hence, it is one of the main challenges in its design. Measuring only a set of small images around each source already decreases this amount of data by a large factor. However, the pre-selected data may still not fit in the downlink. The PDHU, as we propose here, must decide which of these data must be retained for other contacts, and which must be directly discarded.

The PDHU should be implemented in some kind of programmable device, preferably a general-purpose microprocessor because of the varied tasks to perform. After the Detection Algorithm it will be the module with the highest processing requirements. Several microprocessors will probably be used operating in parallel, taking into account the limited processing power of space-qualified devices. The details of the tasks to perform may change after Gaia has been launched or even during the lifetime of the mission. For example, an optimized data compression scheme may be developed when real measurements will be eventually available. Therefore, the devices implementing the PDHU should be re-programmable in orbit. This is the reason why we recommend that some data fluxes go through this module, e.g., the scan rate or the housekeeping data. Overall it will provide a higher flexibility, even in case of failure. Several processes will operate concurrently for implementing the PDHU. We describe each of them below.

5.4.1. Science Data Selection

The three selection criteria operating in Gaia are described in this section. Source selection depends on the hardware resources, and it is performed by the SA in the instruments. The other two are applied by the *Science Data Selection* (SDS). The first one takes into account the amount of memory available on-board, and decides whether to store or discard the source data. The second one takes into account the downlink capability, in order to delete the data or transmit it with a given priority.

An input from the *On-Board Storage* (OBS) informs about the memory available in each of the multi-priority queues. With this information, together with the field density indexes, a resource availability index is computed. This index gives a hint on how much restrictive should the selection be. An estimate of the data compression ratio will also be taken into account, provided by the *Data Handling and Compression* module (DH&C). Science data is afterwards indexed in a stack depending on the source priority flags. This stack is then emptied by blocks, from higher to lower priority, sending the science data towards the DH&C. Lowest priority data may even be discarded if compression resources are not enough. In the compression module different methods will be applied to the data depending on their priority, but specially on the resource availability. If memory or telemetry resources are very low, lossy compression methods may be used for the low priority data. These methods may be as simple as discarding some of the high-resolution measurements, or even transmitting only detection data. The losses, however, must be directly controlled by the data system. It means that compression systems such as wavelets or JPEG will not be applicable at all.

5.4.2. Supervisor (SPV)

This is the process in charge of controlling most of the PDHS modules and sub-processes, as well as the data flux between them. Failures are also controlled here. If one CCD chip, video chain or local sequencer reports a failure, the supervisor will turn it off. A substitute for this will be selected in the case that it is required. For example, an ASM failure would be fatal for the operation of the focal plane. Then, an AF01 would operate as ASM, and AF02 as AF01 (Høg et al. 2003).

Attitude data will be filtered by the SPV before offering a final scan rate estimate. It will select the most *well-behaved* sources such as single bright stars. In this way, sources with high proper motions will not be included in the estimate, thus approaching closely the actual attitude of the satellite. This selection may also be applied to the FDI index if necessary. Integrated FDI (IFDI) is also calculated here, which offers a long-term indication of the stellar density being measured. This integrated index will be a key factor in the SDS module: combined with the resource availability index it will indicate if the selection restrictions can be relaxed or not. The control of the evolution of these indexes makes it possible. For example, the resources may be low at a given moment, and the density indexes be high. If the tendency is decreasing, the SDS can safely assume that the generation of science data will decrease, and so it can assign more resources to it.

5.4.3. Data Handling and Compression (DH&C)

It is another crucial process in the data path of Gaia, since there is still too much data to downlink, even taking into account the sampling method and the selection of sources and data. Some studies (Lammers 2004) reveal that about 5 megabits per second of raw data will be generated. This implies that a data compression ratio of at least 3 must be obtained. Only in this way we can fit these data in the 1.3 megabits per second available in the downlink. Furthermore, a lossless compression system is imperative, because no data from a source measurement can be lost. At most, its resolution or measurement multiplicity could be reduced. The data to be transmitted are not raw images or data with a clear and uniform structure, so standard compression systems do not satisfy the requirements. We did some essays on data compression for the original Gaia design (Portell et al. 2001b, 2002b) which offer interesting results –although they were simple systems. The idea of partitioning the science data in uniform blocks, and to use differential coding for repeated measurement data, seems to be the right approach. It implies that the DH&C must have a *data partitioner* offering sub-frames with uniform data. Afterwards, a *data analyzer* will look for redundancy patterns in each block of data, selecting the most appropriate data compressor for it. This process may be suppressed and the data redirected with a fixed scheme, depending on the final compression system. A set of independent data compressors, such as Huffman, LZW or Rice, will properly compress the set of data sub-frames. All of them must operate in parallel. The real compression ratio will be measured afterwards, and reported to the SPV and SDS. Finally, the telemetry frames will be generated. They will include the compressed science data, as well as the housekeeping and attitude data. CCSDS standards are met in the packet and frame generation. We will elaborate on these issues in the forthcoming chapters of this work.

5.4.4. On-Board Storage (OBS)

Selected and processed data must be temporarily stored on-board until a downlink contact is established. A solid-state mass memory recorder will likely be used for this, implementing some kind of file system for differencing the priority levels. One file will be created for every priority level, although figure 5.1 represents only 2 levels as data queues. The priority is labeled on each block of data, thanks to the SDS. When entering the OBS it will store a data block in one or another queue (or file). The *High Rate Telemetry Formatter* (HRTF) directly links with the OBS. When downlink availability is reported data queues are emptied, from the highest to the lowest priority. Low priority data that cannot be transmitted during that contact will be stored if possible, waiting for the next contact –and slightly increasing its priority level, within a limited range. We note here that the OBS cannot reorder or manage its data. It only inputs, stores and outputs data sequentially, although there are several concurrent processes –for the different priority levels. We can compare the OBS with a set of FIFO queues.

5.5. OTHER MODULES

Most of the modules listed in this section may be standard industrial components, ready to be included in the Gaia payload or service module. Only an overview of their required operation and interfaces is included here for the sake of completeness.

5.5.1. Attitude Data Collection Module (ADCM)

This module receives attitude data from both the Astro instrument and the AOCS. The latter offers gyroscopes and *Wide Field Star Tracker* (WFST) measurements, while the former gives a higher resolution provided by the CCDs. The task of the ADCM is to compile all of these data and redirect them towards the appropriate module. It can be considered as the interface between the payload and service module for attitude data.

5.5.2. Housekeeping Data Collection Module (HKDCM)

All the housekeeping data reported by the payload systems converge into this module, as well as those from several sensors placed in the service module. The HKDCM compiles and monitors them in order to report errors to the SPV. One of the most important sensors is the basic angle monitor (BAM), which is a laser device offering an accurate measurement of the angle between the two astrometric fields of view. In this way, the system will be able to check that the variations in the basic angle do not exceed the nominal limits. This and other HK data will be selected by this module for being included in the telemetry frame.

5.5.3. Clocks Distribution Module (CDM)

Its main purpose is to distribute the several timing products to the payload elements. It is a critical module, because the quality of the final products of the mission highly depends on its correct design. The reason is the high resolution of the measurements, which require a precision in the datation of the order of 10 ns. The main timing product is the *Master Clock*, implemented as a rubidium atomic clock offering a frequency of about 6.4 MHz. This product must be distributed uniformly, this is, avoiding phase differences between elements of the instrumentation. Also a real-time clock must be obtained, which can be as simple as a counter of nanoseconds since a given reference time or reset. Other clocks, such as the TDI or readout clocks, will be obtained in the local sequencers or other PDHS elements. These sub-products must be obtained with the highest precision achievable.

5.6. CHAPTER CONCLUSIONS

The main operational principles of the payload of Gaia have been introduced here, together with an overview of its technical implementation. We have focused on its data handling, an implementation option for which has been proposed following the whole data path step by step. Beginning from the simplest data structures to the most complex ones, the focal plane and proximity electronics have been described. It contains the CCD chips which just read pixels and offer samples. These samples are amplified and digitized by the video chain, and the local sequencers compile a whole measurement of a star. Beyond the proximity electronics, the Window Acquisition Manager (WAM) compiles all the measurements and detection data for outputting the data block of the star passage. The video processing modules include not only the WAM, but also the detection and selection algorithms. The former detects the stars entering the focal plane, and the latter selects which will be measured depending on its detected features. The focal plane and proximity electronics, together with the video processing modules, shape each one of the 10 Astro Trail Units.

We have recommended to assign a priority level to each block of data received by the PDHU, which will determine the data to be transmitted during the next contact. Data with lower priority will only be transmitted if there is enough downlink capability. These data are analyzed, coded and compressed, storing the result into one or another queue depending on its priority. The several high-capacity queues will be emptied in order during the next contact. A supervisor controls all this process and the instruments operation. Other modules in the payload and service module have also been reviewed, such as the clocks, attitude and housekeeping subsystems.

This proposal of payload data handling system takes into account all of the requirements of the Gaia mission. These are really restrictive, so that a fully optimized data system was necessary. Here we accomplish this objective, offering a real-time capable system. Furthermore, parallelism is present in many subsystems, while others are proposed as pipelines. All of this eases the final implementation and makes possible the management of the huge amount of data generated by the instruments.

Part III: Telemetry and Communications

Chapter 6

Telemetry Model

In this chapter we describe the telemetry model for the science data of Gaia, developed in cooperation with the Gaia team at the University of Barcelona and requested by the Simulation Working Group (SWG) of Gaia. It contains a compilation with the main telemetry outputs, focusing again on Astro although now we also include a preliminary MBP (medium-band photometer) and elementary housekeeping data. The design of Gaia as described in chapter 3 has been taken into account, as well as the suggestions of Masana et al. (2004). The MBP outputs are taken from Høg (2002b) and Jordi et al. (2003), although its design has slightly changed since then. The RVS data has not been included here.

The formats described in this chapter are based on simulation environments, such as GASS or GIBIS. We will deal with more flight-realistic telemetry data formats in next chapters of this work. Our purpose here was to compile and identify all the data fields to be generated by the Gaia PDHS, which will be input to the telemetry system and, afterwards, sent to ground. This compilation includes the type and range of each data field, while its detailed codification and field size will also be explained in next chapters.

We start in section 1 describing the instrument data model, including a brief introduction to the timing codification and transmission scheme which is needed to fully understand the telemetry model. Section 2 focuses on this timing scheme for the case of simulated data (as in GASS), and describes the generic data to be transmitted. Sections 3 to 5, finally, summarize in tables the information to be transmitted for Astro, MBP and Housekeeping data.

6.1. INSTRUMENT DATA MODEL

6.1.1. Elementary concepts on the timing scheme

Before getting to the into details on the telemetry formats or the data fields to be transmitted, it is clear that we must define the overall transmission scheme for the science data. First of all, the timing scheme must be defined because, as stated in previous studies (Portell 2000), it is tightly linked to the overall transmission scheme (specially when every measurement must be time tagged, which is the most usual case). The study on the time data codification for Gaia is very complex and will be thoroughly described in the next chapter. Nevertheless, here we will describe the elementary concepts needed to understand the general scheme.

Science data will be basically grouped in large blocks named *Data Sets* (DS), each including a *coarse* time field named *Reference Time*. Each of these data sets will contain data from 1 second of observation time (hence the interval between two successive reference times will be 1 second). Afterwards, every measurement is time tagged only with the *fine* time with respect to the reference time. In this way, as we will see in the next chapter, some telemetry occupation can be saved. This data sets philosophy was already introduced in Vannier (2000) and is equivalent to some studies for other space missions (Portell 2000).

The relative time tags for every measurement can take advantage of the intrinsically quantized operation of the focal plane (this is, the TDI-based operation). In other words, since a measurement will only be done at quantized times (with the quantization step equal to the TDI period), we can code the time tags as an integer number of these periods. This general operation is illustrated in figure 6.1, where it can be seen the reference times and the "TDI counts", also noting some instants of sample readout. We can see that this sample readout operation is done only every 2 TDI counts: this is the effect of the 2×2 pixel binning in the

ASM. Because of this, an additional control flag must appear in the scheme, τ_ASM. We can say that the ASM uses a TDI period which is twice the standard one; this additional flag indicates the "TDI part" in which the source has been detected.

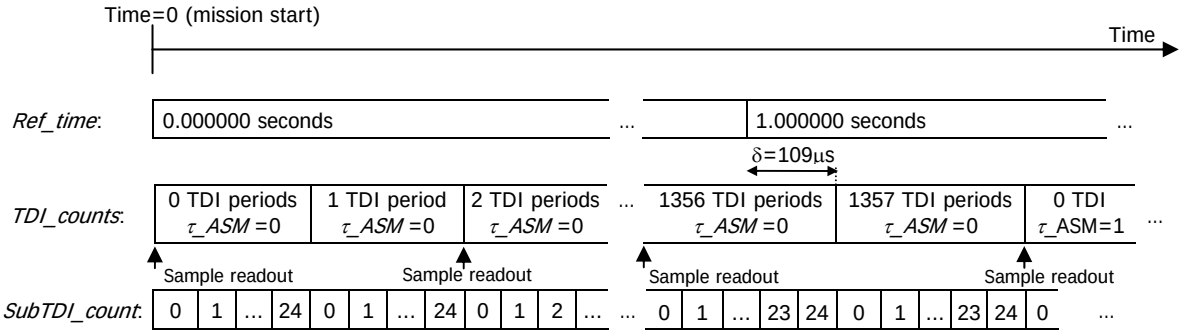


Figure 6.1: Overview of the timing scheme

Since the detection system of Gaia (GD) will offer sub-pixel accuracy (and an along-scan pixel can be translated to a TDI period), "Sub-TDI periods" are also taken into account in order to make possible the codification of this high-resolution time tag. Also, the δ flag is required because one second of time (this is, the length of a data set) is not multiple of a TDI period, and hence the beginning of a new data set will usually not coincide with the beginning of a new TDI period; the δ flag indicates this time difference. We will further describe this timing scheme in the next chapter.

6.1.2. Elementary concepts on the transmission scheme

Now it is clear that the science data of Gaia will be transmitted in Data Sets. Next, we must define how the several data sources (this is, the instruments) will transmit their data within the single communications channel available. This transmission scheme will also be deeply analyzed in the next section, but as a first approximation let us illustrate with figure 6.2 an example of a typical data stream generated by Gaia. We can see here how a data set header, including the reference time, is followed by individual blocks of data from Astro-1 or Astro-2. MBP and RVS data should also be included in this scheme in future works. This sequence is being used by GASS.

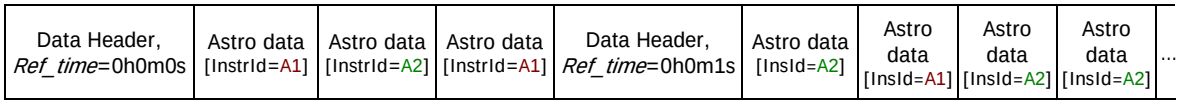


Figure 6.2: Transmission of data sets generated by the instruments

A higher detail of this transmission scheme is shown in figure 6.3, where the main contents of a data set header are shown –as well as the main components of an Astro data block. Each of the *Raw objects* noted in this figure correspond to sources measured by Astro-1 or Astro-2 (or even MBP or RVS).

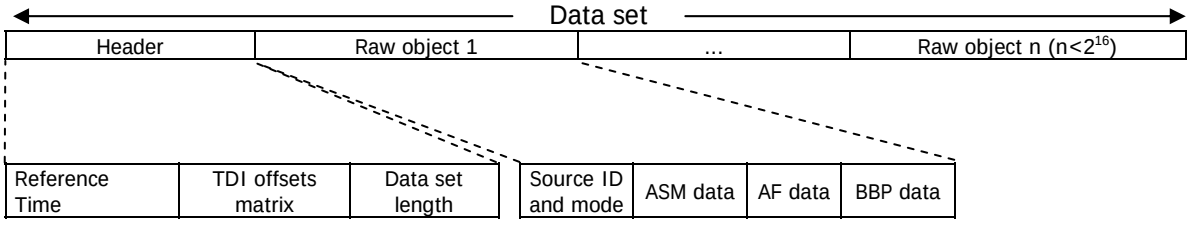


Figure 6.3: Data Source frame

Now focusing on Astro, the contents of one of these blocks of data (Astro-1 or Astro-2) are illustrated in figure 6.4, where the main parts of the science data generated by the instrumentation and PDHS are shown. As already noted in the previous chapters, they are mainly ASM (detection) data, AF data and BBP data. A detail of their contents is also illustrated. An additional source identifier has also been included here specially for GASS, for simulation testing purposes. All of these schemes were developed for Masana et al. (2004) and are being used in GASS and GDAAS.

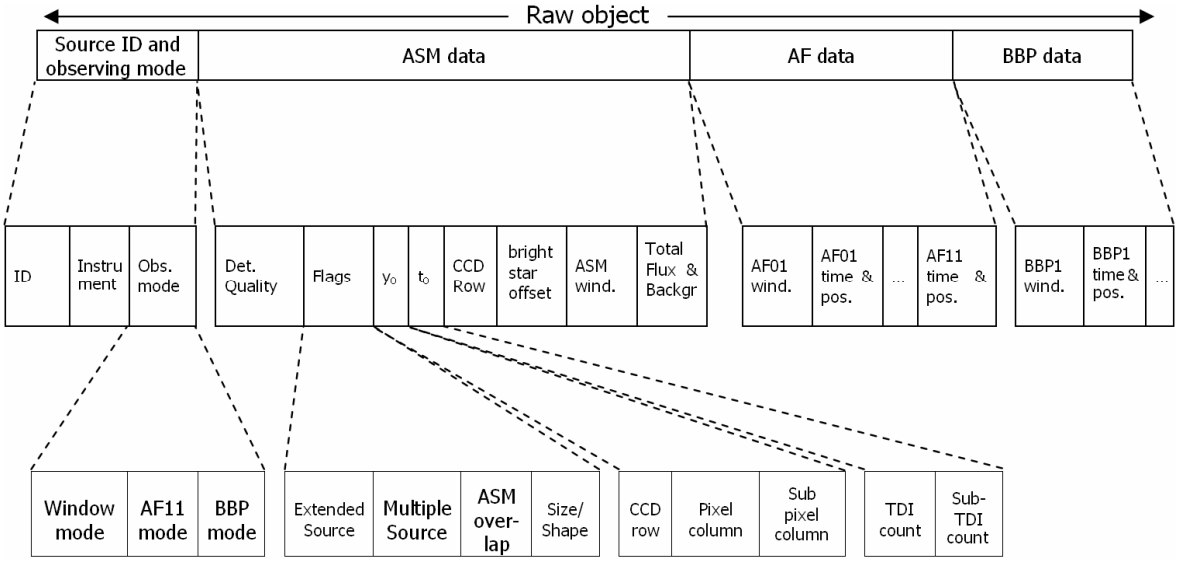


Figure 6.4: Science Telemetry frame

In the next sections of this chapter we will list the sets of data formats for each instrument, source, and data field. However, many details like a bit level coding are not included, neither the sequence to be used when transmitting the data. As a first approximation before going into more details (which will be done in the next chapter), the following assumptions can be taken:

- Data fields will be coded using the minimum number of bits required for the resolution and range specified here. If necessary, this number may be rounded to the next multiple of 8, in order to code it in an integral number of octets.
- If data fields do not have a specific transmission scheme associated, they will be transmitted sequentially following the order listed in the tables, each field starting with the most significant bit.

6.2. TRANSMISSION SCHEME AND GENERIC DATA

6.2.1. Transmission scheme for simulated Gaia telemetry

Gaia will transmit science data to ground using the Packet Telemetry standard (CCSDS 2000, ESA 1988) which uses a Virtual Channelization philosophy (described in the next chapter). However, it can be mostly ignored in the simulations, thus only taking into account the simpler transmission scheme introduced in the previous section. Table 6.1 summarizes it. In this way, science data are grouped in large blocks (the Data Sets), each containing the data from sources detected during 1 second of satellite time. The *Detected Transit Time* (the ASM transit time of

a source) determines the Data Set where the source data will belong to[11]. We will study this issue in detail in chapter 7.

Packet type	Generation rate	Description
Generic and reference data	Once per second	Contains a Reference Time, TDI offset flags and any ancillary data needed for a correct reception and interpretation of science data.
Astro data	Once per source measured	Detection and measurement data.
MBP data	Same as Astro data	Same as Astro data
RVS data	TBD	TBD
Housekeeping data	Once per second (TBC)	Contains Attitude data. Other data like payload status, spacecraft status, TDI corrections, etc. may be introduced in the future.

Table 6.1: Packet types and basic transmission rules for Gaia science data

When generating simulated data files, each file may equal to one of the packets listed in table 6.1. However, in order to simulate as close as possible the transmission process, the packet order should be guaranteed in some way: numbering the files, sequentially combining all data into a single file, etc. As an example, GASS offers an output file with a format like this:

- Generic and Reference Data (→ new Data Set)
- Astro data from a given source
- Astro data from a given source
- ...
- (MBP and RVS data may also be interleaved here following the same scheme; in this way it would be clear that they are related to the previous Reference Data)
- Generic and Reference Data (→ new Data Set)
- Astro data from a given source
- ...

Although this transmission scheme is not optimal, this has been the standard assumed for GASS simulations and for GDAAS. In chapter 7 we will study a way to further optimize this.

6.2.2. Generic and Reference Data

Table 6.2 summarizes the contents of a data set header. The timing scheme indicated in this table defines all the *Detected Transit Times* of each source as relative to the first sample readout time of the ASMs (Masana et al. 2004):

$$\text{Time} = \text{RefTime} + \delta_{i,r,s} + \tau\_\text{ASM}_s \times \text{TDIperiod} + \textit{DetectedTransitTime}$$

where we indicate with the suffixes that δ depends on the instrument, CCD row and CCD strip, and τ_ASM depends on the CCD strip.

[11] We emphasize this because the total transit time of a source over the focal plane is about 1 minute in Astro, so the data may also be classified using the "focal plane exit time" (which is not the case).

Data field		*Type*	*Range*	*Description*
Reference Time		Integer	0...2^{58} seconds (year 2010...2019 at 1 ns resolution)	Reference Time generated each second. It is not CCSDS standard (future improvements may include compatibility).
TDI offsets (δ)	**Astro**	Integer	(18×10×) 0...735μs	Offset of the first readout[12] in each CCD of the focal plane wrt *Ref_Time*
	MBP		(20×2×) 0.00...14.95ms	
	RVS		(TBD)	
ASM TDI flag *(τ_ASM)*		Boolean	(2×) 0 or 1	Number of TDI periods to be added to the ASM *TDI_Offsets*
Data Set length	**Astro**	Integer	(7×) 0...2^{16} sources	Size of each data set (and for each observing mode)
	MBP		(4×) 0...2^{16} sources	
	RVS		(TBD)	
	HK		(TBD)	

Table 6.2: Generic and reference data fields

6.3. ASTRO DATA

This section only summarizes in tables the data to be transmitted. This is the only information that we need in order to continue our work. More details can be found in Masana et al. (2004).

Data field		*Type*	*Range*	*Description*
Source data	**Source Id**	Integer	(Level 9 HTM)	Unique identifier for each source (only for GDAAS verification purposes)
	Instrument Id	Integer	*Astro-1* or *Astro-2*	Which FOV does each source come from
	Window mode	Integer	*Bright*, *medium bright* or *faint*	Window readout mode (G=2-12, G=12-16 or G=16-20)
	AF11 mode	Integer	*Normal* or *narrow*	AF11 readout mode
	BBP mode	Integer	*Normal* or *narrow*	BBP readout mode
ASM data	**Detection quality**	Integer	0...100%	Quality of the detection
	***Flags:* Extended** **Multiple** **ASM window overlap** **Size/shape (*a* & *b*)** **Orientation (*I*)**	Boolean Boolean Boolean Real Integer	*Yes* or *No* *Yes* or *No* *Yes* or *No* (2×) 0.0...8.0 pixels 0°...180°	Detection flags of the source. Size/Shape & Orientation expressed in terms of its equivalent ellipse
	***Detection row:* CCD** **Pixel** **Sub-pixel**	Integer Integer Integer	1^{st}...10^{th} CCD row 1^{st}...1966^{th} pixel 0...24 pixel parts	Across-scan coordinates of ASM transit (as reported by GD)
	***Detec. time:* TDI period** **Sub-TDI**	Integer Integer	TDI period # 0...1361 Sub-TDI period # 0...24	Along-scan coordinates of ASM transit (as reported by GD)

Table 6.3: Science data generated by the Astro focal plane (Source Data and ASM) (I)

[12] This is the first *TDI Clock Stroke* time wrt Reference Time.

Data field			Type	Range	Description
ASM data	***Bright stars:*** **WYxx offset** **Sample height**		Integer Integer	0...46 pixels 2 or 4 samples	Window shape for bright stars (WYxx-type, Høg 2002b)
	ASM Window	**G=2-16**	Integer	(48×) 0...2^{16}-1 ADUs[13]	ASM samples containing the detection
		G=16-20		(25×) 0...2^{16}-1 ADUs	
	Total flux		Integer	0...2^{23}-1 ADUs	Total flux of the source obtained by the Detection Algorithm
	Astro-1 Background		Integer	0...100	Background around the source obtained by the Detection Algorithm
	Astro-2 Background		Integer	0...100	
AF data	**AF readout time**		Integer	(11×) 0…78400 TDI	AL/AC coordinates of the readout times along the AF CCDs.
	AF readout AC position		Integer	(11×) –8…+7 pixels	
	10 AF windows (AF01...10)	**G=2-12**	Integer	10× (16×6×) 0...2^{16}-1 ADUs	Samples from all the acquired windows
		G=12-16		10× (12×1×) 0...2^{16}-1 ADUs	
		G=16-20		10× (6×1×) 0...2^{16}-1 ADUs	
	AF11 window	**G=2-12**	Integer	(16×6×) 0...2^{16}-1 ADUs	Samples from the acquired window
		G=12-16		Normal: (40×1×) 0...2^{16}-1 ADUs Narrow: (12×1×) 0...2^{16}-1 ADUs	
		G=16-20		Normal: (26×1×) 0...2^{16}-1 ADUs Narrow: (6×1×) 0...2^{16}-1 ADUs	
BBP data	**BBP readout time**		Integer	(5×) 0…78400 TDI	AL/AC coordinates of the readout times along the BBP CCDs.
	BBP readout AC position		Integer	(5×) –8…+7 pixels	
	5 BBP windows (BBP1...5)	**G=2-12**	Integer	5× (16×6×) 0...2^{16}-1 ADUs	Samples from all the acquired windows
		G=12-16		Normal: 5× (16×1×) 0...2^{16}-1 ADUs Narrow: 5× (12×1×) 0...2^{16}-1 ADUs	
		G=16-20		Normal: 5× (10×1×) 0...2^{16}-1 ADUs Narrow: 5× (6×1×) 0...2^{16}-1 ADUs	

Table 6.4: Science data generated by the Astro focal plane (ASM, AF and BBP) (and II)

The terms used in these tables, such as *normal/narrow* windows, are taken from the sampling scheme described in Høg (2002b).

[13] Analog-to-Digital Units (equivalent value in e^- TBD)

6.4. MBP DATA

	Data field		*Type*	*Range*	*Description*
Source data	**Source Id**		Integer	(Level 9 HTM)	Unique identifier for each source (only in simulations)
	Window mode		Integer	*WM7*, *WM5*, *WM3* or *WM1*[14]	Window readout mode (Høg 2002b)
MBSM data	**Detection quality**		Integer	0...100%	Quality of the detection
	***Flags:* Extended** **Multiple** **MBSM window overlap** **Size/shape (*a* & *b*)** **Orientation (*i*)**		Boolean Boolean Boolean Real Integer	*Yes* or *No* *Yes* or *No* *Yes* or *No* (2×) 0.0...8.0 pixels 0°...180°	Detection flags of the source. Size/Shape & Orientation expressed in terms of its equivalent ellipse.
	***Detection row:* CCD** **Pixel** **Sub-pixel**		Integer Integer Integer	CCD row # 1 or 2 Pixel # 1...1965 0...25 pixel parts	AC and AL transit coordinates (as reported by GD-MBSM)
	***Detec. time:* TDI period** **Sub-TDI**		Integer Integer	TDI period # 0...67 Sub-TDI period # 0...25	
	MBSM Window		Integer	(12×2×) 0...2^{16}-1 ADUs	MBSM samples containing the detection
	Total flux		Integer	0...2^{23}-1 ADUs	Total flux of the source
	Background		Integer	0...100	Background around the source
MBP data	**MBP readout time**		Integer	(9×) 0…8350 TDI	AL/AC coordinates of the readout times along the MBP CCDs.
	MBP readout AC pos.		Integer	(9×) –8…+7 pixels	
	9 MBP windows	**WM7**	Integer	9× (8×7×) 0...2^{16}-1 ADUs	Samples from all the acquired windows
		WM5		9× (8×5×) 0...2^{16}-1 ADUs	
		WM3		9× (8×3×) 0...2^{16}-1 ADUs	
		WM1		9× (8×1×) 0...2^{16}-1 ADUs	

Table 6.5: Science data generated by the MBP focal plane

6.5. HOUSEKEEPING DATA

Only attitude data is included here. These data are based on quaternions (Masana el al. 2004).

	Data field	*Type*	*Range*	*Description*
Attitude data	**Time**	Integer	0...2^{58} nanoseconds (year 2010...2019 at 1ns resolution)	Time generated at each attitude measurement (once per second, TBC). It should be CCSDS standard.
	Unit vectors	(3×) Real	-1.0...+1.0	Coordinates of e_1, e_2 and e_3 vectors
	Angle	Real	0.0°...360°	Rotation angle

Table 6.6: Housekeeping data (attitude data)

[14] WM7 for G=0-9, WM5 for calibration in G=14-16, WM3 for G=9-14 and calibration in G=16-20, and WM1 for G=14-20.

Chapter 7

Timing and transmission schemes

Now that we know which science data must be transmitted by Gaia to the ground station, this chapter will deeply analyze the problem of coding these data. We will pay special emphasis to time data, in order to combine it with the rest of the data in an optimal way. A compilation of timing requirements will be listed first, which will be used afterwards to define an optimal *time data codification* (TDC). Also, a global transmission scheme is proposed here. These definitions need some simulations in order to guarantee an optimal solution, the results of which will also be shown here.

The first purpose of this chapter is to determine the several requirements for the time data codification of Gaia since they play a crucial role in the mission: resolutions, limitations, items to optimize, etc. We also pursue a definition of a preliminary parameterized proposal for the timing and transmission schemes, which shall be verified and fixed using simulations but should be flexible enough to make possible the inclusion of small changes and accommodations in the final design of the spacecraft. The definition of TDC is tightly linked to the transmission scheme of the science telemetry. The requirements, as well as the codification design, will take into account CCSDS and ESA standards, as well as the design of Gaia described in the previous chapters. It is important to note that, although the main purpose of this chapter may seem related only to time data codification, actually the full structure of Gaia science telemetry will be defined here. Finally, we note that the Astro instrument will be our main interest again, although some schemes include the Spectro instrument (RVS and MBP).

The chapter is organized as follows. Section 1 of the chapter shows the requirements that we have identified for the time data codification. Section 2 describes our recommendations on the timing and transmission schemes for Gaia, providing only a preliminary and parameterized proposal. The models and simulations that we have developed for validating and optimizing the codification will be described in section 3, while the final (fixed) schemes will be presented in section 4. We have included in the annexes the user manuals of the simulators developed here and the list of the software functions.

7.1. REQUIREMENTS OF THE TIME DATA CODIFICATION

All the main requirements related to the *Time Data Codification* (TDC) in Gaia are listed in the following subsections. These requirements will be used in the next section to design an optimal codification, which should fulfill all of them when possible, or give a trade-off alternative when the requirements imply opposite solutions. A numbering is included in order to make easier to reference to each requirement.

7.1.1. General requirements

1.1. Time data codification (TDC) must allow the transmission down to the ground station of the main timing products of each measurement. Here we will consider only the Astro data:

a) Transit time in the ASM1 (if observed with telescope #1) or ASM2 (if observed with telescope #2), as reported by the GD system.

b) Transit time in each one of the CCDs: AF01...11 and BBP1...5

If a direct transmission of all of these data is not possible, then it should be reproducible on ground when combined with other data, like attitude data or housekeeping data.

1.2. It must offer the main timing products (listed before) with the appropriate resolution, needed to obtain the final astrometric resolution of the mission.

1.3. It must allow the obtention of unambiguous absolute values of the main timing products, after doing the appropriate calculations on ground. Hence, each time data field must have the appropriate range.

1.4. TDC should use the minimum telemetry bandwidth of Gaia, opening the way to an optimized codification of all the science data.

1.5. TDC must also permit the transmission (or reproduction) of the several parameters implied in the measurements and, if necessary, codification processes:

a) Actual TDI period for every CCD, and actual time at which the charge transfers occur.

b) Correlation of the main timing products with an on-ground or barycentric clock.

1.6. It should make possible the inclusion of future (small) changes in the timing design of Gaia, for instance:

a) Independent TDI cycles and periods in each CCD

b) Eventual calibrations or corrections (e.g., offsets) in the TDI cycles or periods

c) Others

1.7. TDC must guarantee the correct reception of time data, even when errors in the communication channel happen.

1.8. Inconsistencies in the generation of the time data itself must be avoided. This may imply giving a recommendation or, at least, some hints for the payload implementation (synchronization of on-board clocks between them, calibration wrt an on-ground clock, avoidance of unbalanced delays and drifts...).

7.1.2. Codification requirements

2.1. In order to avoid an unnecessary transmission of absolute time values for each source, the TDC scheme should be implemented using a data set philosophy:

a) Science data will be packed in *data set* (DS) blocks, each one identified by a header containing an unambiguous *reference time* complemented with other ancillary time data. These data sets will be generated at fixed times (e.g., once a second).

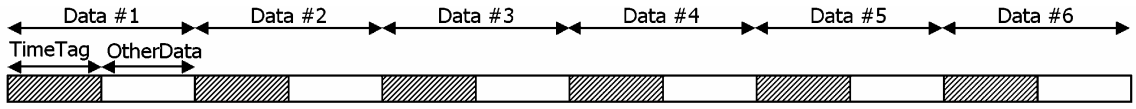
b) These data sets may be partitioned in data subsets (or *time slots*, TS), each identified by a header containing a *tick mark* or *time slot mark* (which will indicate the time elapsed since the previous reference time). These data subsets may also be generated at fixed times (e.g., 10 times per second), although it is not mandatory.

c) After this, the time data for each star will be referred to the active time slot (i.e. to the last tick mark generated). In this way, many less bits will be needed to indicate the detection time of each source.

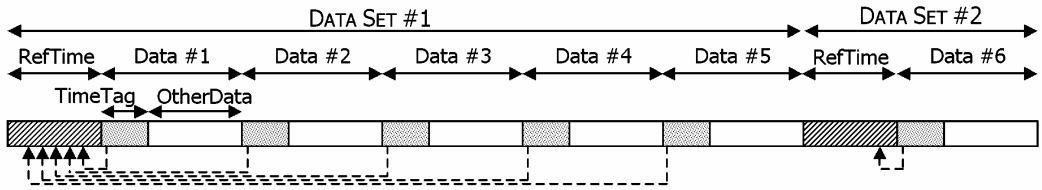
Figure 7.1 illustrates this recommended transmission scheme, showing an evolution from the least optimized method (case *1*) to the most optimized one (case *3*).

2.2. TDC should avoid the transmission of unnecessary time data fields. Whenever possible, a time data field should be omitted (in the transmission) and reconstructed on ground using other complementary time data. As an example, in the Astro instruments:

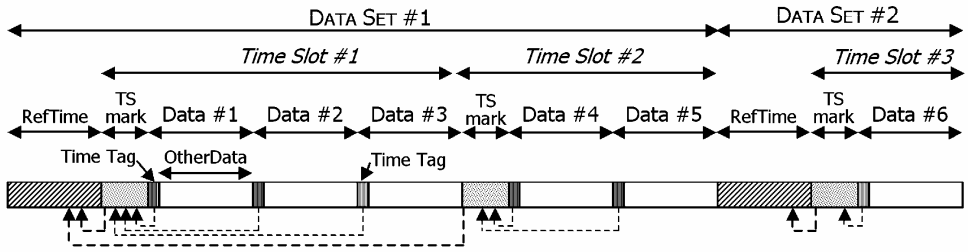
a) Only the *detected transit times* (ASM transit times of the sources) should be transmitted.



***Case 1**: Time tags coded with absolute values. Optimal for very low rates (e.g., <10 tags/second)*



***Case 2**: Data grouped in data sets, time tags coded relative to the last reference time. Optimal for medium rates (e.g., ~50 tags/second)*



***Case 3**: Data grouped in large blocks (data sets), each one divided in several time slots. Time tags coded relative to the last time slot mark. Optimal for high rates (e.g., >100 tags/second)*

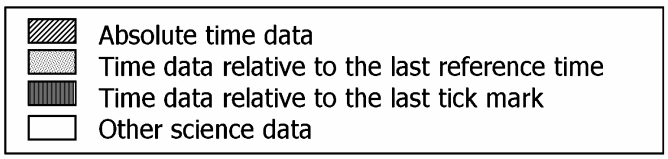


Figure 7.1: Example of TDC optimization using time slots

b) For each CCD of the AF and the BBP, its transit time may be obtained combining the detected transit time with the corresponding attitude data, geometry calibration and readout operation rules. Literal transmission of readout times should only be used if unavoidable, e.g., if there are problems for assuring a correct reconstruction of these times on ground.

c) In order to reconstruct correctly on-ground the readout operation done in the focal planes of Gaia, the TDI times (charge shifts, readout operations, etc.) must be known with the highest possible precision. In the current design, a single TDI period (integration time for a single pixel) will be used for all the CCDs of a focal plane, so only one TDI offset (time elapsed between a given reference time and the next TDI readout operation) should be transmitted.

This requirement is illustrated in figure 7.2.

2.3. The TDC scheme should be defined and designed in a way compatible with the CCSDS and ESA standards, specially *packet telemetry* (CCSDS 2000, ESA 1988) and *time code formats* (CCSDS 2002). Some examples are the following:

a) The format of the time data fields should be compatible with the CCSDS standards at Level 3 or Level 2 (although it is not mandatory). However, a UTC-compatible time format is forbidden (we must avoid the inclusion of any leap second), and the use of a pre-fixed epoch is not recommended (we should use a privately defined epoch for Gaia, since it will have its own clock and time scale, so we should not define a Level 1 compatible codification).

b) Part of the time data may be included in the *secondary header* of the *source packets* (SP).

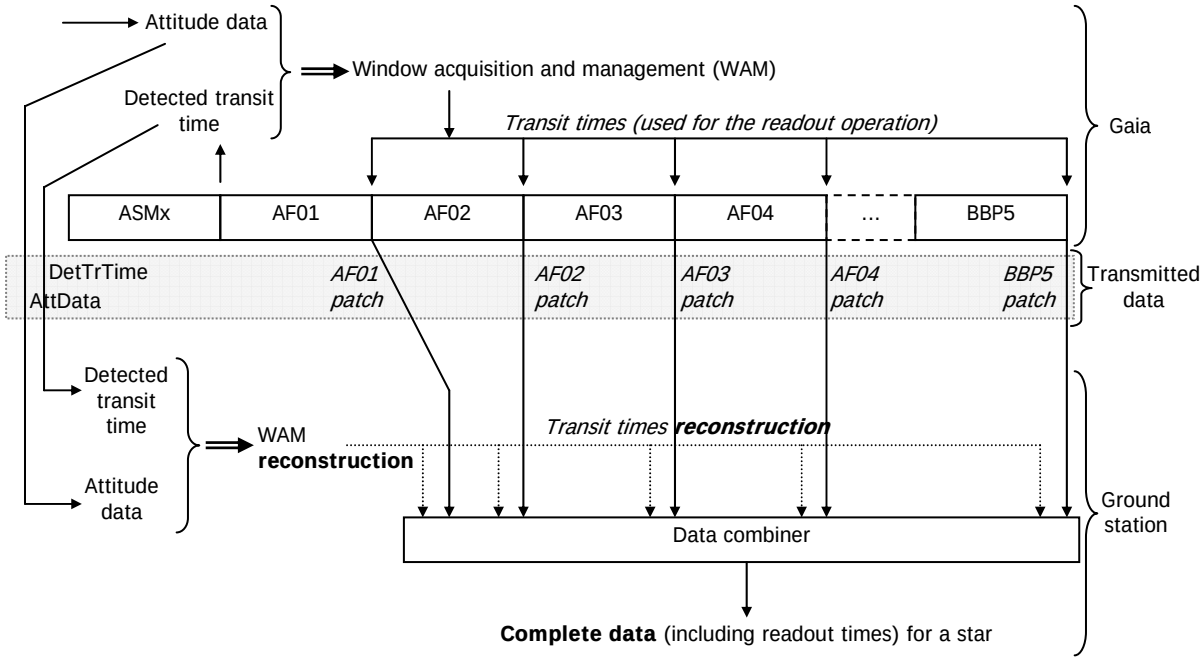


Figure 7.2: Reconstruction of transit time data from attitude data and detection time

c) Multiplexing of the eventual *virtual channels* (VC) and/or applications must be taken into account, which may lead to a transmission sequence in an order different than that at which the data was generated (e.g., for packets with very different sizes). The TDC recommendation, therefore, must include the CCSDS multiplexing option chosen, i.e., specifying the way Gaia data is distributed among virtual channels and application process identifiers (APIs).

7.1.3. Resolution and range requirements

3.1. The resolution of the time data must take into account the required resolution of the final astrometric data products (they directly affect the requirements on time data resolution).

a) As proposed by some scientists of Gaia, about 1ns resolution for most time data would be preferable. This would avoid errors on the measurement of the basic angle, apparent glitches in the interpretation of the attitude, and other serious problems. The latest calculations revealed that the minimum absolute requirement for this resolution is about 16ns, although at least 5-10ns is highly recommended.

b) Some data fields may not need as high a resolution as 1ns, due to their intrinsically lower resolution. For example, about 1μs resolution may be enough for the detected transit time: the detection algorithm will probably offer a resolution worse than this. However, a TDI-based resolution (e.g., $1/10^{th}$ of the current TDI period) with respect to a given reference time is preferable, because this will be the real resolution offered by the on-board detection algorithm. Afterwards, on ground, this time may be converted to a nanosecond-based absolute time.

As shown in figure 7.3 (provided by F. Arenou), the best resolution currently given by any detection algorithm is about 0.025 samples along-scan. Recent results from the OBD Working Group indicate a resolution of up to 0.014 AL pixels. Taking into account possible future improvements of the algorithms and samples of only 1-pixel along-scan (as in AF01), we can assume a final resolution of 1/100 pixel along-scan in the best of the cases. Therefore, in order to minimize the quantization errors and following the recommendations by Gaia scientists, a time resolution of at least 1/200 pixel should be enough (which would increase the scatter of the derived values in about 4%), although 1/500 pixel is highly recommended.

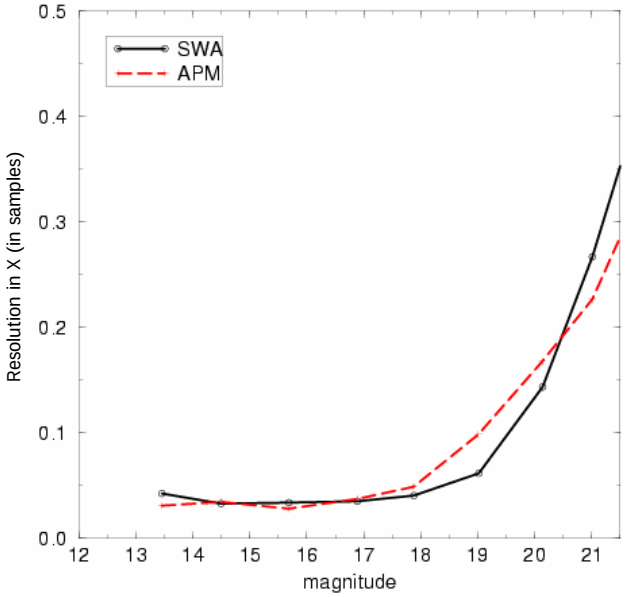


Figure 7.3: Resolution given by two different detection algorithms studied for Gaia

c) It is important to note that, although a lower resolution (e.g., TDI-based) may be enough for downlinking some time data fields, the *internal* calculations and operations must be performed with the appropriate resolution. If not, important errors may propagate in the time data. As an example, let us study the following situation (based on realistic values):

Resolution = 1/500 TDI

TDI period value = 736.6μs, and therefore TDI/500 = 1.4732μs

→ A possible implementation for labeling the detected transit time would be a counter of "TDI fractions" (reset at each reference time used).

Master clock = 6.4MHz

→ Basic period (period of the master clock, or T_{MC}): 156.25ns.
The nearest multiple to 1.4732μs (a "TDI fraction") is T_{MC}×9 = 1406.25ns

Detected transit time = 31.42 TDIs (offered by the detection algorithm), this is, 15710 "TDI fractions".

When using the correct resolution, we obtain:

Detected transit time = 15710×1.4732μs = 23143.972μs

But if we use a T_{MC}-based resolution (a multiple of the period of the master clock), we would obtain:

Detected transit time = 15710×1.40625μs = 22092.1875μs

→ In this case the error is of 1051.7845μs (1.43 TDI periods), leading to an unacceptable astrometric accuracy.

A possible improvement to this measurement method is to use frequency multipliers for obtaining a better resolution, but the results would also be unsatisfactory: the error would not be uniform, increasing as the detection time does.

Another possible implementation would be to offer these time data fields as an absolute time value (with a good enough resolution), and convert them to a TDI-based value only when inserting them into the telemetry stream. However, we should also take into account possible errors when using this option: assuming that the payload can use an internal resolution of, e.g., 1 nanosecond, figure 7.4 shows how to correctly manage the detected transit time value (including some numerical examples in brackets). In this figure we see how Gaia should "truncate" the absolute

nanosecond-based transit time in order to perform on-board and on-ground operations with exactly the same resolution. This is, Gaia must not perform its internal operations with the nanosecond-based transit time, but with the TDI-based time. If not, in the example of figure 7.4, there would be an error of 690ns in the on-ground reconstruction of this time value. In general, it would lead to an uncertainty in the detected transit time –added to the uncertainty already introduced by the detection algorithm. However, because of the high enough coding resolution, it only represents a 0.09% of the TDI period –and, in any case, it would imply at most 0.1% TDI error. It remains to be studied if this error is high enough to take it into account.

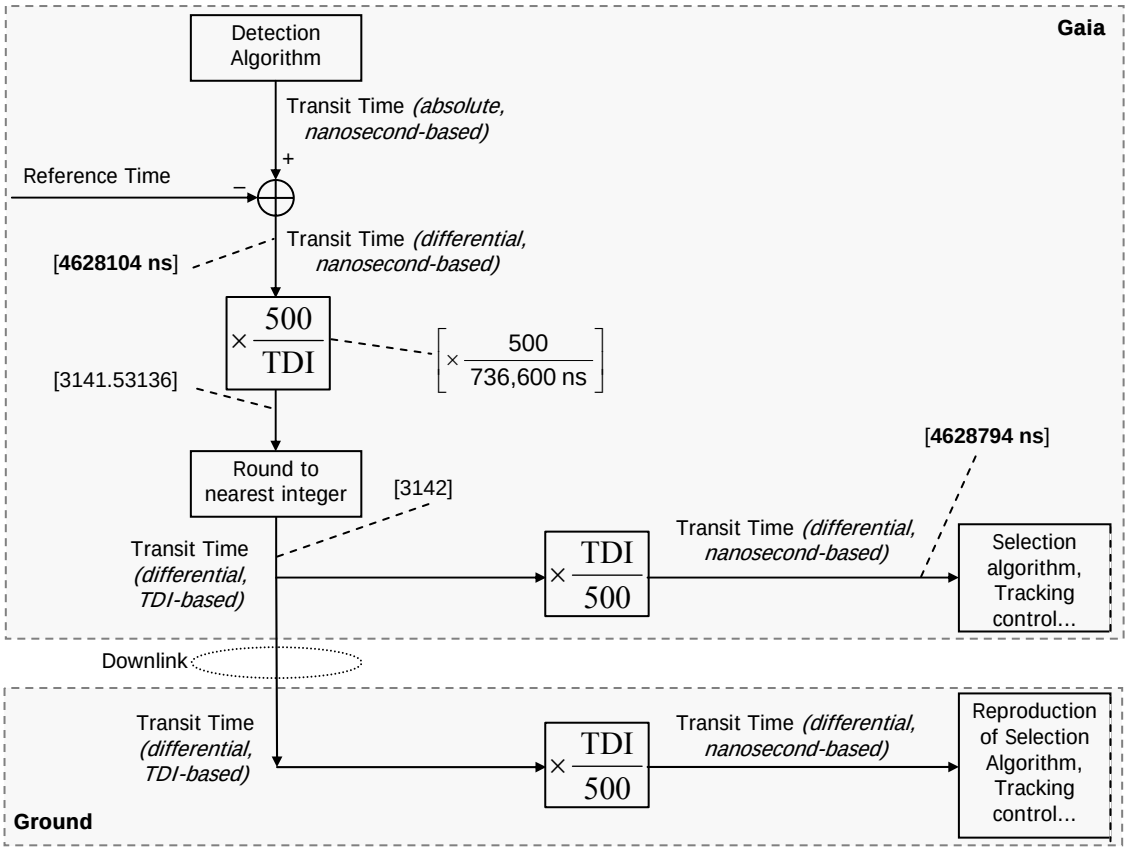


Figure 7.4: Correct use of TDI-based resolution in transit time data

Also, following the example of figure 7.4, if a lower resolution (e.g., 1μs) is used instead, an even higher uncertainty would be introduced in the results: a transit time of 4628104ns would lead to a coded value of 3142, while a truncation towards the nearest microsecond would be 4628μs, leading to a coded value of 3141 –adding another error of 0.2% TDI.

3.2. Adequate ranges for each data field must be defined:

a) The recommended Time Reference is 2010.0 (see chapter 2 and Bastian 2004).

b) Absolute time fields, e.g., the reference times of each data set, should be compatible with the worst predictions for Gaia, i.e., a delayed launch and an extended mission. We may consider as the worst case a 5-years delayed launch (i.e., in 2015) and an extended duration of 10 years, so the absolute time data fields should be capable to code up to 2025.

3.3. The combination of resolution and range requirements give a first approximation to the number of bits required for absolute time fields, this is, data fields that shall code the time

in absolute value and with no "reset" during the whole mission. The following expression shows how to obtain this value[15]:

$$\left\lceil \log_2\left(\frac{1}{res} \times 3600\text{sec/hr} \times 24\text{hr/day} \times 365.25\text{day/yr} \times range \right) \right\rceil \tag{7.1}$$

Where:

$\lceil\ \rceil$ represents the top integer of the inner value (e.g., $\lceil 5.03 \rceil = 6$)

$\log_2(\ )$ stands for the binary logarithm, which equals to $\frac{\log(\ )}{\log 2}$

res is the time resolution (in seconds); e.g., *res* = 10^{-9} seconds stands for a 1ns resolution.

range is the total time range to be coded; e.g., in the case shown in requirement 3.2, *range* = 15 years.

Using this expression, a resolution of 1ns (req. 3.1a), and a range of 15 years (req. 3.2), we obtain a coding requirement of 59 bits, which even make possible a range of about 18 years (i.e., until 2028, using the time reference 2010.0). If octet-based codification is required then 64 bits should be used. It is obviously a "worst case", and shall be regarded only as indicative. On the other hand, if telemetry restrictions turn out to be really important, the resolution may be reduced up to 5ns or 10ns, thus reducing the coding requirement to 57 or 56 bits respectively, leading to a maximum range of 22 years.

7.1.4. Communication requirements

4.1. The TDC scheme and transmission scheme must provide robustness to the transmission of Gaia science data. This is, the coding and schemes must be developed in a way that an unrecoverable error in the transmission, leading to a lost frame on ground, implies the minimum amount of lost scientific data. A clarification on this can be seen in Pace (1998), while Geijo (2002) studies some effects of transmission errors and recommends a solution. Also, the intrinsic operation of the telemetry system must not affect the correct reconstruction of time data. The following are the implications of this requirement:

a) The decoding and reconstruction of time data cannot depend on the data transmitted much time ago. For example, let us assume that Gaia transmits an absolute reference time only every minute of measurement time, and differential time data (relative to this reference time) for each source. Then, if the frame containing the reference time cannot be recovered on ground, we would lose all the time data of the sources measured during 1 minute. In other words, and using the predicted environmental parameters (350 stars/s and 1.2Mbps downlink), a simple error in 1 bit of the reference time would imply the loss of about 72 Mbits of science data (from more than 20000 sources). Instead of this, we may transmit a reference time, e.g., once a second, partial tick marks more than 10 times a second, and finally the time data of every source as relative to the last tick mark. Furthermore, the reference time may have a resolution of 1 second, sending sub-second data only in the differential fields. In this way the reference times would be sequential –so that a lost packet may be easily reconstructed. Note that this transmission scheme should be taken only as an example, not as a requirement. An example of this problem is shown in figure 7.5.

b) The TDC cannot assume a sequential transmission of the data generated by the several data sources of Gaia (e.g., instruments, attitude and housekeeping modules, etc.) for using it to reconstruct time data. The communications system of Gaia, based

[15] We have considered 365.25 days/year.

on the packet telemetry standard (CCSDS 2000, ESA 1988), may reorder the source packets from several applications or virtual channels. In other words, the reconstruction of time data should not depend on other data transmitted via a different application process identifier (API) or virtual channel (VC). Figure 7.6 illustrates this. However, if this kind of dependency is needed, then the correct reconstruction of data packets must be guaranteed, i.e., the data handling system of the ground station must be able to reconstruct the original sequence (e.g., by referencing the next packet in the sequence with a unique identifier). In other words, all the time data must be as self-consistent (and independent from other data) as possible. When it is not possible to make independent time data fields, their dependencies should be grouped in the smallest possible groups and their sequence order must be signaled in some way.

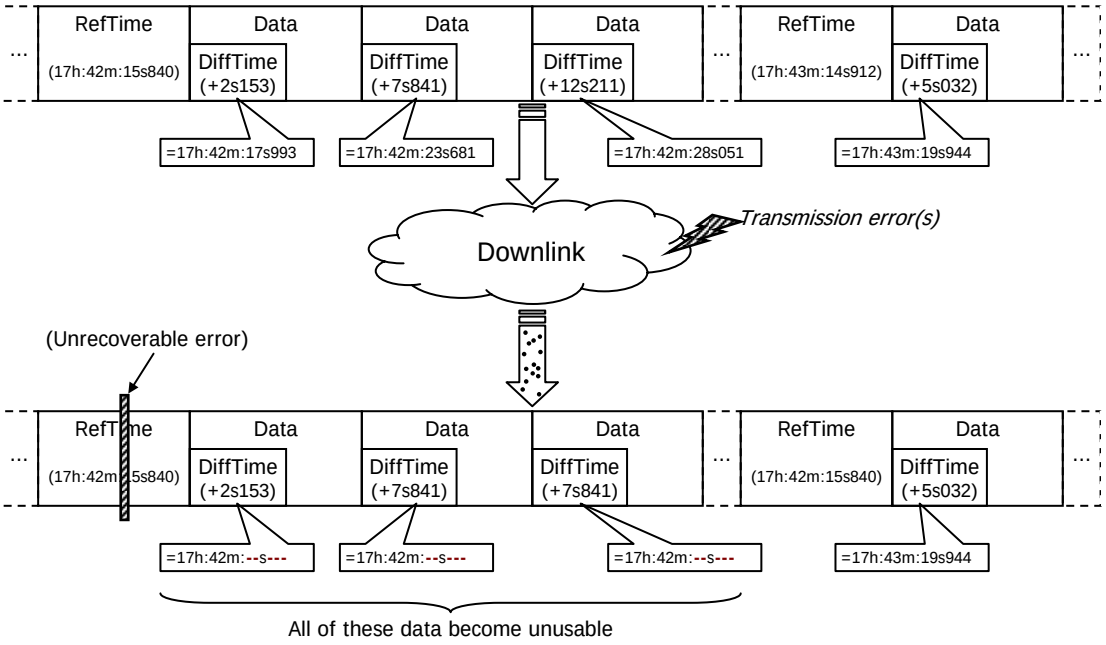


Figure 7.5: Weakness of a TDC example in front of transmission errors

7.1.5. Payload requirements

5.1. The payload clocks and counters must be highly stable and synchronized, in order to generate consistent time data products. This requirement involves many implications where engineers and contractors must take care:

a) A single spacecraft-based *master clock* is highly recommended for Gaia, which will surely be implemented with an Rb-type atomic clock (de Bruijne 2003a). Although there may be a redundancy scheme (to guarantee its operation during the entire mission), this master clock must be the only one offering a fundamental time reference to both the payload and service module of Gaia, including all of their subsystems. All the counters and sequencing signals will be obtained from this clock.

b) Some studies (de Bruijne 2003a) request a precision of 10-50ns in the CCD-sample time tags, and an absolute time datation with at least 1ms resolution. However, here we will assume more recent recommendations from Gaia scientists implying a knowledge of the focal plane operation (i.e., TDI and readout operations) with 10ns resolution or better, and detected transit times with 1/200 TDI resolution or better. All of these values should be confirmed or updated as soon as possible.

c) The master clock must eventually be calibrated wrt an on-ground clock, e.g., a TAI-based or a barycentric-based clock. Also eventual synchronizations may be performed, if necessary (de Bruijne 2003a).

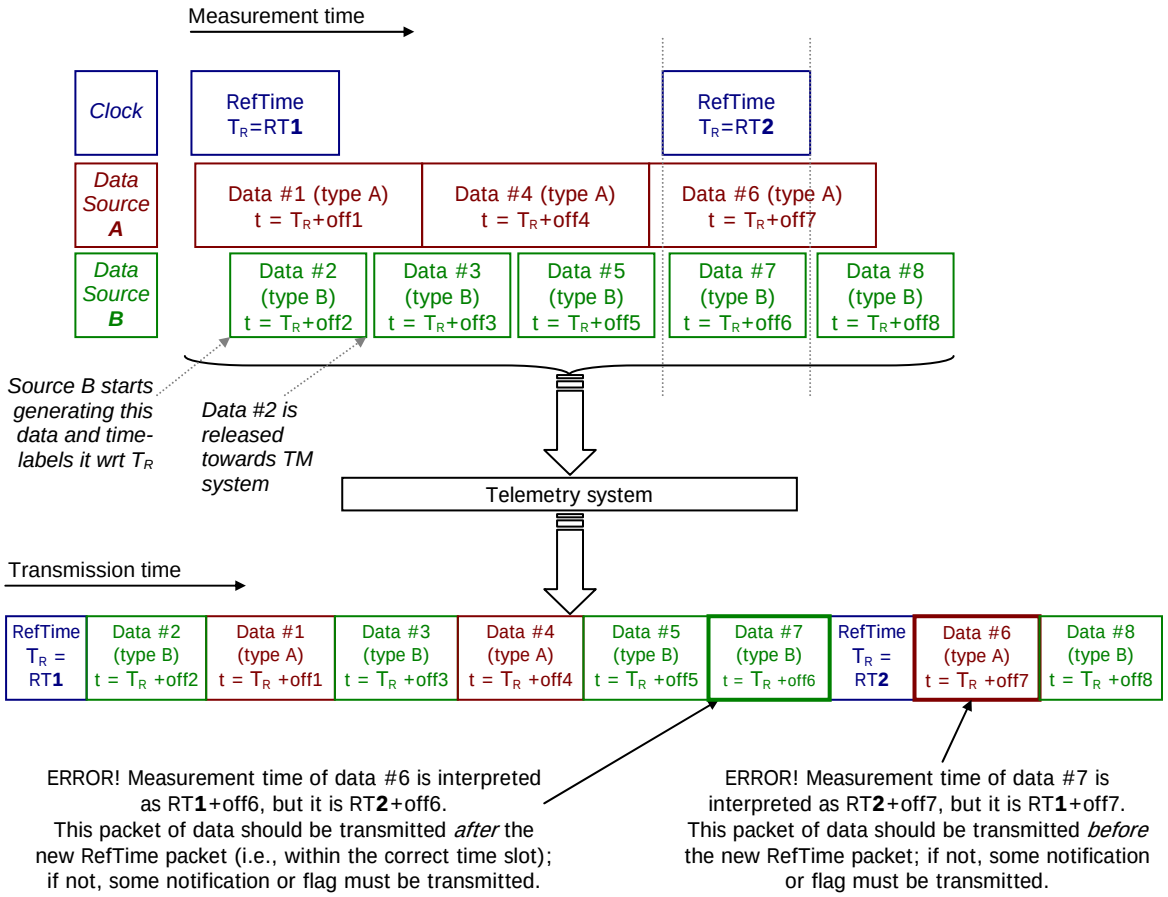
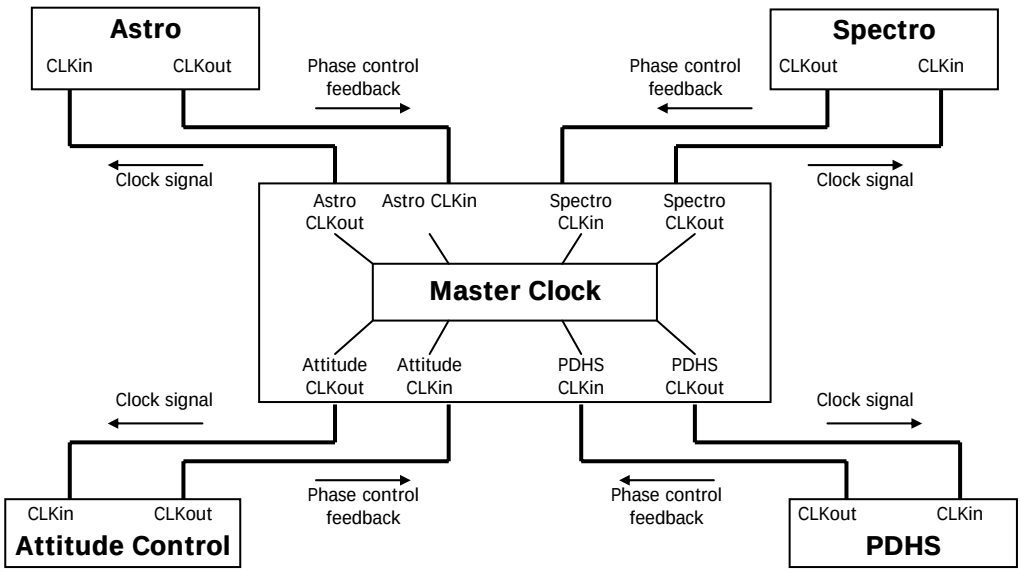


Figure 7.6: Wrong interpretation of time data due to packet reordering in the telemetry system

d) The distribution of this clock data to all the elements of Gaia should be done using a symmetrical structure, specially for the payload instruments (Astro and Spectro) and the PDHU. It is essential for obtaining consistent time tags in all of the instrumentation set. We say "symmetrical structure" for indicating that distribution lines (transmission lines) must be implemented using the same cable type and length. Moreover, bi-directional synchronization signals may be exchanged to guarantee that all the instrumentation is operating with the same clock signal, fulfilling the phase difference requirements. Figure 7.7 shows an example of this.

e) Some instruments, like the Astro focal plane (with 180 CCDs), will include many internal counters or timers (e.g., for the CCD sequencers). These counters must be synchronized, specially in the case of the Astro focal plane, and they also must have an excellent stability. To be specific, the clock that controls the charge shift from one pixel to the next (thus implementing the TDI operation itself) must be the same for all the CCDs of the focal plane, with a very high precision (in the nanosecond scale). This must also be accomplished for all of the pixels of each CCD. Also, time tagging must be performed with a very high precision in all the CCDs.

These and other requirements should be further studied and determined by the corresponding engineering teams, e.g., by the PDHE contractors. ESA (2000), chapter 5 of this thesis and de Bruijne (2003a) may be somehow useful in this task.



Indications: All of the 8 transmission lines have the same length and use the same cable type.
The master clock controller uses a single internal clock and outputs its signal, adding independent delays for each output if necessary; these delays are controlled by, e.g., some kind of PLL, using the clock feedback data from the instruments.

Figure 7.7: Example of a system that generates consistent time data products

7.1.6. Other requirements

6.1. The TDC scheme must be developed according to the new design of Gaia (EADS/Astrium 2002), sometimes named "Gaia-2". For this, chapter 3 of this work can be taken as the baseline parameters –although recent changes in the design or parameters must be taken into account whenever possible. Some examples of this configuration (the most relevant for the TDC) are the following:

a) There is only 1 astrometric focal plane, with 180 CCDs, on which the images from both astrometric telescopes are combined.

b) The TDI period is constant all over the focal plane (736.6μs per pixel in Astro). Furthermore, all the parallel TDI clocks (for the charge transfer operation) will be assumed to evolve like a single one, with no phase differences. It implies that the TDI offset matrix (Masana et al. 2004) will not be needed anymore –now we just need one single TDI offset per focal plane.

6.2. This new Gaia design may still have some changes or modifications, so the TDC must be parameterized in order to adapt it in an easy way to small changes in this design. Numerical values should be used only to offer a first implementation of the codification.

6.3. The current TDC design only takes into account the Astro instrument. However, the TDC should be prepared for the future inclusion of Spectro data.

7.2. PRELIMINARY AND PARAMETERISED SCHEMES

Here we describe a proposal of codification and transmission schemes that fulfills the requirements listed in section 7.1. Afterwards, an expression for restoring the time data fields will be introduced. The code format for time data fields will be described next, and finally a multiplexing option (with a control system for maintaining the packet sequence order) will be explained. With all of these elements introduced, the last subsection will describe the contents of the recommended codification for Gaia time data –although this definition will also be related to the rest of Gaia data. We will specially take into account the CCSDS and ESA recommendations for packet telemetry (CCSDS 2000, ESA 1988).

7.2.1. Time codification method

First of all, we must define a concept that will help us in defining a consistent and optimized time data codification scheme: the *readout finish time* or ROFT. It can be defined as:

> The time when the pixel line currently loaded in the readout register has been totally readout, and the next pixel line is ready to be loaded immediately. That is, the time of the immediately next TDI clock stroke.

The reason why we consider this ROFT as being exactly coincidental with a TDI clock stroke (not with the end of a readout operation) is that several issues related to time, data handling, and even astrometry depend on the charge shift operation, not on the readout operation itself[16]. Just after this ROFT, the next pixel line is loaded into the readout register (i.e., the charge is shifted in parallel). Figures 6.1 and 5.7 may help in understanding this. With this concept in mind, we are ready to describe the main features of the method selected for transmitting time data from Gaia to ground:

- Science data from measured sources will be transmitted in data sets and time slots, as shown in Figure 7.1 (case 3).
  - The length of a data set (DS) will be 1 second (measurement time), this is, it will contain data from sources detected (transited over the ASM) during 1 second. Note that their total measurement time will be much longer than 1 second (about 56 seconds in Astro).
  - A new DS will be generated every second, and when all its source data is available it will be transmitted. Actually, it will not be immediately transmitted, but sent to the PDHS and stored on board until the next contact with the ground station.
- Each DS will contain a header with:
  - The reference time, with 1-second resolution and absolute coding (i.e., number of seconds since a given epoch).
  - A set of flags revealing the CCD operation, specially the time at which the first readout operation is finished (the time at which the first sample of the DS is available). This is named the *first effective ROFT* and should be coded with 1-ns resolution (TBC), although up to 16-ns resolution may be used. This first effective ROFT will be sent in two parts: *TDI offset* (δ, or the first ROFT) and *ASM TDI flag* (τ), both explained in Masana et al. (2004). Taking into account the features listed in chapter 3, the maximum value of the first effective ROFT (wrt the reference time) is twice the TDI period, this is, 1473.2μs for Astro.
- The time data within a DS will be coded wrt the first effective ROFT.
  - Using the concepts of TDI offset (δ) and ASM TDI flag (τ), the absolute value of the first effective ROFT is:

$$1^{st}\text{ eff. ROFT} = \text{reference time} + \delta + \tau \qquad (4.1)$$

- Each DS will be divided in several time slots, each labeled with a *time slot mark* (TSM).
  - A time slot may (and most times will) contain data from several sources.
  - Time slots will have a given maximum length (in measurement time).

[16] We recall that the TDI operation, i.e., the charge shift operation (from one pixel line to the next) is a parallel transfer, while the readout operation (of a pixel line stored in the readout register) is a serial transfer. The parallel transfer is managed by the TDI clock (with a period of 736.6μs in Astro), while the serial transfer is managed by clocks at different rates depending on the operation to perform: readout (low speed) or flush (high speed).

- Time slots will contain a maximum number of sources of value TBD, e.g., 20.
- TSMs will be coded wrt the first effective ROFT, in order to take advantage of the implicitly quantized readout times and charge transfer times. The resulting value will have a TDI-based resolution (1/500th of the TDI period), and may take negative values: a TSM may fall between the current reference time and the first effective ROFT. The most negative value that a TSM can take is –1000 (for a resolution of 1/500th TDI), for TSM = reference time.
- The beginning of a new time slot will coincide with the transit time of a new source over the ASM, this is, the TSM value will equal to the detected transit time of the first source within the time slot.

Figure 7.8 illustrates this coding scheme. Dotted gray arrows indicate the route to follow for obtaining an actual time, e.g., the detected transit time of source 5 is: DTT5 + TSM2 + (τ × TDI_period) + δ + RefTime.

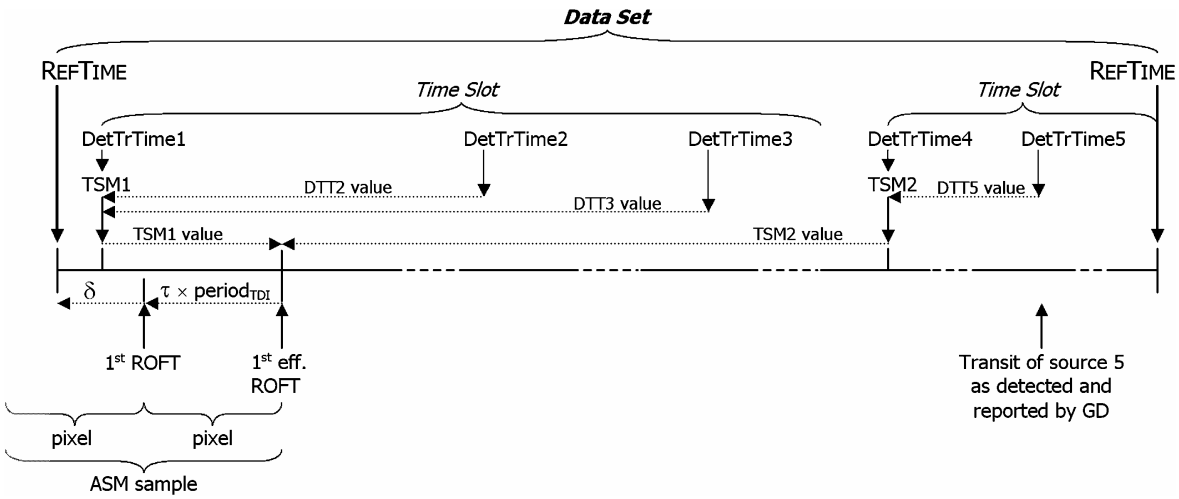


Figure 7.8: Illustration of the time codification method selected for Gaia

- The only time data of the source transmitted to ground will be its detected transit time.
  - This is the time output by the GD (Gaia Detection) algorithm, i.e., the transit time of the PSF of the source over the ASM1/2 readout register.
  - The rest of transit times (for AF and BBP) will be reconstructed on ground using attitude data, geometry data and focal plane behavior data (i.e., the ROFT). They will be transmitted only if this reconstruction is found to be impossible or extremely difficult, and in this case they should be transmitted using a differential coding wrt a nominal set of transit times.
- Each source will indicate its detected transit time wrt the last TSM.
  - The transit time of the first source of a time slot will always be zero (i.e., it will coincide with the TSM). Therefore, its transmission is not necessary. This condition implies that every time slot will contain at least one source –if not, it is not necessary to generate a time slot.
  - The maximum value of a transit time is equal to the maximum length of a time slot. It can also be noted as the maximum offset from the time slot mark.
  - Detected transit times will be coded using a TDI-based resolution: 1/500th of the TDI period, this is, 1/500th of a pixel (1/1000th of an ASM sample), although this resolution may be as worse as 1/200th of the TDI period. GD will offer a worse resolution than this, so we assure that coding and quantization errors will be negligible.

7.2.2. Reconstruction of absolute time data

The most important time data field to transmit is the detected transit time of a source. The following expression shows how to reconstruct it on ground from the several partial data sent by Gaia:

$$Absolute\_DetTrTime = \mathbf{RefTime} + \mathbf{TDI\_off} + \mathbf{ASM\_flag} \times TDI\_per + \mathbf{TSM} \times TSM\_res + \mathbf{DetTrTime} \times TrTime\_res \quad (4.2)$$

Where:

RefTime is the reference time (absolute, 1-second resolution)

TDI_off is the corresponding TDI offset (1-nanosecond resolution, TBC)

ASM_flag is the corresponding ASM TDI flag (1 or 0)

TDI_per is the TDI period (with 1-nanosecond resolution)

TSM is the value of the time slot mark, expressed as a number of *TDI_per* fractions

TSM_res is the TSM resolution (=1/500 *TDI_per*)

DetTrTime is the detected transit time of the measurement (number of *TDI_per* fractions)

TrTime_res is the resolution of the detected transit time, equal to *TSM_res*

It is important to note that *TDI_per*, *TSM_res* and *TrTime_res* are not transmitted by Gaia, but will be known (or fixed) beforehand. The only data transmitted within a data set are highlighted in bold face in the previous expression.

It is also important to note that the master clock of Gaia will be a simple counter, which may be reset as necessary. The generator of the reference time, a counter of seconds since a given epoch, may be considered as a secondary clock which should already take into account the resets of the master clock, i.e., it should accumulate the corresponding time values in order to offer an absolute counter without leap seconds and glitches. If not, the corresponding time correlation and/or decoding should be performed on ground.

The delimiting time values of the data sets (and coded in the reference times) cannot be considered ideal, but the best approximations that can be obtained from multiples of the master clock period. I.e., they are not exact physical times, but very precise technical ones. Furthermore, other issues like Doppler shift due to the radial velocity of the spacecraft, or time dilation due to the gravity of the Earth, make the satellite time values different than on-ground time values.

7.2.3. Time code formats

There will be a total of 5 time data fields in this proposal of TDC, each one with its corresponding codification. The following is a list of these fields and codification features:

- **Reference time**: in the data set header. It is CCSDS standard:
  - *CCSDS unsegmented time code* (CUC). As explained in CCSDS (2002) this format simply contains a counter of seconds since a given epoch.
  - Level 2 (Agency-defined epoch)[17]. Epoch: 2010.0 (Bastian 2004).
  - Number of octets of coarse time: 4 octets → Approx. range (years): 136.
  - Number of octets of fine time: 0 octets → no sub-second time (i.e., resolution is 1 second).
  - Total size of code: 5 octets (1 octet P-field, 4 octets T-field[18]), this is, 40 bits.

[17] If necessary, a Level 1 code can be selected instead, i.e., using CCSDS-recommended epoch 1958-Jan-1 (TAI). It would imply no extra coding bits –the current code size already fits Level-1 requirements.

- **TDI offset (δ)**: in the data set header. It is not CCSDS standard:
  - Meaning: time elapsed between the reference time and the first ROFT of the data set.
  - Coding: simple binary number.
  - Resolution: 1 nanosecond for Astro (TBC). Spectro may use worse resolutions (e.g., 50 nanoseconds) with satisfactory results, because of its longer TDI period.
  - Range: 0…1 TDI period, this is, 0…736600ns for Astro (TBC). For Spectro it will count up to 16.37ms (TBC).
  - Total size of code: 20 bits for Astro (TBC). The code size for Spectro, if using a resolution of 50ns, would be 19 bits.
- **ASM TDI flag (τ)**: in the data set header. It is not CCSDS standard:
  - Meaning: whether or not a full TDI period has to be added for obtaining the first *effective* ROFT in the ASM.
  - Coding: single bit (*yes* or *no*)
  - Resolution and range: N/A
  - Total size of code: 1 bit.
- **Time slot mark (TSM)**: within the data set. It is CCSDS standard:
  - *Agency-defined*: fixed number of octets, which conform a binary number that represents the fine time (sub-second time) within the data set.
  - Level 3: the binary number does increment monotonically with time, although not uniformly, and this number is reset at the recycling instant.
  - Coding: TDI-based.
  - Resolution: 1/500 TDI period.
  - Range: up to 1 second. In Astro this is 1s / 1.4732μs = 0…678794 fractions of TDI period. If Spectro also uses 1/500 TDIs resolution, the range would be 1s / 32.74μs = 0…30543 fractions of TDI period.
  - Total size of code: 32 bits for Astro (4 octets: 1 for P-field, 3 for T-field), and 24 bits for Spectro (only 2 octets for T-field). If CCSDS-compatibility is not mandatory we could then save 8 bits per TSM, because the P-field would not be needed anymore.
- **Detected transit time**: linked to the scientific data of each source. It is not CCSDS standard:
  - Meaning: transit time of the centroid of the PSF of the source (as reported by GD) over the ASM readout register, expressed as the number of TDI periods (including fractions) since the last TSM.
  - Taken from: GD system.
  - Restrictions: it must be optimized because of its higher transmission rate, so it cannot be restricted to an integer number of octets. This time data will surely be compressed together with the rest of source data.
  - Coding: TDI-based.
  - Resolution: 1/500 TDI period.

[18] See CCSDS (2002). P-field or *preamble field* indicates the type and format of the time code, while the T-field or *time field* contains the time data.

- Range: equivalent to the maximum length of a time slot, which will be determined hereafter by simulations (absolute maximum: the equivalent to 1 second).
- Total size of code: TBD together with the code range.

7.2.4. Integration within packet telemetry

As explained in the previous sections, all of the science data generated by Gaia will be transmitted in data sets. Their integration within the telemetry standard (CCSDS 2000, ESA 1988) will be done in this way:

- **Data set header**:
  - Transmitted as a separate *application process* (AP).
  - Coded as a whole *source packet* (SP).
  - The reference time will be included in the SP *secondary header*.
  - Other data (TDI offsets, ASM TDI flags, etc.) will be included in the SP *source data* field.
- **Data set contents**:
  - Transmitted in separate APs, depending on the instrument.
  - A time slot will equal to a source packet.
  - The TSM will be included in the source packet secondary header.
  - The data corresponding to the several sources of the time slot will be included in the *source data* field.

7.2.4.1. Telemetry structures used in this codification scheme

- **Virtual channels**:
  - Only 1 virtual channel (VC) will be used for the Gaia science data related to the measurements (Astro and Spectro).
  - Another VC will be used for housekeeping and attitude data (TBC).
  - Other channels may be used for telecommand or other ancillary data (TBD).
- **Application processes (APs)**: there will be several application processes in Gaia generating data, each one with its corresponding *application process identifier* (API) and being multiplexed using one or another VC, as shown in table 7.1.

Figure 7.9 illustrates this multiplexing scheme. Some clarifications on this recommendation:

1. APIs and VCs have been numbered with letters and roman numbers to avoid any restriction on its numerical identifiers. Engineering teams will determine the final numerical identifiers.
2. If necessary, the BBP science data may be included in API #D or in separate APIs, instead of being included in APIs #B and #C. This proposal assumes that BBP data will be included in the APIs corresponding to Astro, together with their associated AF data.
3. An alternate multiplexing option may take into account the CCD row of the detection, separating the data in different APs depending on the row. It has not been considered here because of the estimated average number of sources entering the focal plane (less than 400 per second), which may lead to less than 4 sources per channel and, therefore, an excessive overhead in the transmission.

API #	AP (data source)	Data	VC #
A	Clock system & PDHU	Data set header (reference time, TDI offset, DS ancillary data…)	I
B	Astro-1 (preceding)	Science data corresponding to sources measured with Astro-1 telescope	I
C	Astro-2 (following)	Science data corresponding to sources measured with Astro-2 telescope	I
D	MBP	Science data measured with the MBP	I
E	RVS	Science data measured with the RVS	I
F	AOCS	Attitude data	II
G	HK subsystem	Housekeeping data	II

Table 7.1: Multiplexing scheme recommended for Gaia science telemetry

We can consider the telemetry streams from each AP as "virtual sub-channels", each one containing a given scheme of source packets (SPs) as it will be described in section 7.2.5. See CCSDS (2000) or ESA (1988) for the explanation of some of these data fields.

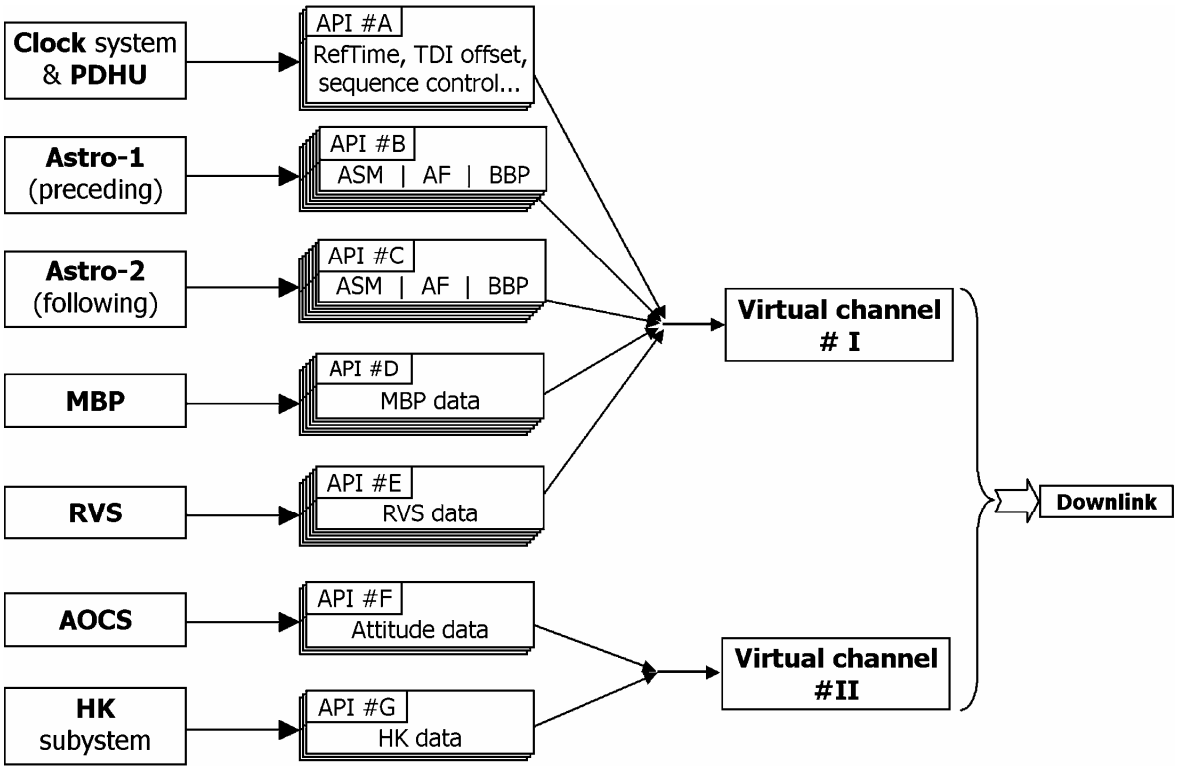


Figure 7.9: Illustration of the multiplexing process recommended for science telemetry

7.2.4.2. Transmission errors and packet sequence control

An error in the reception of certain time data fields may imply an important burst of errors. More in detail, an error in the reception of a data set header may imply that all the data measured during 1 second[19] may become unusable. Let us review, first of all, the main data fields transmitted in the DS header, and the implications of a transmission error on them:

[19] In average, about 350 sources will be scanned every second in Astro, which can represent more than 1 megabit of raw data after packetisation.

- Reference time: in principle it is totally sequential, i.e., Gaia transmits this time value with 1-second resolution every second, so its value is incremental –with a constant increment value.

  → An error on the reception of an isolated reference time may be easily reconstructed.

- TDI offset and ASM flag: the behavior of a focal plane is deterministic, this is, we may deduce its behavior using the data from a previous (or next) DS.

  → An error on the reception of a few TDI offsets or ASM flags may be easily reconstructed. In fact, it is not mandatory to send these flags every second; we propose to do it just in order to avoid unnecessary calculations on ground –and to assure on ground reconstruction of the behavior of the focal planes.

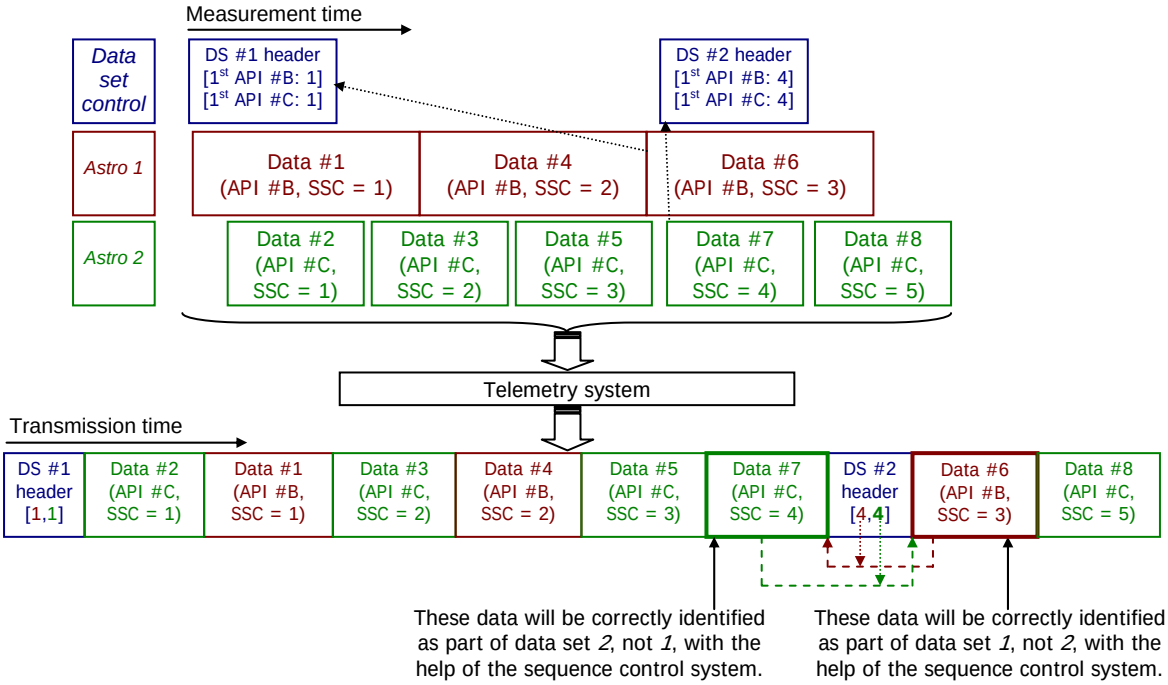


Figure 7.10: Packet reordering with the proposed sequence control system

Therefore it is clear that a loss of a single data set header may not be so serious. But as explained in previous sections, the telemetry system may transmit the several SPs in a way that their order in transmission time is different than their order in measurement time. Sometimes, given the differential coding scheme for time data, it may imply a wrong time data interpretation on ground. As the data set header is the natural delimiter between data sets, the responsibility of maintaining the packet sequence order must rely on it. Therefore it must include some flags indicating which are the first packets (of each AP) of the current data set. We will use the *source sequence count* values (SSC, individual counters for the source packets of each AP, CCSDS 2000), to indicate it. On the other hand, we have seen that the loss of a single data set header is permissible –at least from the point of view of its data. Therefore, we should use a packet sequence control system with enough robustness to permit a single loss. A simple method is to add a redundancy, transmitting not only the current but also the previous SSC values. In summary, the data transmitted in API #A (DS header) will include the following fields as *data set sequence control*:

- Current and previous SSC for API #B (Astro-1 data).
- Current and previous SSC for API #C (Astro-2 data).
- Current and previous SSC for API #D (MBP data).
- Current and previous SSC for API #E (RVS data).

Each one of these fields will include the value of the *source sequence count* field corresponding to the first source packet (of each AP) in the current and previous data set. In this way, the case shown in figure 7.6 can be easily solved (even with some transmission errors), as shown in figure 7.10. This figure shows only the *current* SSC values.

Also, there is another detail to take into account: the generation of a packet with the data set header will be almost instantaneous (say, at most a couple of TDI cycles), while the generation of an Astro packet will last about 1 minute. Therefore, it may be interesting to reduce this latency discrepancy by, e.g., adding some transfer delay to the data set header packets –this is, before they enter the TM system. This issue will be treated in more detail in the forthcoming sections.

7.2.5. Source packet contents for each Application Process

7.2.5.1. AP #A: data set header

Features of a source packet (SP) containing the DS header:

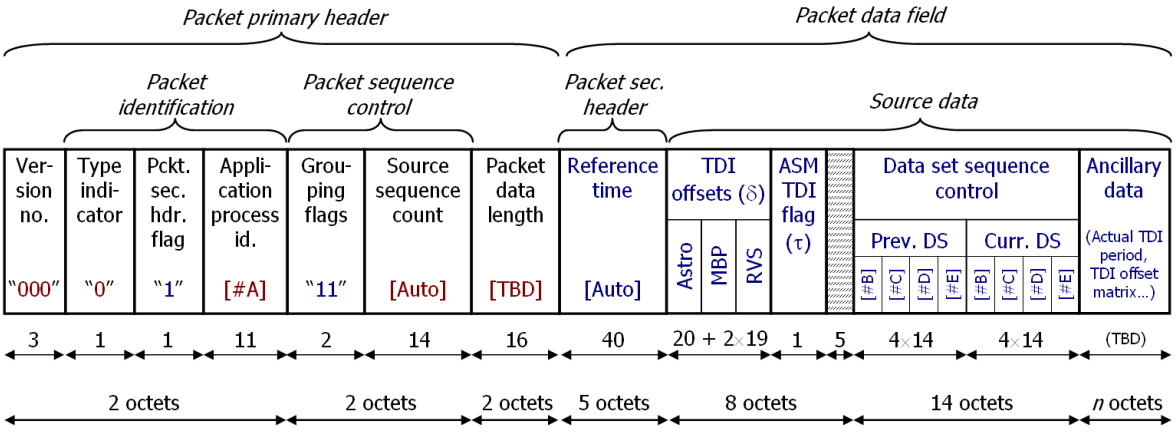
- General:
  - SPs are generated at a fixed rate: exactly 1 per second.
  - SPs will be internally time-labeled (e.g., by the PDHU) with a sub-second time equal to zero, i.e., with the current value of the *seconds counter*[20] of Gaia.
  - Every SP will determine the beginning of a new data set, the contents of which will be generated by other APs (and transmitted in source packets with different API values).
- Contents:
  - *Packet secondary header flag* = "1" (secondary header present).
  - *Grouping flags* = "11" (SP not belonging to a group of SPs).
  - *Packet secondary header* contains:
    - Reference time: it indicates the coarse time, this is, absolute time elapsed since the selected time reference (2010.0) and 1 second resolution, with the format described in section 7.2.3. It is the time tag of an AP #A source packet.
  - *Source data* contains:
    - TDI offset: this field indicates the first ROFT of the current data set as the nanoseconds elapsed since the reference time, with the format described in section 7.2.3. We assume the same TDI period for all the CCDs of a focal plane, distributed with a unique parallel clock, so only 1 TDI offset per focal plane will be needed. Therefore, this field is composed of the following sub-fields:
      - TDI offset for Astro
      - TDI offset for MBP
      - TDI offset for RVS
    - ASM TDI flag: a single flag indicating whether or not a full TDI period has to be added to the detected transit times, due to the 2×2 sampling of the ASM (Masana et al. 2004). It has to be confirmed if we can assume a uniform behavior of all the

[20] Secondary clock obtained from the master clock. It is a simple counter of the seconds elapsed since a given epoch, with no resets, taking into account the considerations given at the end of section 7.2.2.

CCDs of the ASM[21]; if not, this data field would be composed of a 2×10 matrix, each value corresponding to an ASM CCD. No flag of this kind is currently envisaged for Spectro.

- Data set sequence control: a set of numbers indicating the *source sequence count* value of the source packets (from other APs) immediately following the AP #A packet. As explained in 7.2.4.2 there will be a total of 8:
  - First SSC values for the previous data set: for APs #B, #C, #D and #E.
  - First SSC values for the current data set: for APs #B, #C, #D and #E.
- Ancillary data: the presence or absence of data set header ancillary data will be determined by the *data length* field of the SP primary header. The following are possible ancillary data that may be included if necessary:
  - Actual TDI period used in every instrument. It may be a matrix if necessary, indicating the actual TDI period of each CCD of every instrument.
  - TDI offset matrix: if a single TDI offset per focal plane is not enough, this matrix will indicate the TDI offset for each CCD of each focal plane. It may be needed if independent TDI clocks are used for each CCD.
  - ASM TDI flag matrix (set of Boolean values): if a single flag is not enough, this matrix will indicate whether or not a full TDI period has to be added to the detected transit times, depending on the ASM CCD where the source is detected.

Other fields of the SP (*API* value, *source sequence count*, *packet data length*, etc.) will be automatic, determined by engineering teams, or fixed by the telemetry standard. These fields are shown in red in figure 7.11, while fields assigned by this proposal are indicated in blue.



Note: The total size of the ancillary data must be an integer number of octets, as well as the total size of a source packet. This requirement of the packet telemetry standard is the reason why we have inserted the shaded 5-bit field (dummy data).

Figure 7.11: Packet structure and contents for AP #A (data set header)

The total size of an AP #A source packet, as seen in the figure, is 33 octets (264 bits). The data represent only 211 bits, so there is about 20% overhead –the highest in this proposal, because of the small size of these packets. DS header packets, generated once a second, will occupy 264 bps of the telemetry. These values have to be confirmed, either by industrials (i.e., verifying that a single TDI offset and a single ASM TDI flag are enough for all the focal plane), or by simulations (i.e., testing lower time resolutions).

[21] Also assuming that both strips of the ASM (for Astro-1 and Astro-2), operating with different instances of GD, behave uniformly. If not, 2 flags will be needed.

7.2.5.2. APs #B and #C: Astro science data

Features of a source packet (SP) containing data related to Astro data:

- General:
  - SPs are generated when new source data are available.
  - The generation of one of these SPs (specially its time tag) marks the beginning of a new *time slot* (TS). A time slot is linked to a given AP (whether it is Astro-1, Astro-2, MBP or RVS) and cannot be applicable to other APs. Every AP has its own set of time slots.
  - SPs are the coding equivalence of a time slot, this is, a source packet equals to a time slot and contains all of its data.
  - SPs contain data from several sources, output by the corresponding application process. The total size of a SP (i.e., total number of sources, and length in measurement time or in bits) is determined by several issues.
- Contents:
  - *Packet secondary header flag* = "1" (secondary header present).
  - *Grouping flags* = "11" (SP not belonging to a group of SPs).
  - *Packet secondary header* contains:
    - Time slot mark (TSM): indicates the relative time wrt the first effective ROFT of the current data set, with the format explained in section 7.2.3. It is the time tag of an AP #B/C source packet. This relative time tag can be converted to an absolute time tag by following expression 4.2.
  - *Source data* contains:
    - Number of sources in the time slot. This is necessary because the *packet data length* field of the primary header does not provide complete information (the data size of a source is variable because of the compression, the different sampling mode depending on its brightness, etc.). The size required for this field depends on the maximum number of sources of a time slot (*max_Astro_TS_sources*), which will be determined by simulations in the following sections.
    - Coded and compressed data related to multiple sources, from 1 to *max_Astro_TS_sources*. Source coding and compression details are partly described in chapter 6 and will be further studied in the next chapters, but to a first approximation the data related to each source will contain:
      - Measurement mode: Parameters used for measuring the source, such as the window mode or the width of the AF11/BBP windows.
      - ASM data: detection data related to a given source, e.g., across-scan coordinate of its ASM transit (CCD and pixel), total flux, or other detection flags. It also includes the *detected transit time* (DTT) of the source, in a TDI-based coding (see section 7.2.3 for coding details). This transit time is the time tag of a measurement.
      - AF data: set of patches read for the given source.
      - BBP data: set of patches read for the given source.

Other fields of the SP (*API* value, *source sequence count*, *packet data length*) will be determined by engineering teams or will be automatic. As shown in figure 7.12, the minimum size of a source packet generated by AP #B or #C is 10 octets (80 bits). We may assume an average size of about 3000 bits for every source (Masana et al. 2004), a typical compression

ratio of 3 and, e.g., 10 sources per packet. Therefore, an AP #B/C source packet may be about 10000 bits in length –which implies an overhead lower than 1%.

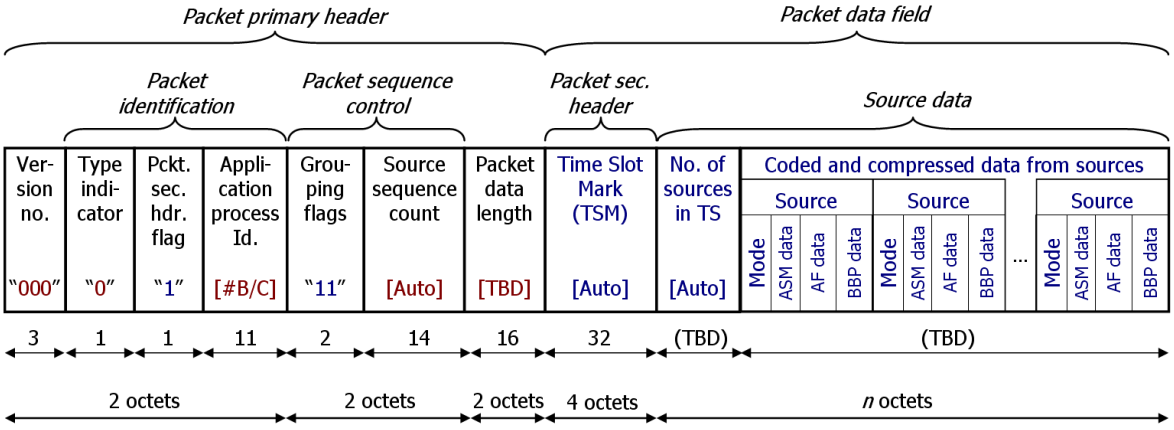


Figure 7.12: Packet structure and contents for AP #B/C (Astro data)

7.2.5.3. APs #D and #E: Spectro science data

The contents of the source packets related to these APIs are still TBD. However, its main structure will mostly be similar to SPs related to APIs #B and #C (Astro data). The time slots methodology will be maintained, as well as the time data codification using TSMs and detected transit times.

7.2.5.4. APs #F and #G: attitude and housekeeping data

The contents of the source packets related to these APs are still TBD. In a principle, they are not required to use the same data set scheme as used for Astro and Spectro data: they will surely be transmitted at a much lower rate than Astro/Spectro data, such as once per second, so they may include their own reference time. Also, packet sequence control is not needed for these data (for the same reason).

7.2.6. Overview of the data generation process

As seen before, science data will be grouped in data sets, each containing smaller time slots. The operation of coding these data is rather intuitive when data sizes from each application process are similar (as seen in figures 7.6 and 7.10). But the real Gaia mission will contain several data sources, each one with very different *data generation latencies*[22]:

- AP #A (clock, PDHU and focal plane monitoring): less than 1 second.
- AP #B/C (Astro focal plane): about 57 seconds.
- AP #D (MBP): about 89 seconds (TBC).
- AP #E (RVS): about 145 seconds (TBC).

Despite of these differences, the correlation between the data set headers and its contents must be maintained. Here we propose the following solution, offering a simplified processing, pipeline-capable and with an almost constant processing load:

[22] It is the total generation time, this is, from the creation of the data (e.g., the transit of a source over a sky mapper, or the generation of a reference time) to the completion of this data (e.g., when the source exits the focal plane, or when the TDI offsets, etc. have been measured).

- First of all, we must determine the application process with the highest latency in generating the data. In a principle, it will be the AP #E (the RVS), with a ~145s latency (Jordi et al. 2003). This latency calculation should be the worst of the cases, i.e., assuming a source with the highest proper motion opposite to the along-scan direction. Also a small security margin (e.g., 1%) should be added.
- AP #A will be used as the main reference (as proposed all along this chapter), which will create a *reference queue*:
  - A reference time will be generated every second, and the focal plane parameters (TDI offset and ASM TDI flag) will be measured.
  - All of these data will be fed into a queue, the size of which will be equivalent to the maximum latency measured (*max_meas_latency*, ~145 seconds TBC).
  - Each block of this queue, corresponding to a data set, will have enough room to include a set of pointers to source data generated by the rest of APs. These pointers will correspond to sources transited over the sky mappers during the data set time lap, and should somehow include (or indicate) the detected transit time of every source.
  - About *max_meas_latency* seconds after its generation, a data set header (and its set of pointers) will exit the reference queue, and will be fed into the codification and telemetry system –together with the source data indicated by the pointers; data that will already be complete because of the delay introduced by the queue.
- Each of the other APs (for Astro and Spectro) will:
  - Assign a unique identifier to each detected source, the value of which will be used as the pointer included in the reference queue. It may be its detected transit coordinates (including the detected transit time), or a simple counter with a large enough recycling time (a 24-bit counter would permit a maximum of about 115.000 sources/s).
  - Allocate enough memory in a *sources queue*, where the AP will store the several data that will be generated while measuring every source (during its measurement time). Assuming that *max_meas_latency* is 145s, about 256MB should be enough for queuing all the data from the sources measured by Astro and Spectro.

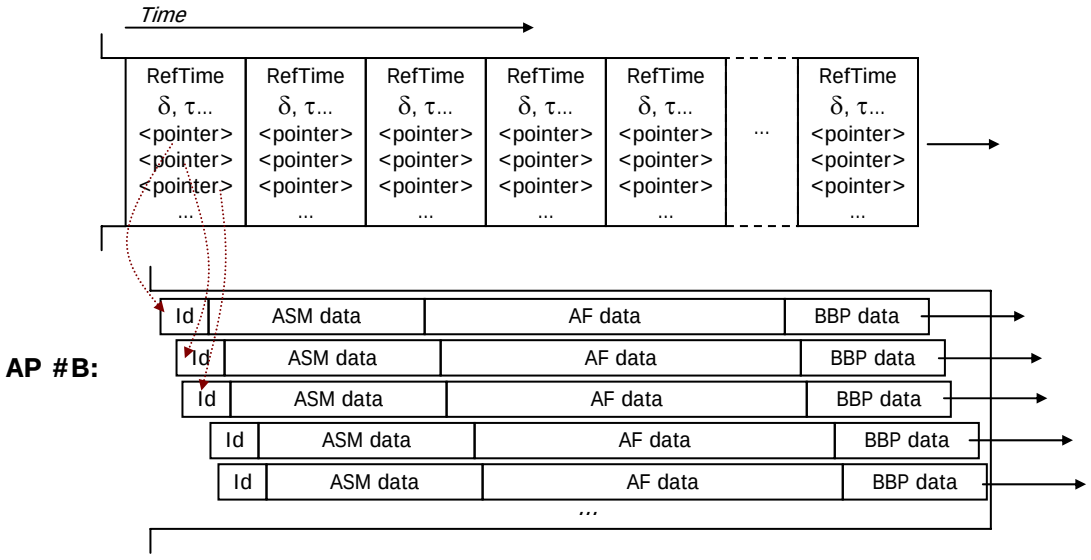


Figure 7.13: Concurrency and interrelations in the data generation process

- When serving each block of the reference queue:
  - The data set header will be coded, formatted and fed to the mass storage, ready for being transmitted.
  - Using the pointers, the sources queue will be accessed for retrieving the data from each source of that data set, coding, formatting and compressing it, and finally storing the result in the mass storage.
  - The data from this processed source can now be deleted from the sources queue.

Figure 7.13 illustrates this proposal for generating the data (only considering Astro-1). We should now introduce the following two concepts (or parameters) required for creating the source packets –the coding equivalences of a time slot. They will be verified while serving each block of data, for knowing if a new source packet must be generated or not:

- *Maximum_TS_Sources*: maximum number of sources that can be included in a single source packet (or time slot). Its optimal value will be determined with simulations. Exceeding its value may imply reception problems, specially over a noisy channel –so that a single unrecoverable error may imply the lost of many data (Pace 1998).
- *Maximum_TSM_Offset*: maximum length (in measurement time) of a time slot, i.e., the maximum value that a TSM may take. In other words, it is the coding capacity of the TSM field, expressed as the number of TDI periods (and fractions) elapsed since the last reference time. Its optimal value will also be determined with simulations.

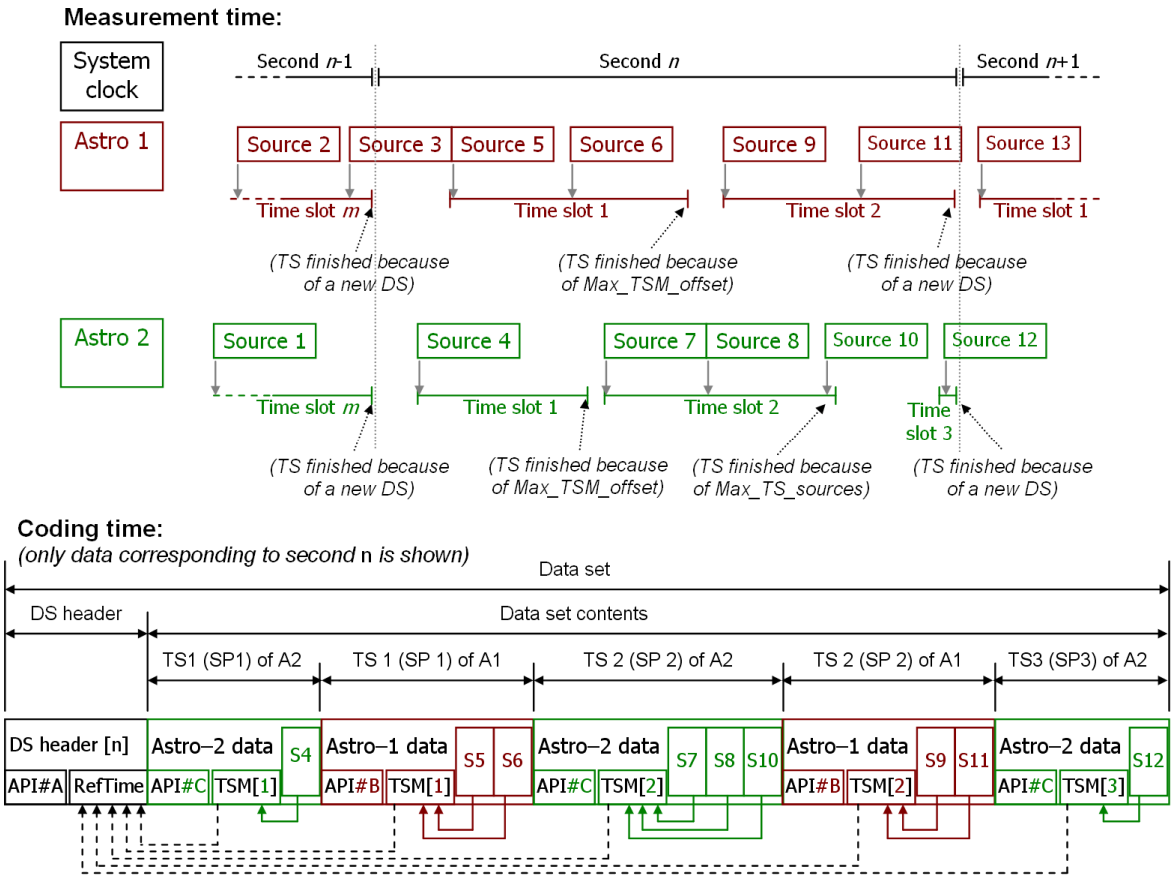


Figure 7.14: Measurement time and coding time. Example with Astro-1 and Astro-2 data

It is important to note that this coding scheme creates two different time domains: *Measurement time* and *coding time*. The measurement time domain is the only one to be taken into account –the one included in the coded data. The coding time domain is just taken into account for engineering purposes and should be avoided from the science point of view; in fact, this is the main purpose of the packet sequence control system, explained in section

7.2.4.2. Figure 7.14 shows the difference between these two time domains: in measurement time, there is only one time scale determined by the master clock. Over this, the several time slots are created, which overlap depending on the source data generation for each instrument. In the same figure 7.14 can be seen how these several time slots do not necessarily have the same length, neither the full time scale must be covered by time slots: there can be "holes", where no time slot is needed because no science data is generated. On the other hand, while in measurement time the time slots of the several instruments can overlap, in coding time all of these time slots are multiplexed over a single time line, thus transmitting sequentially the different SPs generated (each containing data from different sources).

7.2.7. Low priority and outdated data

As indicated in chapter 5 and in Høg et al. (2003), Gaia will generate science data with different priorities. An example is the *partial observations* (PO), which are data from rejected observations (because of electronics, storage or telemetry limitations), from discarded detections (cosmic rays or noise), etc. and may be as simple as the transmission of only ASM data (without AF and BBP data). These data will probably be stored in a separate queue of the mass storage, being transmitted only if there is enough downlink capacity. Therefore, they will most probably be transmitted as outdated data –i.e., totally apart of the measurement sequence. It implies that the data sets scheme is no longer useful for these data. These are some proposals for solving this problem:

1) Abandon the data set structure, i.e., each outdated source packet would be standalone. In this way, every source packet would contain an absolute time tag, as well as the data from several sources within a small time range. This option may imply an important overhead but, on the other hand, there would be no restrictions on the transmission order of the outdated source packets.
2) Complete the transmission scheme proposed in section 7.2.4.1, e.g., by including another virtual channel for outdated science data. Then an equivalent multiplexing scheme may be used for this data (even without the need of using another virtual channel):
   - API #H: outdated data set header
   - API #I: Astro-1 outdated data
   - API #J: Astro-2 outdated data
   - API #K: MBP outdated data
   - API #L: RVS outdated data

   This option would maintain the optimized structure recommended in this chapter, although the data codification would get a bit complicated. These data sets would have no relation with the main data sets, and would be generated on the fly –as the several outdated data shall be transmitted.

Much effort should be done towards this direction, thus offering a full, multi-priority and consistent time data codification, but this is beyond the scope of this work.

7.3. SIMULATIONS FOR TIME DATA CODIFICATION

In this section we describe the problem of optimizing the codification of time data by using models of data rate, which afterwards will be used in a set of Matlab® simulations. These simulations, specially developed for optimizing this coding scheme, will show the average data rate predicted for the Astro instrument, although some aspects of the Spectro will also be taken into account.

It is very important to remark that these simulations are extremely simplistic, because they only take into account an average rate of stars per second that will be processed by Gaia during its lifetime. Therefore, the simulations will not take into account parameters like the following:

- Nominal scanning law (NSL) of the satellite
- Galaxy model
- Limitations of the PDHE, on-board memory, or telemetry
- Rejected or missed detections

Other documents and simulations will take them into account (Lammers 2003), even offering *telemetry curves* –the evolution of the data rate as a function of time. Our simulations only include averaged estimations of some of these factors because their main objective is not to offer detailed and precise results of the telemetry system, but to optimize a set of codification parameters.

7.3.1. Data rate modelization

7.3.1.1. Codification parameters and mission parameters

First of all we have to determine which parameters are to be optimized, and how will they be referred to. We recall that we are focusing on Astro, so only the source packets (SPs) related to it will be treated here. Nevertheless, some Spectro parameter is also included in the generic *data set header* packets, so they will also be included in the calculations:

1. *TdiOffResA* and *TdiOffResS*: Resolution of the TDI offset, for Astro and Spectro, in seconds.
2. *DTTres*: Resolution of the detected transit time, in parts of TDI period. That is, a value of 500 indicates a resolution of $1/500^{th}$ of TDI period.
3. *TSMres*: Resolution of the time slot mark, in parts of TDI period. By definition, it equals to *DTTres*, so both parameters will be referred using a single name (*DTTres*).
4. *MaxTSMoff* (or *MTO*): Maximum offset of a time slot mark, in TDI periods. If converted to seconds, its meaning is, approximately, the inverse of the number of time slots in which a data set is divided.
5. *MaxTSsou*: Maximum number of sources in a time slot (or source packet, which is equivalent).

These are the *codification parameters*. Afterwards, we also have to take into account other *internal parameters* (or *mission parameters*), which may be interesting to include in the models, e.g., for making easier an update after small modifications of the mission:

1. *TdiPerA* and *TdiPerS*: TDI periods (integration time for a pixel) used in Astro and Spectro, in seconds.
2. *AvStarRate*: Average star rate processed by each FOV of the Astro instrument, in stars per second. This is one of the simplifications introduced by this model: the star rate processed by Gaia at a given time will vary in more than 1 order of magnitude, so the use of a single average is not realistic. Anyway, this can be improved by using a *mean* of the star rate (obtained from a probability density function of the stars per second processed by Gaia), instead of the average star density obtained from a Galaxy model.
3. *AvStarSize*: Average star size obtained from the telemetry model (Masana et al. 2004), in bits, excluding the data relative to this optimization (i.e., the detected transit time, time slot marks, etc.). Again, an average value is obtained by weighting the different star sizes (selected as a function of the star brightness) using the probability of every range of brightness, whether using a Galaxy+NSL model or a simple Galaxy model (ESA 2000).

4. *AvCmpRatio*: Average compression ratio predicted after applying the data compression system to the star data. This is, again, an average (and predicted) value, although its effect on the TDC optimization will be negligible –but will offer an approximate data rate value for the Astro science telemetry.
5. *AvDowlink*: Average downlink capability for Gaia, in Mbps (megabits per second).
6. *AstroDlPercent*: Percentage of the downlink reserved for Astro. Obviously the real mission will not reserve a fixed percentage for one or another instrument, but will adapt to the data generation from each instrument; anyway this parameter provides an estimate of the envisaged telemetry occupation by Astro.
7. *TFsize*: Total size of the transfer frame (TF) also including overhead (ESA 1988), in bits.
8. *TFoverhead*: Total size of the overhead in a transfer frame (ESA 1988) in bits, including the Reed-Solomon codes, TF headers and synchronization markers.
9. *PFL*: Probability of Frame Loss (see Pace 1998 and ESA 1988).
10. *MissionDur*: Duration of the mission, in years.

7.3.1.2. Transmission rate of the source packets

Once the parameters to be used have been determined, next we show the modelization of the data rate from Astro and DS headers when using the time data codification proposed in the last section. The first element that we will study is the generation rate of the source packets (which we name *SP rate*). This rate is constant for AP #A (data set headers): 1 per second. On the other hand, AP #B and #C (Astro data) will generate source packets depending on several parameters, such as the maximum TSM offset, the maximum TS sources and, obviously, the star rate:

- One packet per star will be generated if the period between star detections is longer than the maximum TSM offset (*MaxTSMoff*) in seconds, i.e., if the star rate is lower than (or equal to) a *standard time slot rate* defined as:

$$StdTSrate = \frac{1s}{MaxTSMoff(s)} = \frac{1}{TdiPerA \cdot MaxTSMoff} \tag{5.1}$$

- If the star rate is higher than this standard TS rate, but not enough to exceed the maximum sources per time slot (i.e., per packet), then a fixed number of packets per second will be generated. This number will be equal to the standard TS rate.
- For higher star rates, the ratio between the star rate and the maximum sources per packet (*MaxTSsou*) will determine the packet rate. We also must take into account that a single star exceeding *MaxTSsou* will already generate another packet.

All these conditions are summarized in the following 3-segment expression, showing the packets generated by AP #B or #C every second:

$$SP\_rate = \begin{cases} StarRate & (for\ StarRate \le stdTSrate) \\ stdTSrate & (for\ stdTSrate < StarRate \le stdTSrate \cdot MaxTSsou) \\ \left\lceil \frac{StarRate}{MaxTSsou} \right\rceil & (for\ StarRate > stdTSrate \cdot MaxTSsou) \end{cases} \tag{5.2}$$

This formula, illustrated in figure 7.15, shows how the overhead per star introduced by the telemetry system is fixed both for very low and very high star rates (although at very low star

rates this overhead is very important), while in a given interval it is constant and independent of the star rate.

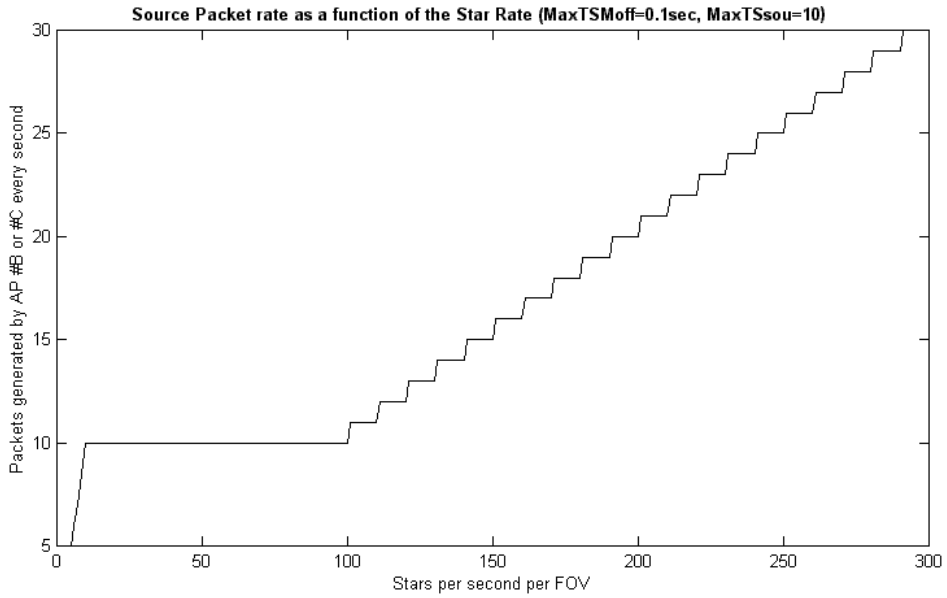


Figure 7.15: Source packets generated by Astro-1 or Astro-2 every second (averaged)

7.3.1.3. Size of the data fields

Some data fields have a fixed and well-known size, such as the primary headers of the source packets (ESA 1988). On the other hand, the size of other data fields depends essentially on the codification parameters, basically on the time resolutions and ranges. The same guidelines used in expression (3.1) are used for obtaining the following field sizes. All the parameters indicated in the following formulae must use the units indicated in section 7.3.1.1.

Although figure 7.11 already defines the TDI Offsets and ASM TDI Flag fields, we prefer to parameterize and include them in the modelization, just for verifying the data rate that we could eventually save:

$$FEROFT\_size = 8 \cdot \left\lceil \left(\left\lceil \log_2 \frac{TdiPerA}{TdiOffResA} \right\rceil + 2 \cdot \left\lceil \log_2 \frac{TdiPerS}{TdiOffResS} \right\rceil + 1 \right) \div 8 \right\rceil \tag{5.3}$$

This formula includes the field sizes (in bits) of the TDI Offsets for Astro, Spectro (MBP+RVS) and the ASM TDI Flag. All together give the size of the indicator of the first effective ROFT, which must be an integer number of octets. Now, continuing with the data set header, we recall the other codification-dependant parameters:

$$RefTime\_size = 40 \text{ bits}$$

$$DSSeqControl\_size = 2 \times (4 \times 14 \text{ bits}) = 112 \text{ bits}$$

Finally we calculate the sizes (in bits) of every data field generated by AP #B and #C, beginning with the time slot mark:

$$TSM\_size = 8 \cdot \left\lceil \left(\left\lceil \log_2 \frac{DTTres}{TdiPerA} \right\rceil + 8 \right) \div 8 \right\rceil \tag{5.4}$$

In this formula we use *DTTres* instead of *TSMres* because of its identical value (and for unifying the parameters used in the several formulae). We also include the 8 bits of the CCSDS P-Field (CCSDS 2002), and we take into account that this field, being the *packet secondary header*, must be an integer number of octets. On the other hand, if CCSDS-compatibility is not

mandatory and the field can be an arbitrary number of bits, the following formula (which implies a smaller field size) can be used:

$$TSM\_noCCSDS\_size = \left\lceil \log_2 \frac{DTTres}{TdiPerA} \right\rceil \tag{5.5}$$

Now we continue with the field that indicates the actual number of sources in the current source packet:

$$NSourcesDS\_size = \lceil \log_2 (MaxTSsou) \rceil \tag{5.6}$$

Next, the detected transit time:

$$DTT\_size = \lceil \log_2 (MaxTSMoff \cdot DTTres) \rceil \tag{5.7}$$

And finally the star data itself (ASM data without DTT, AF data, and BBP data), as a very simple approximation:

$$StarData\_size = \frac{AvStarSize}{AvCmpRatio} \tag{5.8}$$

All the data fields that belong to data set headers are generated once a second. Data fields belonging to science data itself (AP #B and AP #C) will be generated at a *source packet rate* (as shown in expression 5.1), except for DTT and the star data. The latter will be generated obviously at the same star rate, but the DTT of the first star in a source packet is already indicated by the TSM (see section 7.2.1 and figure 7.8). Therefore, the first DTT of a source packet can be omitted, thus implying a transmission rate of:

$$DTT\_rate = StarRate - SP\_rate \tag{5.9}$$

7.3.1.4. Packets received with unrecoverable errors

We are describing a simple model of the data rate transmitted by Gaia, but we should also take into account the packets that are received on ground with unrecoverable errors[23]. These unusable packets, transmitted by Gaia (and therefore occupying downlink) but discarded on ground, represent an extra overhead in the communications channel that can be modeled as:

$$WrongSP\_OH_{(\text{bps})} = NumWrongSP \cdot \frac{SP\_size_{(\text{bits})}}{MissionDur_{(\text{seconds})}} \tag{5.10}$$

This is, the total data size received with errors (*NumWrongSP*×*SP_size*) divided by the total mission time. Now we will obtain each one of these values; first of all, the total number of packets received with errors can be defined as *TotSPnum*×*PPL*, where *TotSPnum* is the total number of packets transmitted during the mission, and *PPL* is the *probability of packet loss* (Pace 1998, ESA 1988):

$$PPL = PFL \cdot \frac{FS + SP\_size}{FS} \tag{5.11}$$

In this expression we use the concept of *PFL* (*probability of frame loss*), the probability that a transfer frame is received with more errors than the correcting capability of the Reed-Solomon system. Engineering teams will determine this value. *FS* is the size of the transfer frame data

[23] Errors that the Reed-Solomon decoder cannot correct, because the number of errors in a transfer frame is larger than the correcting capability of the codes.

field (i.e., the size of the transfer frame without any overhead). The total number of source packets transmitted during the mission (*TotSPnum*) can be obtained from:

$$TotSPnum = AvDownlink_{(\text{bps})} \cdot \frac{MissionDur_{(\text{seconds})}}{SP\_size_{(\text{bits})}} \tag{5.12}$$

Altogether makes possible the simplification of (5.10):

$$WrongSP\_OH_{(\text{bps})} = TotSPnum \cdot PPL \cdot \frac{SP\_size_{(\text{bits})}}{MissionDur_{(\text{seconds})}} \tag{5.13}$$

$$WrongSP\_OH_{(\text{bps})} = AvDownlink_{(\text{bps})} \cdot \frac{MissionDur_{(\text{seconds})}}{SP\_size_{(\text{bits})}} \cdot PPL \cdot \frac{SP\_size_{(\text{bits})}}{MissionDur_{(\text{seconds})}} \tag{5.14}$$

And we can conclude:

$$DataRate\_errors = AvDownlink \cdot PFL \cdot \frac{FS + SP\_size}{FS} \tag{5.15}$$

It is important to note that *AvDownlink* here stands for the predicted downlink occupied only by Astro, instead of the total downlink capacity (so actually its value can be obtained from *AvDownlink*×*AstroDlPercent*). We can see in this formula that *DataRate_errors*, the overhead due to packets with errors, can only be adjusted by the parameter *MaxTSsou*, which determines the relation between the frame size and the packet size. Nevertheless, *MaxTSMoff* can also affect in some way, because shorter values of this parameter imply more packets generated, and therefore, smaller packet sizes. This is the reason why the final results and performances can only be determined by simulations, although these will only reveal approximate results because of their simplistic, averaged nature.

Summarizing, the only codification-dependant parameter that affects the number of wrong packets received is the source packet size, which will be determined by the formulation for AP #B and #C data (AP #A can be negligible because of its much smaller transmission rate). As a first approximation (using average values again):

$$AvSPsize = \frac{AvDataRate}{SP\_rate} \tag{5.16}$$

7.3.1.5. Final formulation of the data rate model

The data rate generated by the instruments and PDHS of Gaia can be divided in three parts:

$$Astro\_dr = dr\_fix + 2 \cdot dr\_var + dr\_err \tag{5.17}$$

where *dr_fix* is the data rate from the data set headers, which does not depend on the measuring conditions (i.e., star rate). On the other hand, *dr_var* is determined by the two Astro instruments, i.e., by AP #B and #C (in our case of study, where only Astro is considered). Finally, *dr_err* is the data rate due to transmission errors, already calculated before.

The calculation of *dr_fix* is straightforward, because packets are generated once a second:

$$dr\_fix = PPH\_size + RefTime\_size + FEROFT\_size + DSSeqControl\_size \tag{5.18}$$

Ancillary data, if required, must be also included. *PPH_size* is the size of the packet primary header. Using the data sizes defined in the previous sections and in ESA (1988):

$$dr\_fix = 48 + 50 + 8 \cdot \left\lceil \left(\left\lceil \log_2 \frac{TdiPerA}{TdiOffResA} \right\rceil + 2 \cdot \left\lceil \log_2 \frac{TdiPerS}{TdiOffResS} \right\rceil + 1 \right) \div 8 \right\rceil + 114 \quad (5.19)$$

Therefore, the resolutions of the TDI offsets are the only codification parameters affecting the fixed data rate. Now we move towards the variable data rate, which depends not only on the field sizes but also on their generation rate. We can distinguish three different types of data: packet headers (generated at *SP_rate*), detected transit times (generated at *StarRate* but excluding the first DTT of every packet), and the star data itself (generated at *StarRate*):

$$\begin{aligned} dr\_var = {} & \left(PPH\_size + TSM\_size + NSourcesDS\_size\right) \cdot SP\_rate + \\ & DTT\_size \cdot \left(StarRate - SP\_rate\right) + \\ & StarData\_size \cdot StarRate \end{aligned} \quad (5.20)$$

We can now include the several values and formulae obtained before (except the *SP_rate* three-segment formula, for the sake of clarity):

$$\begin{aligned} dr\_var = {} & \left(48 + 8 \cdot \left\lceil \left(\left\lceil \log_2 \frac{DTTres}{TdiPerA} \right\rceil + 8 \right) \div 8 \right\rceil + \left\lceil \log_2 \left(MaxTSsou\right) \right\rceil \right) \cdot SP\_rate + \\ & \left\lceil \log_2 \left(MaxTSMoff \cdot DTTres\right) \right\rceil \cdot \left(StarRate - SP\_rate\right) + \\ & \frac{AvStarSize}{AvCmpRatio} \cdot StarRate \end{aligned} \quad (5.21)$$

7.3.1.6. Data rate model for a GDAAS2-like codification

Now that we know how much data will be generated by Astro (including the data set headers and errors), it would be interesting to compare it with the classical codification used in GDAAS-2 (Masana et al. 2004), so that we can explore the performance of our codification. A direct comparison would be misleading –usually exaggerating the benefits of this optimized TDC– because of some big differences: GDAAS-2 transmits a big matrix of TDI offsets (instead of a single value), it uses much smaller resolutions in the detected transit times, the reference time and TDI offset resolutions are also very different… Therefore, we will consider a modified codification that will basically be the same than our optimized TDC, but without using any time slot structures (among other small differences):

- Data is also organized in data sets of 1-second length
- The reference time also uses 40 bits, and the data set sequence control is also included.
- The TDI offsets have the same resolution, and only 1 value per focal plane is transmitted (as in the optimized TDC).
- Source packets are generated taking into account the *MaxTSsou* parameter only; *MaxTSMoff* parameter does not exist anymore.
- Detected transit times are coded wrt the reference time; there are no time slots (case *2* in figure 7.1).

This *GDAAS2-like* codification, which we name *Equivalent GDAAS2*, is the codification that can be comparable in the best way with the *Optimized TDC* (the codification proposed in this chapter). Also, for a richer comparison, we will also consider a *worst GDAAS2* codification, identical to this one but generating one source packet for each and every star (therefore including much more overhead).

The formulae for the *equivalent GDAAS2* codification are identical to those in the *optimized TDC* except for *dr_var*. Moreover, the TSM fields are not included anymore, and the sizes of DTT are now larger and transmitted for every star (even for the first star of the packet):

$$DTT\_size = \left\lceil \log_2 \frac{DTTres}{TdiPerA} \right\rceil \tag{5.22}$$

The source packet rate can be approximated as:

$$SP\_rate = \left\lceil \frac{StarRate}{MaxTSsou} \right\rceil \tag{5.23}$$

Therefore the variable data rate is:

$$dr\_var_{EqGDAAS2} = \left(48 + \left\lceil \log_2 \left(MaxTSsou\right) \right\rceil\right) \cdot SP\_rate + \left(\left\lceil \log_2 \frac{DTTres}{TdiPerA} \right\rceil + \frac{AvStarSize}{AvCmpRatio} \right) \cdot StarRate \tag{5.24}$$

On the other hand, the formulae for the *worst GDAAS2* codification also modify *dr_fix*. The reason is that now we do not need to indicate the number of sources per packet (we now transmit 1 source per packet), but it would be interesting to indicate the total number of sources in the data set. As in Masana et al. (2004), 16 bits should be enough:

$$dr\_fix_{WorstGDAAS2} = dr\_fix_{OptTDC} + 16 \tag{5.25}$$

Finally, *dr_var* can be expressed as:

$$dr\_var_{WorstGDAAS2} = \left(48 + \left\lceil \log_2 \frac{DTTres}{TdiPerA} \right\rceil + \frac{AvStarSize}{AvCmpRatio} \right) \cdot StarRate \tag{5.26}$$

7.3.2. Assumed parameters and predicted optimal values

In section 7.3.1 we have modeled the effects of the main parameters of the data codification system. For example, *MaxTSMoff* (or *MTO*) determines the size of the DTT field, which is the most transmitted field (once per star) and, therefore, the most important field to optimize. *DTTres* also determines its size, although the choices for selecting one or another value are almost fixed *a priori*, because of the resolution requirements. *MTO* also affects in the number of packets generated per second (*StdTSrate*), although we do not necessarily have to generate a packet if no source has been detected (as seen in figure 7.14, there can be "holes" in the timescale). *MaxTSsou*, on the other hand, determines both the minimum overhead and the probability of packet loss due to transmission errors, although *AvStarSize* and *CmpRatio* will also affect to this. Respecting *AvStarSize* we have used the telemetry model proposed in Masana et al. (2004) and summarized in chapter 6 for determining the several fields and the corresponding sizes, while U. Lammers gently provided the data needed to obtain the probabilities of measuring bright, medium or faint stars.

Summarizing, here we list the several parameters introduced in section 7.3.1.1 with their corresponding values, whether they are predicted values or valid ranges (codification parameters), or assumed values (mission parameters):

1. *TdiOffResA*: The minimum requirement is 16ns, although it is recommended to use a resolution better than 10ns. A resolution better than 1ns seems completely unnecessary.
2. *TdiOffResS* requirements are not still clear, although it should be better than 350ns if compared to the case of Astro. Spectro, in principle, does not need as much timing precision as Astro, so a resolution better than 50ns seems not necessary.
3. *DTTres*: It should be better than 200 parts of TDI (i.e., 1/200th TDI). More than 500 parts of TDI seems unnecessary. *TSMres* has this same requirement.

4. *MaxTSMoff* (or *MTO*): Its absolute maximum is 1358 TDI periods (equivalent to 1 second, in Astro), and values lower than 5 TDI periods will surely introduce too much overhead. About 100 TDI periods seems a reasonable value.
5. *MaxTSsou*: Transmitting only 1 source per time slot (i.e., per source packet) may introduce an overhead of more than 5%, so at least 2 or 3 sources should be included in every packet. More than 20 sources per packet may lead to unacceptable packet losses (see expression 5.11).
6. *TdiPerA*: 736.6μs (Jordi et al. 2003, also summarized in the first annex).
7. *TdiPerS*: 16.37ms (Jordi et al. 2003).
8. *AvStarRate*: Although some studies using the Galaxy model indicate an average value of about 300 stars/s (per FOV), the inclusion of the NSL leads to a mean value of about 174 stars/s per FOV (as U. Lammers verified with simulations).
9. *AvStarSize* can be obtained combining the field sizes from Masana et al. (2004) with the probabilities of measuring a bright, a medium or a faint star, indicated by U. Lammers. It is important to note that only the relevant data fields are included here, this is, excluding the source identifier (which is only for GDAAS), detected transit time, instrument identifier, etc. This is a summary of the data fields and sizes, and their associated rules for being included in the budget:
   - Window mode (2 bits), for every star
   - Normal or narrow AF11 (1 bit), for every star
   - Normal or narrow BBP (1 bit), for every star
   - Detection quality (7 bits), for every star
   - Multiplicity/shape flags (23 bits), for every star
   - ASM detection, AC coordinates (20 bits), for every star
   - WYxx window offset (7 bits), only bright stars
   - ASM window:
     - 48×16=768 bits (bright and medium bright stars)
     - 25×16=400 bits (faint stars)
   - Total flux (23 bits), for every star
   - Astro1 and Astro2 background values (7+7=14 bits), for every star
   - AF and BBP readout coordinates (16×(17+4)=336 bits), for every star, although its transmission is being studied (if it is finally mandatory, a better codification can be used)
   - AF1-10 windows:
     - 96×10×16=15360 bits (bright stars)
     - 12×10×16=1920 bits (medium bright stars)
     - 6×10×16=960 bits (faint stars)
   - AF11 window (without considering the narrow AF11 windows):
     - 96×16=1536 bits (bright stars)
     - 40×16=640 bits (medium bright stars)
     - 26×16=416 bits (faint stars)
   - BBP windows (without considering the narrow BBP windows):

- 96×5×16=7680 bits (bright stars)
- 16×5×16=1280 bits (medium bright stars)
- 10×5×16=800 bits (faint stars)

If we assume that the scanning probability for bright stars is 0.26%, for medium bright stars is 7.66%, and for faint stars is 92.08%, the average size for all the measured stars is:

$$\begin{aligned} AvStarSize = {} & (2+1+1+7+23+20+23+14+336)\cdot 1 + \\ & (7+768+15360+1536+7680)\cdot 0.0026 + \\ & (768+1920+640+1280)\cdot 0.0766 + \\ & (400+960+416+800)\cdot 0.9208 \end{aligned}$$

which gives a mean star size of about 3218 bits. The worst case (bright stars) gives about 25778 bits per star, while the best case (faint stars) gives about 3003 bits per star.

10. *AvCmpRatio*: Predicted values range from a very conservative ratio of 2, up to an optimistic ratio of 5. Here we will assume a compression ratio of 3, which has been verified using some preliminary compression systems (Portell et al. 2001b, 2002b).
11. *AvDowlink*: The current baseline is about 1.2Mbps, although some studies assume an even higher average.
12. *AstroDlPercent*: As a first approximation we assume that Astro will use 50% of the downlink.
13. *TFsize*: 10232 bits including overhead (ESA 1988)
14. *TFoverhead*: 1360 (ESA 1988)
15. *PFL*: 10^{-5}, although some preliminary studies like Pace (1998) indicate a better quality of the channel
16. *MissionDur*: 5.0 years

7.3.3. Simulations

Although most of the software for Gaia is being developed using Java, C or Fortran, these simulations are devoted only to optimize the data codification and not to exchange data with other simulators. Therefore, Matlab® (by *The MathWorks, Inc.*) was our choice for developing these optimizations because of its large amount of tools, libraries and graphical utilities available. Our simulations have been tested on versions 5.3 (R11.1) and 6.5 (R13) for MS-Windows. Although some tests have been done for obtaining standalone executable files, the current version of this software needs to be run in a Matlab® environment.

The several Matlab® functions (*.m* files) that have been developed for these simulations are listed and briefly explained in Annex C. Here we will only show some hints on the main programs (or functions) and their main results. Although they are GUI-based and rather intuitive, we also include their user manuals in Annex C.

The main program for testing all the parameters of the data codification is `gaia_otdc_main.m`. After writing its name in the Matlab® environment and pressing the <*Enter*> key, we obtain the screen shown in figure 7.16. In this screen we have included all the relevant parameters affecting the data rate generated by the Astro instrument of Gaia. Also, both graphical and numerical results are included, showing the total data rate (without Spectro) and the data rate saving wrt a GDAAS2-based codification. All the models described in section 7.3.1 are used for offering these results, as well as the several parameter values listed in section 7.3.2. Although we can manually change the several codification parameters (placed in the top left corner of the screen), some functions have also been developed for making easier the optimization of their values. Figure 7.17 shows two of these sub-programs, for optimizing the TDI offset resolutions (left panel) and the maximum TSM offset (right panel).

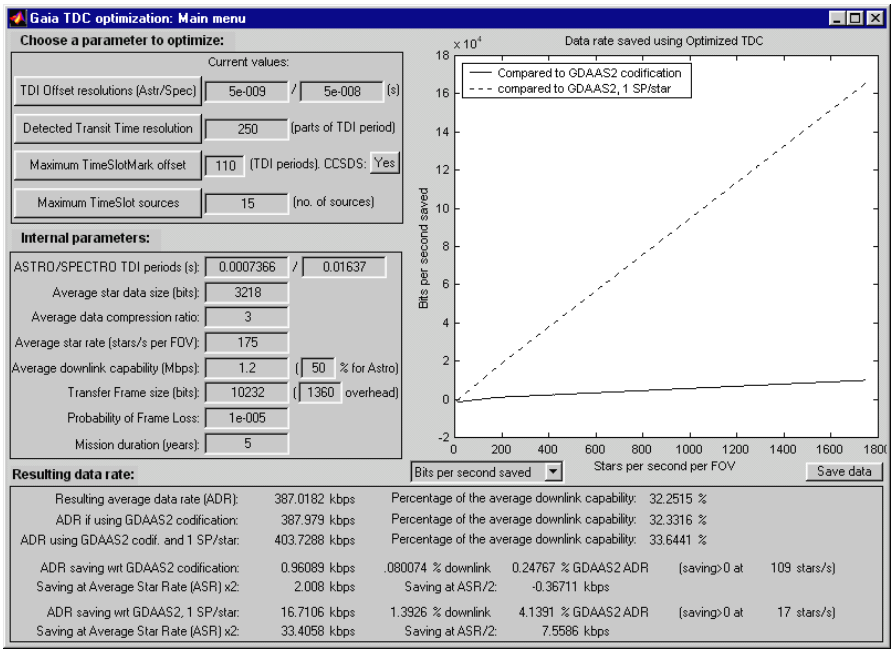


Figure 7.16: Main menu of gaia_otdc_main.m

Using these dialogs as explained in their user manuals (cf. Annex C) for testing several values, we can obtain a set of codification parameters totally optimized for the environmental conditions fixed by the mission parameters. In the case of the TDI offsets, a change in their values (within the recommended ranges) does not affect much the final result: we can only save up to 16 bits per second (since they are transmitted only once a second). Therefore, we can assign whichever values to them. We think that 5ns for Astro and 50ns for Spectro shall be enough (taking into account the recommendations given by U. Bastian).

The effect of resolution of the detected transit time (*DTTres*) on the final telemetry consumption is a little higher (about 660 bps), although when compared to GDAAS2 its effect is negligible. Even though, when comparing to an equivalent GDAAS2-like codification, we save more bits with higher resolutions –this is where the time slots structure gets more useful. Therefore, we will assign a somehow arbitrary value, always trying to guarantee a minimal quantization error on the detected transit time but with acceptable telemetry consumption: 1/500th TDIs.

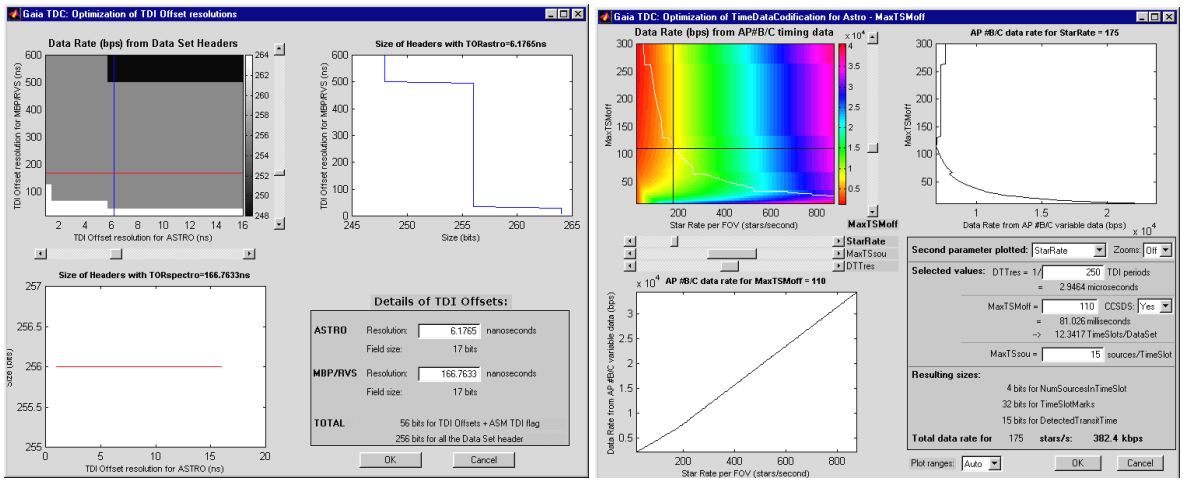


Figure 7.17: GUI routines for optimizing some codification parameters

The case of *MaxTSMoff* is more complicated and must be treated by taking an accurate look to the simulation graphs. Furthermore, it has a tight interrelation with *MaxTSsou* and the star rate. This is why we prefer to leave this parameter for the end of the section, turning now our

attention to *MaxTSsou*. As explained before, this parameter affects both the packet overhead and the probability of packet loss (PPL). Too low values imply a higher overhead, and too high values lead to unacceptable PPL values. Figure 7.18 shows the data rate wrt *MaxTSsou* and the star rate, where we can see an interesting low consumption for 15 or more sources per packet.

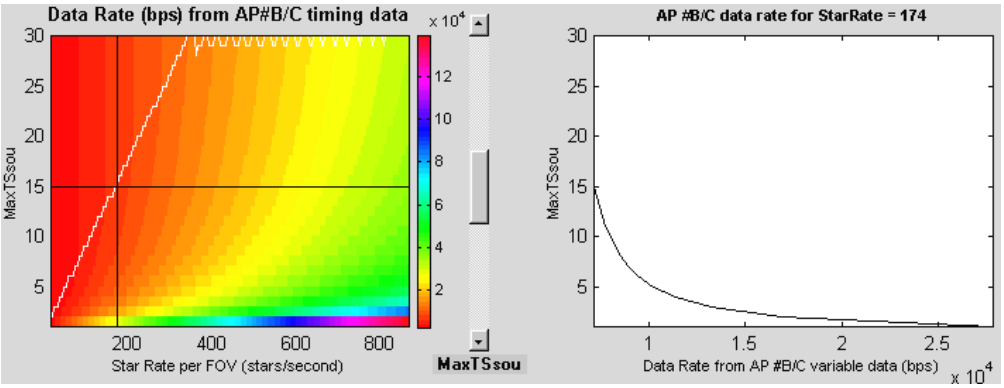


Figure 7.18: Data rate as a function of 'MaxTSsou', for an average star data size

On the other hand, we must take into account that packet sizes (in bits) will not be uniform, but will depend on the brightness (i.e., sampling scheme) of each star, as well as on the actual compression ratio achieved. Figure 7.19 illustrates a zoom of the data rate consumption wrt *MaxTSsou*, for the worst case in the left panel (star size = 25778 bits, compression ratio = 1) and for the best case in the right panel (star size = 3003 bits, compression ratio = 5). Here we can see how transmitting 15 stars in every packet is the best choice.

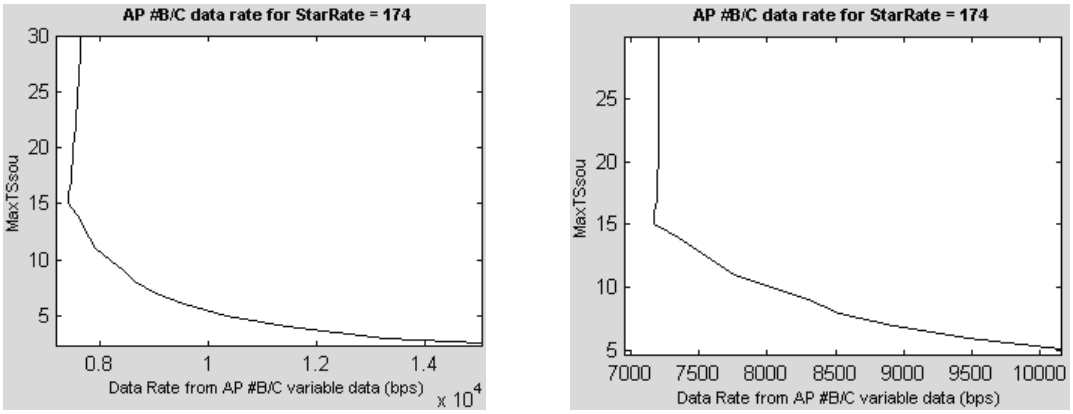


Figure 7.19: Data rate wrt 'MaxTSsou', for the worst and best cases of star data size

Now we turn back our attention to *MaxTSMoff*. Figure 7.20 shows how this parameter has a well-defined set of optimal values (white line in the color plot), which depends on the star rate. The right panel of this figure shows its relation wrt *MaxTSsou*. Their optimal values (in white) have an almost-linear relation, which is completely logical: the longer the *MaxTSMoff*, the more the sources that should be able to fit in a source packet. And vice versa: the more the sources that can be transmitted in a single packet, the longer the *MaxTSMoff* should be (for making possible the inclusion of more sources in the packet). These complex interrelations recommend us taking the best value for our average situation, which is 174 stars per second per FOV. Therefore, the maximum TSM offset shall be 116 TDI periods (about 8.9 ms, a 112th part of a second). Using all of these parameter values we can obtain the final results, offered by the main program when exiting:

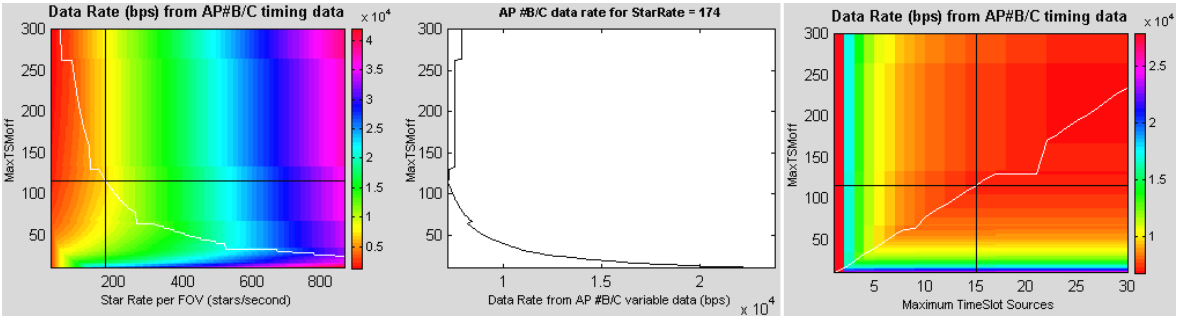


Figure 7.20: Data rate consumption wrt 'MaxTSMoff', wrt star rate and wrt 'MaxTSsou'

```
<*========== Optimization results: ==========*>

Bit lengths for each data field:

TDI Offset for Astro: 18 bits. TDI Offset for MBP/RVS: 19 bits.
   -> TDI Offset resolutions: 5 ns in Astro, 50 ns in Spectro.
Time Slot Mark (TSM): 32 bits, including the 8-bit P-Field for CCSDS compatibility.
   -> Maximum TSM Offset: 116 TDI Periods (85.4456 ms).
   -> Standard rate of Time Slots (of a given AP) per Data Set: 12
Packet length (number of sources in the packet): 4 bits.
Detected Transit Time (DTT): 16 bits
   -> Coding resolution: 1/500 TDI Periods (1.4732 us).
Within a given AP, a new Time Slot will have to be started (i.e. a new Source Packet will
have to be generated) after 15 sources, or when reaching the Maximum TSM Offset, or when
reaching a new Data Set.

Overheads and error rates:

Source Packet overhead (average): 0.30771%
Transfer Frame overhead: 13.2916%
   -> Total overhead (average): 13.5993%
Total amount of data received during the mission:
      Useful, error-less data: 6.8294 TBytes   (in 3.6957e+9 Source Packets)
      Data with unrecoverable errors: 319.6197 MBytes, a 0.0044631% of total data received
      (164.9489e+3 Source Packets)
      Probability of Packet Loss: 2.8321e-005
Average size of an Astro Source Packet: 16254.5151 bits
```

The results of the overhead and transmission errors are satisfactory, as well as the data rate saving (in average) obtained with this codification: 1048 bps with respect to an equivalent GDAAS2-like codification. This result is extremely interesting: we are saving some downlink despite of the better-quality data we are obtaining with this codification, such as better transmission reliability and CCSDS compatibility in some time data fields. Furthermore, we must take into account that this saving is compared to an *equivalent* GDAAS2 codification: compared to the GDAAS2 codification "as is" (Masana et al. 2004) we would save much more data and, furthermore, obtaining much higher time resolutions[24]. Compared to a codification that includes only one star per source packet, we save more than 16.5 kbps –only with a codification scheme, i.e., without any data compression yet.

Nevertheless, we must recall that Gaia will scan sky fields with very large density variations, from a few stars per second up some 1500 stars/second (or even more). We have also seen that some codification parameters offer optimal results only for very narrow ranges of star rate. Therefore, we cannot rely on static, single-averaged results (centered at 174 stars/second), but we should also take care of what happens with different sky densities. Unfortunately, our codification does not seem to operate optimally in this sense: as seen in figure 7.21 (and as reported by the numerical output of the software), below 108 stars/second we are occupying more downlink than using a GDAAS2 codification. For example, at 87 stars/second (per FOV)

[24] For example, GDAAS2 uses a resolution for DTT (detected transit time) of only 1/25 TDI, 20 times worse than the resolution that we are using here.

we are losing 272 bps. On the other hand, at 348 stars/second we save more than 2 kbps. Therefore, some improvement should be done for obtaining the highest data rate saving at any sky density.

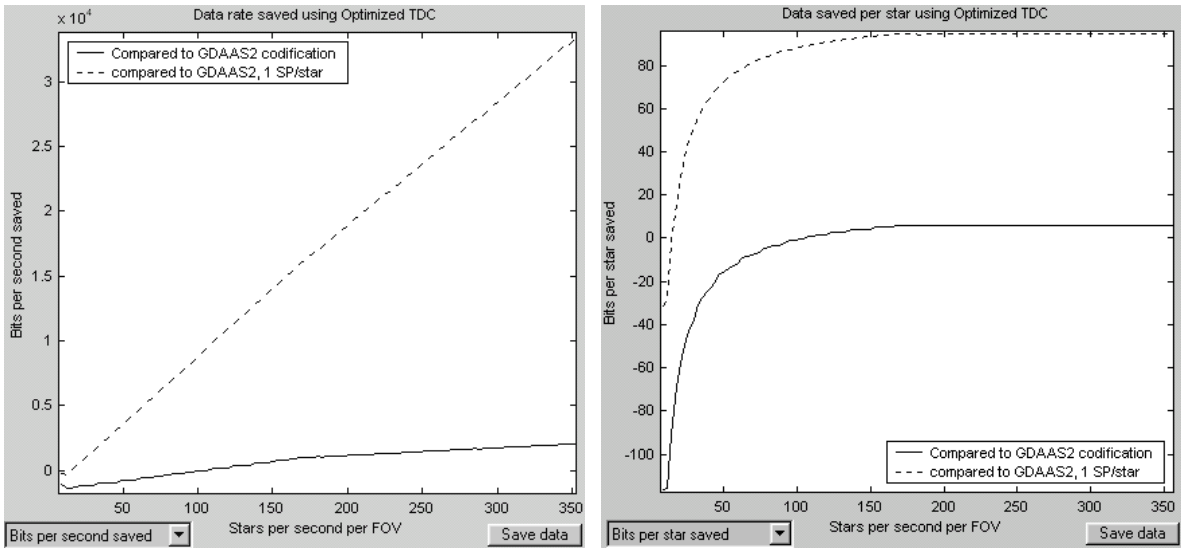


Figure 7.21: Data rate saving and bits per star saving with the static codification parameters

7.3.4. Improving the current scheme: Adaptive TDC

In the color plot of figure 7.20 (left panel) we see a clear relation between the optimal value of *MaxTSMoff* (*MTO*) and the star rate per FOV. We have aimed for taking full advantage of this, which has led to the development of another optimization software (also based on Matlab®). The idea behind this software is to use an *adaptive codification system*, using one or another value of *MTO* depending on the star rate. Its implementation may be based on the use of the FDI (field density index), already introduced in chapter 4. This index can be as simple as a counter of the stars detected every second. Then, an Adaptive TDC (time data codification) would take this value and use one or another *MTO*, offering always the best results.

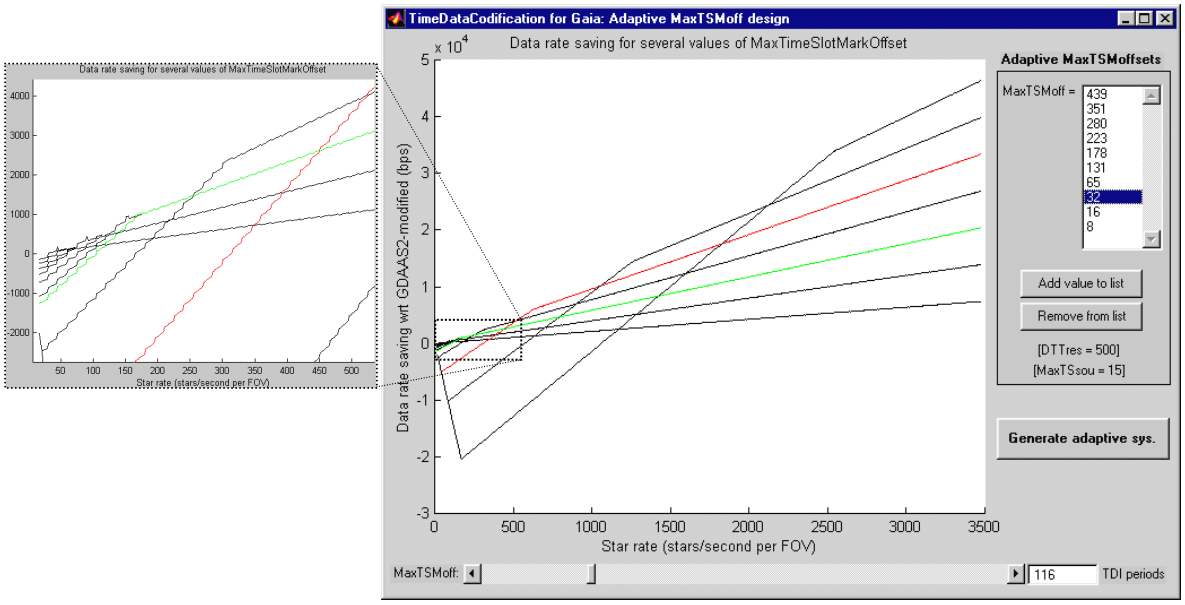


Figure 7.22: Main dialog of gaia_otdc_adap.m with low star rates zoomed

The program developed for testing this, `gaia_otdc_adap.m`, shows the evolution of the data rate saving as a function of the star rate, and for several values of *MTO*. Figure 7.22 shows its main dialog box. We can see there that very low *MTO* values are optimal for very high star

rates, leading to data rate savings of up to some tenths of kbps. On the other hand, these very low *MTO*s imply very short packets, so more packets must be generated and, hence, more redundancy is introduced. This penalization is huge at very low star rates (up to 20 kbps), so at those low rates the value of *MTO* must be much higher –thus implying a low saving per star but much less redundancy.

The several curves plotted in the figure correspond to the *MTO* values listed in the top right of the window. The user can add or remove values from this list, trying to obtain a minimal set of curves with the maximum saving achievable. The plot can also be zoomed, as seen in the composition of figure 7.22, making possible a better determination of *MTO* value for low star rates. When finished, we click on the "Generate adaptive system" button and the program exits, doing some calculations for offering the final adaptive system. A set of plots and some numerical results are also offered.

The numerical results obtained for our case lead to an adaptive system using 7 different *MTO* values (ranging from 8 to 524 TDI periods) with their associate star rate applicability. One of the most interesting issues in this software is that a *star rate histogram* (gently provided by U. Lammers) is used for offering more realistic telemetry estimations. We should not forget that all of these simulations use very simplistic and averaged approximations, but in this case we get much closer to realistic results. Figure 7.23 shows this histogram, where we can see a peak close to 55 stars/second per FOV, but a long trail up to 1380 stars/second. As already said before, U. Lammers reported us a mean value of 174 stars per second.

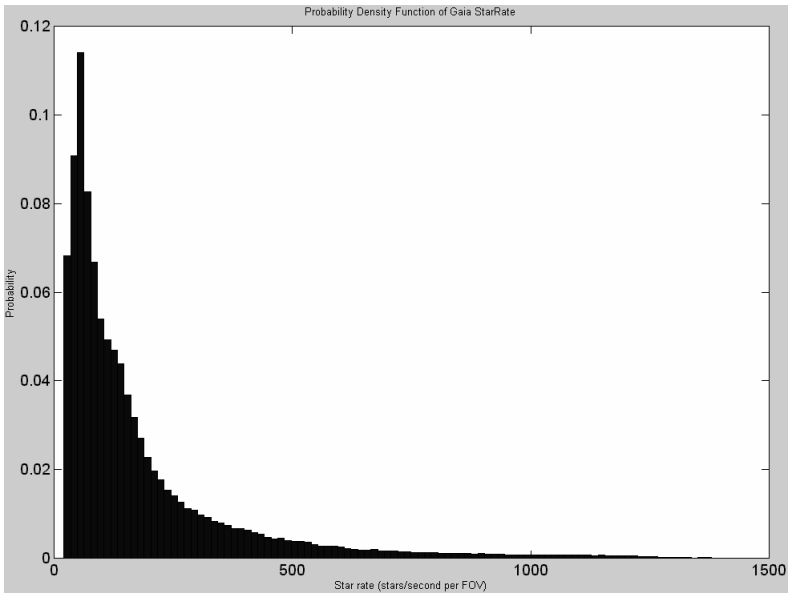


Figure 7.23: Star Rate histogram for Gaia

When applying this histogram to the data rate formulae and, therefore, obtaining more realistic data rate savings, we find that a static system (at *MTO*=116 TDIs) actually saves only 525 bits per second, instead of the predicted (averaged) 1048 bits per second. On the other hand, the use of an adaptive coding leads to a total telemetry saving of 1215 bps –here we can see the power of this system. With the composition shown in figure 7.24 we illustrate how this is possible: the adaptive system follows the envelope of the several curves, so that we are always working at the optimal regime.

7.3.5. Final optimal values of the parameters

Here we summarize all the codification parameters and methods recommended for coding the time data (and transmitting the science data in general) for Gaia, as well as the results obtained with this system. First of all, the values for the codification parameters are the following:

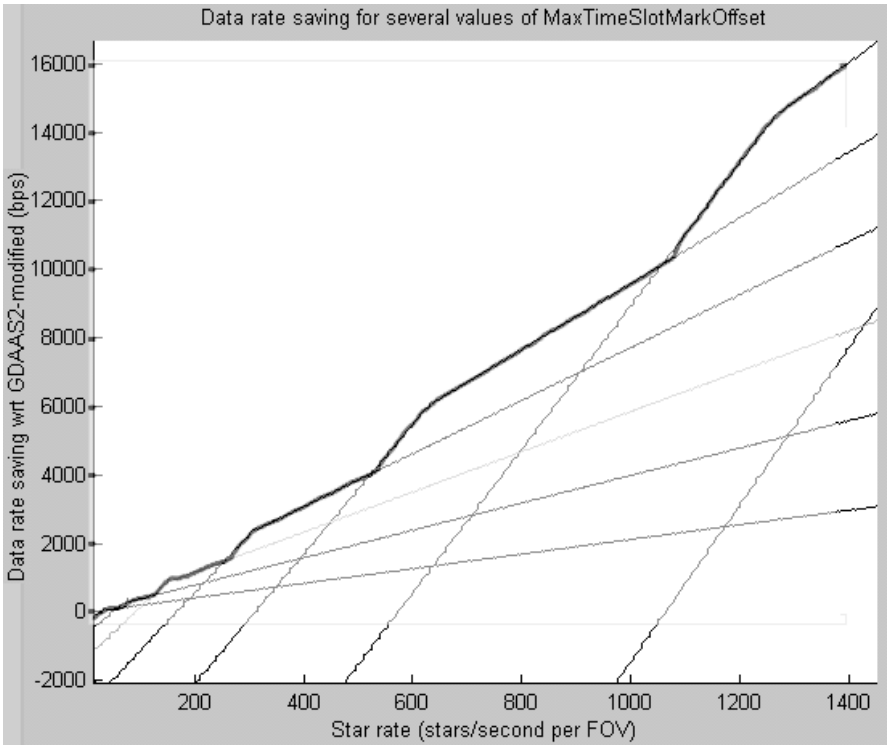


Figure 7.24: Optimal envelope obtained using the Adaptive TDC system

- TDI Offset (Astro): 5 ns (18 bits)
- TDI Offset (Spectro): 50 ns (19 bits)
- Detected Transit Time (and Time Slot Mark) resolution: 1/500th TDI period (1.4732 μs). Size of Time Slot Mark: 32 bits.
- Maximum Time Slot Sources: 15 sources per time slot (i.e., per source packet). Size of the *NumSourcesTS* field: 4 bits.
- Maximum Time Slot Mark Offset[25]: adaptive, depending on the star rate:
  - 524 TDI periods (1/2 seconds) for <61 stars/s. Size of DTT: 18 bits
  - 262 TDI periods (1/5 seconds) for 61 – 120 stars/s. Size of DTT: 17 bits
  - 131 TDI periods (1/10 seconds) for 120 – 256 stars/s. Size of DTT: 16 bits
  - 65 TDI periods (1/21 seconds) for 256 – 526 stars/s. Size of DTT: 15 bits
  - 32 TDI periods (1/42 seconds) for 526 – 1066 stars/s. Size of DTT: 14 bits
  - 16 TDI periods (1/85 seconds) for 1066 – 2145 stars/s. Size of DTT: 13 bits
  - 8 TDI periods (1/170 seconds) for >2145 stars/s. Size of DTT: 12 bits

  This selectable *MTO* value can be indicated at the beginning of every data set with a simple flag of 3 bits.

The results obtained with this system are the following:

- Source packet overhead (average / min / max): 0.3151% / 0.0139% / 0.6159%
- Transfer frame overhead: 13.2916%

[25] MTO values indicated in seconds are roughly approximate (decimals are omitted)

- Total uncompressed data received during the mission (approx.):
  - 20.2 TB with no errors, in 3800 million source packets
  - 710.2 MB unusable, in 166420 source packets

  The total size of the data received is not far from the predictions available for Gaia, which indicate an average of 82 observations per star (Jordi et al. 2003): assuming the maximum of 15 stars per packet, these simulations report 57 billion star measurements, which correspond to about 57 observations per star (assuming 1 billion stars measured by Gaia).
- Probability of Packet Loss: $<3\cdot10^{-5}$ (typical)
- Average size of an Astro source packet (average / min / max): 15.9 kb / 7.8 kb / 387 kb (hence 64 KB boundary is not exceeded)

The average, minimum and maximum calculations have been obtained using `gaia_otdc_main.m` and assigning several values to the parameters: mean star rate and star data size for average results (and using *MTO* = 131), maximum star data size without compression and a star rate of 60 stars/s (using *MTO* = 524) for smallest sizes and maximum overheads, and minimum star data size with compression 5 and a star rate of 2145 stars/s (using *MTO* = 8) for largest sizes and minimum overheads.

Using the *Adaptive TDC* system we save bandwidth for star rates higher than 31 stars/second (while the static system needs 108 stars/second), and the penalization is lower than 200 bps even at 14 stars/second. In this way, we save about 1.2 kbps in average when compared to a GDAAS2-like codification, leading to an average data rate of 398 kbps (assuming a data compression ratio of 3), or 1.18 Mbps without compression. If we include the Transfer Frames overhead, the final data rate is about 451 kbps (38% of the average downlink), or 1.34 Mbps without compression. If we do not need CCSDS-compatibility for the time slot marks (TSM), we would save about 309 bps more, which is not really worth it so this option will not be considered here.

7.4. RECOMMENDED TIMING AND TRANSMISSION SCHEMES

In the previous section we have obtained and tested a completely optimized codification scheme, offering the lowest telemetry occupation for any star density. Here we complete the preliminary packet structures shown in figures 7.11 and 7.12 with the improved scheme and the final values, also showing a possible implementation of a telemetry coder fulfilling this recommendation.

7.4.1. Summary

Section 7.3.5 lists the several parameter values to be used in this codification, as well as the time resolutions obtained with them. In order to use an adaptive codification system as proposed in the previous section we need to indicate in someway the selectable value currently being used. We recommend to update the MTO value from one data set to the next, thus avoiding rapid changes in this parameter (which would be extremely difficult to signal) but still adapting to changes in the star densities being measured. Section 7.4.2 will explain this how can be implemented, but for the moment we need to include a signaling flag in the data set header (*MTO flag*), with the signification indicated in table 7.2. Here we can see how this flag also indicates if the data set contains no data, i.e., if no Astro source packets are generated until the transmission of the next data set header. This is required for avoiding problems with the sequence control system. This way, the *current* SSC values transmitted in the data set header will not be taken into account.

***MTO Flag* value**	**Maximum TSM offset**	**Size of DTT fields**
"000"	8 TDIs	12 bits
"001"	16 TDIs	13 bits
"010"	32 TDIs	14 bits
"011"	65 TDIs	15 bits
"100"	131 TDIs	16 bits
"101"	262 TDIs	17 bits
"110"	524 TDIs	18 bits
"111"	*<void data set>*	*<void data set>*

Table 7.2: Values of the MTO flag

The final recommended structure for the AP #A source packets (data set headers) is shown in figure 7.25. We can see that the total packet size has not changed (although we have introduced a new flag), implying a telemetry consumption of only 264 bps. There is an empty space of 1 bit available, which may be used in the future if necessary –e.g., for transmitting the second ASM TDI flag, if it is finally necessary to transmit one per instrument. Also, the field for ancillary data keeps reserved for eventual transmission of the full matrix of TDI periods (or offsets), or any other requirement that may be discovered in the future.

It is important to note that an error in the reception of the *MTO* flag would make almost impossible the recovery of AP #B and #C data. In order to avoid this, we will use a similar mechanism to that one used for the sequence control: every data set header will not only transmit the *MTO* value used in the current data set, but also the value used in the previous data set.

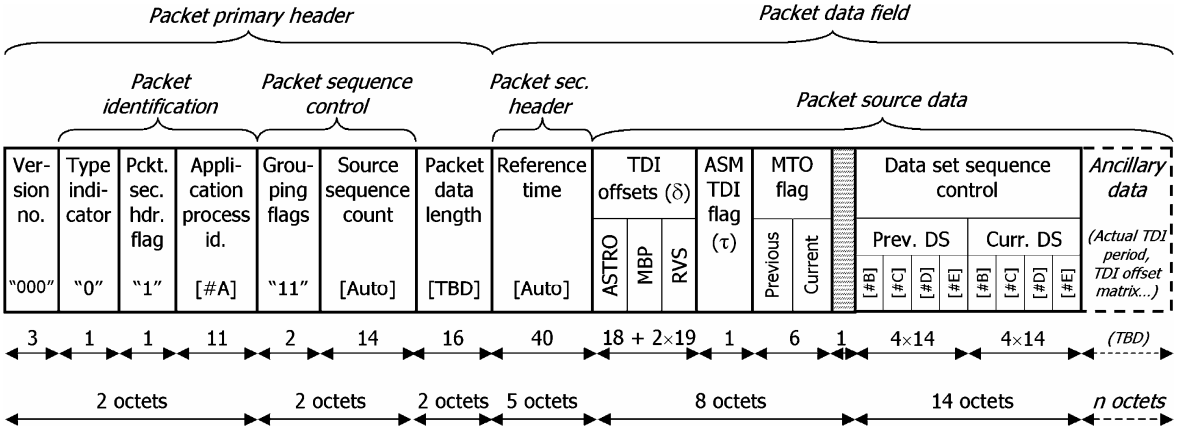


Figure 7.25: Final packet structure and contents for AP #A (data set header)

The science data generated by Astro will be transmitted as AP #B (Astro-1) or #C (Astro-2) source packets. Although the preliminary structure shown in figure 7.12 is still valid, we have decided to slightly modify it in the way illustrated in figure 7.26: since the TSM is equivalent to the first DTT of the packet, it seems more logical to send all the times contiguously –except for the *NumTSsources* field, required for knowing how many DTTs must be read before reaching the star data. In this way, we separate these time data fields from the rest of data, which will be compressed afterwards. The receiver will wait for *NumTSsources*-1 DTT fields, the size of which will be determined by the *MTO* flag of the current data set. In any case, the number of sources transmitted in a source packet will not exceed 15 –so the maximum number of DTT fields transmitted is 14. It leads to a maximum length of the *Detected transit times* field of 14×18 = 252 bits.

Every AP #B/C packet (Astro packet) will contain data from, at least, one source. If no source is detected, no packet must be generated. On the other hand, no packet from AP #A (data set header) can be omitted; if no source is detected during a whole data set (thus leading to a *void data set*), it will be indicated by assigning "111" to the *Current MTO Flag*.

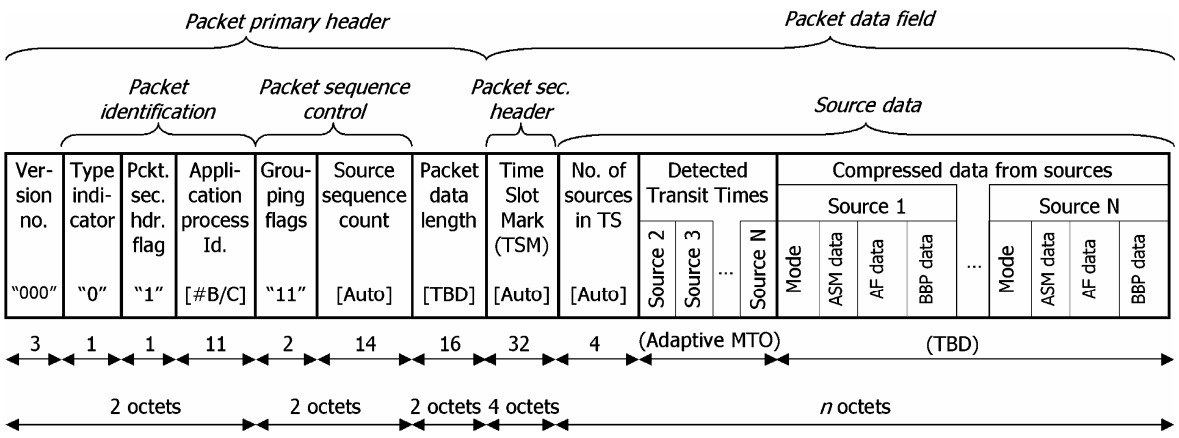


Figure 7.26: Final packet structure and contents for AP #B/C (Astro data)

Recent telemetry models developed for GDAAS-2 show that the window readout times of every AF and BBP should also be transmitted. This issue must be thoroughly studied, but although the reasons for including them in GDAAS-2 are only simplicity and execution speed of the simulator, its heritage in a flight-realistic telemetry model can happen. Therefore, an alternative structure of Astro packets should be developed for optimizing the transmission of these readout times to the maximum. A differential coding seems a good solution, since the variation from one readout time to the next will surely be small. Here we propose this preliminary scheme:

- A nominal *Readout time increment* will be transmitted, in an absolute coding indicating the number of TDI periods to add from one CCD to the next. Two parameters may be transmitted if necessary (one for wide CCDs, another for narrow CCDs). This nominal *time increment* will be calculated in the PDHS for every star. With a theoretical increment of 4900 TDIs, a range of 4400-5400 TDIs seems enough, which implies 10 bits.
- 16 *readout time modifications* will be transmitted, one per CCD (AF and BBP), in a differential coding. Each one will indicate the difference (in TDI periods), wrt the nominal increment transmitted, for reading every AF and BBP window. A range of ±30 TDIs seems enough, which implies 6 bits.

All in all, this preliminary coding proposal would imply only $10 + 6 \times 16 = 106$ bits, in front of the 272 bits assumed in Masana et al. (2004). Nevertheless, the study on data compression described in next chapters will be able to improve these values still much more.

7.4.2. Communication and codification layers

Figure 3.1 of ESA (1988) illustrates the end-to-end operation of an ESA packet telemetry system, indicating the several data structures implied: source packets, telemetry packets and transfer frames. In this chapter we have designed how the PDHS must fill the several types of source packets. Although this design uses the terms from the CCSDS packet telemetry standard (CCSDS 2000), its application in ESA packet telemetry standard (ESA 1988) is almost immediate. With the definition of source packets offered in this chapter we already offer a full telemetry definition, since the rest of data structures will be fixed by engineering teams (TBC). Anyway, the basic guidelines for a full integration in this ESA telemetry standard are the following:

- In section 7.3.5 we have verified that the maximum size of a source packet is 387 kb (i.e., 48 KB), so the 64 KB boundary imposed by the source packets is not exceeded. It means that no segmentation is necessary, and therefore, the telemetry packets exactly equal to the source packets defined here.
- This work focuses on the Astro instrument. Spectro may generate larger packets, so the affirmation of the previous point should be verified for that instrument when possible.

- Transfer frames will be generated as defined in section 5 of ESA (1988). No special treatment must be applied, this is:
  - No secondary header is necessary (TBC)
  - No transfer frame trailer is necessary (TBC)
  - The proposed multiplexing in virtual channels is illustrated in figure 7.9.
  - Standard packets are synchronously inserted, i.e., the octet boundaries of the source packets coincide with the octet boundaries of the transfer frames.
  - The packets are inserted in the transfer frames in a *forward* order.

The need of using some extraordinary transmission scheme, such as the secondary header extension for guaranteeing the order of the transfer frames, should be studied in detail by engineering teams. Figure 7.27 illustrates the scheme with the several communication layers implied in this recommendation, from the observation instruments to the transmitting antenna, and their corresponding layers at reception.

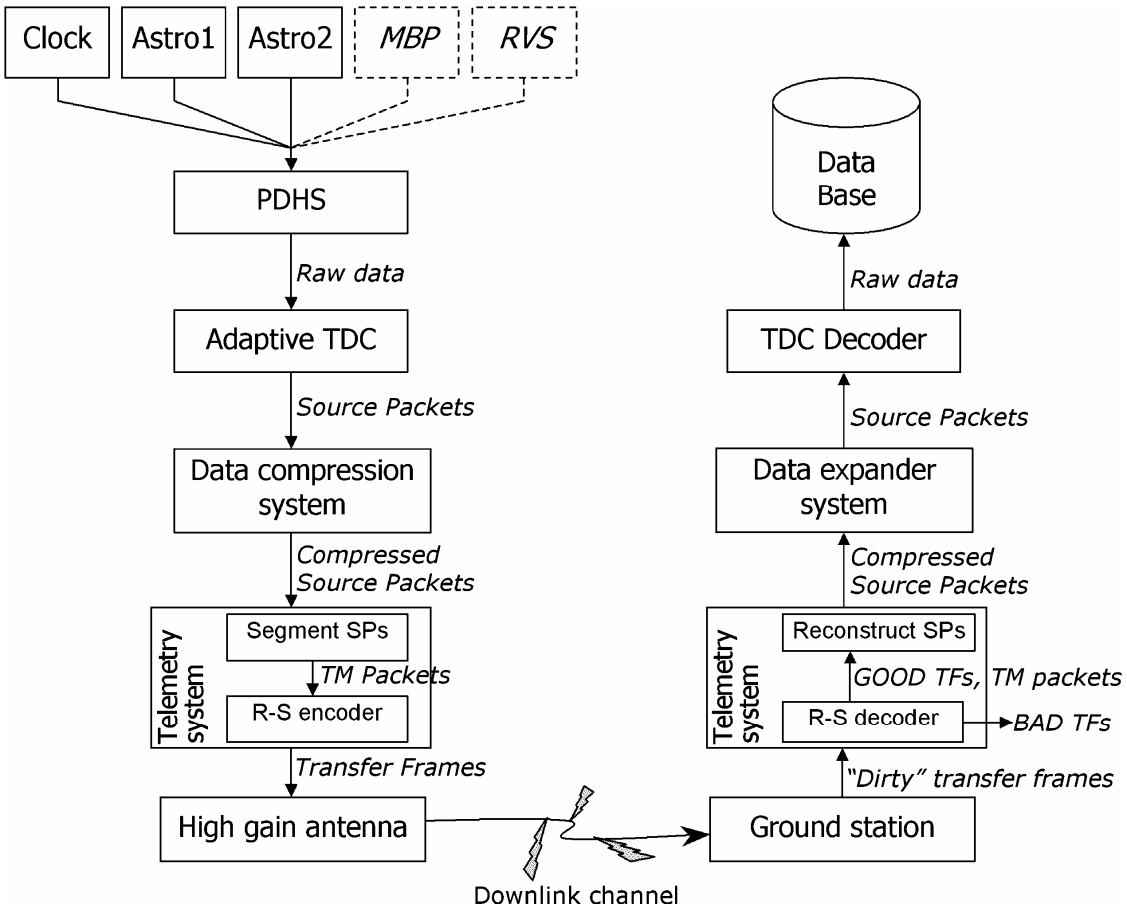


Figure 7.27: Data communication layers in Gaia

7.4.3. Implementation guidelines for a telemetry coder

We have proposed a codification system that is not trivial, so an accurate description of a possible implementation is more than interesting. Here we will describe a method for generating the source packets of section 7.4.1. First of all, let us define a set of counters and parameters that will be needed during the codification process. Table 7.3 lists them, with their associated operation rules and signification. This table will be useful both for understanding the flux diagram of figure 7.28 and for guiding an eventual implementation (software or hardware) of this codification system.

Parameter	Long name	Meaning	Reset	Increment rule
RefTime	Reference time	Seconds since a given epoch	J2010.0	Every second
TdiOff	TDI offset	Time since the last RefTime, in *TdiOffRes* units	At every change of RefTime	Every *TdiOffRes* seconds
TDI_cs	TDI clock strokes	Number of TDI clock strokes since FEROFT	At FEROFT of every data set	Every TDI clock stroke
TSM	Time Slot Mark	TDI clock strokes since FEROFT (first effective ROFT since the last RefTime)	N/A (value taken from TDI_cs at the beginning of every time slot)	N/A (TDI_cs assigned at the 1st detection of every source packet)
TSMoff	TSM offset	TDI clock strokes since the last TSM	At the beginning of every time slot	Every TDI clock stroke
TSsou	Time Slot sources	Number of sources included in the current time slot (or source packet)	At the beginning of every time slot (i.e., source packet)	Every new source detected (including the 1st one of the packet)
AsmTime	ASM time (one per instrument)	Fractions of TDI clock strokes since TSM. It will be assigned to DTT when detecting a source	At the beginning of every time slot	Every *DTTres*th fraction of *TdiPer*
FDI	Field Density Index	Density of the field currently measured, expressed in stars/s per FOV	At every change of RefTime	At every star detection

Table 7.3: Counters and parameters used during the coding process

The flux diagram of figure 7.28 illustrates a possible implementation of a telemetry coder fulfilling the *Adaptive TDC* recommendation described in this chapter. This flux diagram uses a set of queues and processes, and the parameters of table 7.3. The queues have already been described in section 7.2.6 (and illustrated in figure 7.13), which are used for storing the data set headers (*Qref*) and the science data from the sources detected and measured (*Qsou*). On the other hand, the following are the main processes executed:

- *GenPointer* (generate pointer): it generates a unique pointer for the current source detected, which may be as simple as its detected transit time (DTT) with an absolute coding.
- *MeasSou* (measure source): it will take all the data of the source being measured (including DTT and other detection data) and store it in the sources queue (*Qsou*), without any change of the time codifications (so the data will surely be stored in an absolute form). The corresponding codifications proposed in this chapter will be applied afterwards.
- *GenTimeSlots* (generate time slots): it takes the MTO value determined from the current FDI (using the guidelines given in section 7.3.5), and will generate the time slots for the several sources detected (more exactly, for their detected transit times). It labels every time slot with its corresponding TSM, and stores the time slots array in *Qref* (in the same "file" where the current RefTime, TdiOff, etc. are stored).
- *CodeDTTs* (code detected transit times): it converts the DTT fields to their adequate coding (with the bit length determined by MTO). Both *Qref* and *Qsou* will be updated with the new codifications of the DTT fields.
- *ServeDS* (serve the data set): when all the data of a data set is available (i.e., after *Maximum Measurement Latency*), it will be "served" by putting them in the corresponding data set structures. This is, it will execute the guidelines given in section 7.2.6, by generating the AP #A source packet with data from *Qref*, and the several AP #B/C source packets with data from *Qsou* (using the time slots generated by *GenTimeSlots*). Source Packets will then be stored in the solid-state mass memory. When finished, the current *Qref* and *Qsou* files can be deleted.

The figure uses the term *FEROFT* (first effective ROFT) for referring to both the TDI offset and ASM TDI flag, since $FEROFT = TdiOff + AsmTdiFlag \times TdiPer$. Note that several instances of the processes (specially *MeasSou*) will be executed concurrently (or will wait for

their execution on the background): although at *CodeDTTs* exit the system starts again with a new *RefTime*, the several processes are still running in background during *MaxMeasLatency*.

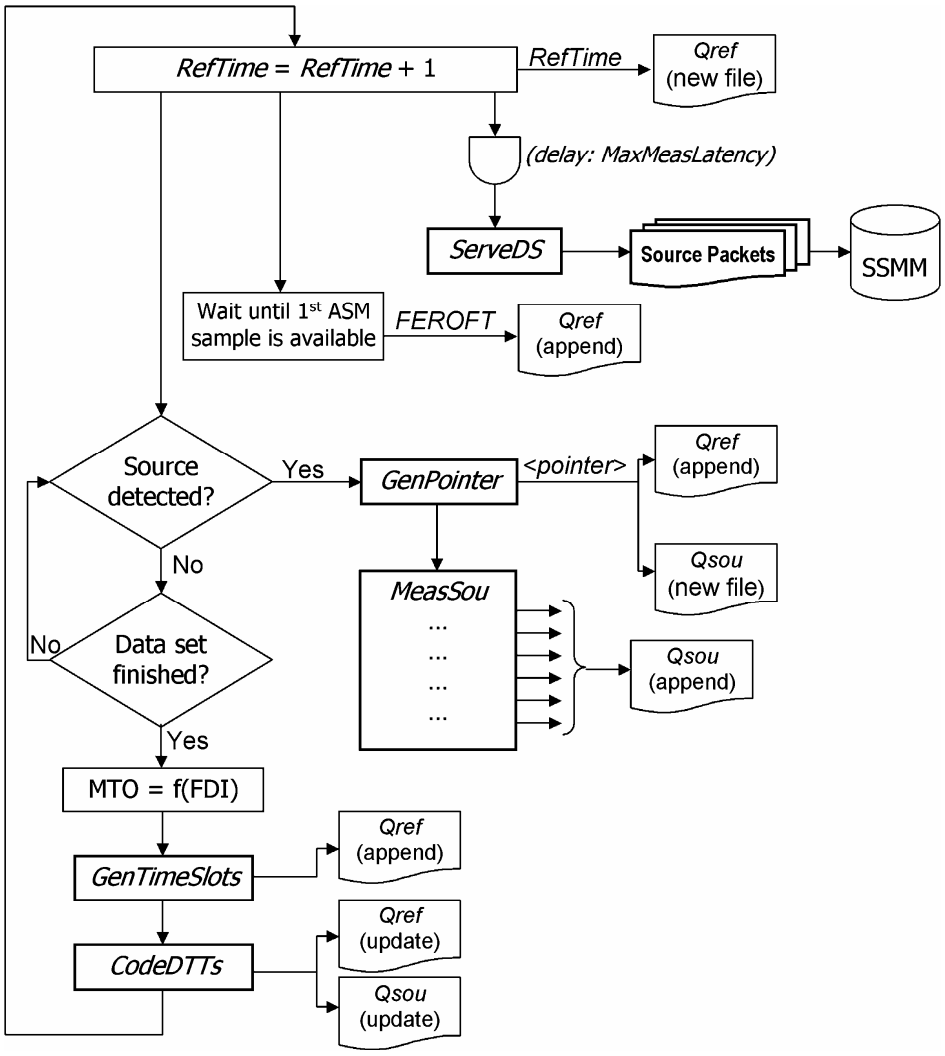


Figure 7.28: Flux diagram of a telemetry coder

7.4.4. Conclusions on the performance of the codification scheme

In section 7.3.5 we show how we can save some telemetry bandwidth by using this codification scheme. More than 1 kbps will be saved in average (compared to an equivalent GDAAS2-type codification) by simply transmitting the data in an optimized way, without the inclusion of any data compression but just using *time slots*. Furthermore, the quality of the data (and its reliability) is much higher, offering a resolution in the DTT fields 20 times higher than in GDAAS-2 (Masana et al. 2004) and fulfilling the CCSDS standards on time code formats (CCSDS 2002). Finally, the source packets generated in this way are totally prepared for being input to the data compression system, which will be described in the next chapters.

It is important to note that the data rate saving and total data rate obtained with the simulations may be pessimistic compared to the real case: our simulated codification system seems to lose accuracy at very low star densities (which have a very high probability in Gaia), because the real system will not generate as many packets as we have simulated. This is, when measuring an area of the sky with very few stars, Gaia will generate only the necessary packets (as defined by our guidelines), while we have simulated an averaged case. Therefore, it is possible that a telemetry coder applied to, e.g., GASS output data, will offer yet better results than we expect.

Another important feature of this codification system is its robustness in front of transmission errors. This robustness is obtained thanks to the sequence control system included in the data set headers, which indicates the starting packets of the current (and previous) data set for every application process. Since these indicators (the *source sequence count* values) are 14 bits long, even in very dense star fields (say, 1500 stars/s per FOV) they have a recycling time of more than 10 seconds, so the loss of single AP #A packets can be easily solved. Furthermore, in extremely dense star fields (where the recycling time can be less than 1 second), the behavior of the TSM fields would help in reconstructing the data sets. The same applies to the adaptive system, the behavior of which is signaled enough with a redundant MTO flag. Finally, single losses of source packets containing Astro data are also acceptable, since every packet includes a CCSDS-compliant time tag.

7.4.5. Alternative timing schemes

The timing scheme recommended in this chapter is based on a counter of the seconds elapsed in the satellite clock, adding a couple of data fields for indicating the readout time of the first sample in the current data set. Afterwards, every time field is referenced to this *FEROFT* (first effective readout finish time). Although this scheme seems optimal in terms of telemetry consumption and simplicity, conversations with U. Bastian and U. Lammers raised the possibility of improving this codification. For this, the following assumptions are used (some of them already used in this chapter):

- All the CCDs of the astrometric focal plane operate with exactly the same TDI period.
- The charge transfers in all the pixels of all the CCDs (i.e., TDI clock strokes) are commanded from the same, unique clock line.
- Time delays in the propagation of electronic signals to the focal plane, implying phase shifts in the individual commanding clocks for each CCD, are negligible for the final astrometric accuracy.

With these prerequisites in mind, one can think of an easier timing scheme, avoiding as much as possible any complication in the decoding of time data (reference time, plus a TDI Offset, plus an ASM TDI Flag, plus the time data itself, etc.). This alternative scheme would be based on ASM readout times, not on the "ideal" second-based scale. These are the main guidelines:

- The overall organization would be similar, based again on data sets.
- A data set would start at the exact readout moment of an ASM sample. This is, at the first effective ROFT (equivalent to RefTime + TDI Offset + ASM TDI Flag×TDI_period, in the current scheme).
- The length of a data set would be as close as possible to a second. The best approximation, assuming that 1 Astro TDI period is 736.6μs, is 1358 TDI clock strokes. This would lead to an effective data set length of 1 second and 302.8μs. It means that a data set can be considered about 1 second length, adding a "leap second" every 55 minutes. Anyway, since we isolate the data set structure from the second-based scale, a data set may have an arbitrary length –always of an integer number of TDI clock strokes. This may lead to further optimizations of the codification scheme.
- The data set header would only need a reference time (with no extra flags), with an improved resolution between 1 and 16 nanoseconds.

These are the basics when only Astro instrument is taken into account. As expected, the timing scheme is much more simplistic, and possibly some bits may be saved this way. On the other hand, we get in trouble when including the MBP and RVS instruments, which –obviously– use different focal planes than Astro. It means that we will have to include anyway a set of TDI offsets (and even SM TDI flags) in order to know the behavior of those focal planes. Furthermore, a timing scheme based on the Astro focal plane is obviously excellent (in terms of telemetry) for the Astro itself, but not for the MBP and RVS, thus losing

any advantage gained for Astro. A solution to this, as indicated by U. Bastian, would appear naturally if the TDI period used in Spectro would be an integer multiple of the Astro TDI period. In this case, we could base the data sets on the Spectro readout times (instead of Astro), thus reducing the required flags. Anyway, in the simulations we have seen that the consumption of telemetry due to the data set headers is not really significant, so any improvement on this direction is not really worth it. Further studies may reveal improvements in this (or other) alternatives, but in our work we will assume the use of this optimized time data codification.

Chapter 8

Improving the Communications Channel

This chapter describes a proposal for improving the downlink capability of Gaia based on reducing the minimum elevation angle of reception to increase the daily contact times between the ground station and the satellite. Its feasibility by including extra correcting codes was already verified in a previous study (Geijo 2002), where the case of a LEO minisatellite was studied. In that case, the satellite was able to transmit with an elevation angle of only 3° over the horizon, although for Gaia we will limit it to 5° to keep a larger margin. The main purpose of the work described in this chapter was to inform the scientific and technical community of Gaia about the possibility of increasing the average downlink capability of Gaia, which would not only relax other mission requirements (such as the required data compression ratio) but would also make possible the transmission of more scientific data than expected. The scientific impact of this improvement is obvious, while the technical, design and implementation costs are acceptable.

8.1. IMPROVING THE DOWNLINK CHANNEL OF GAIA

As already described in previous chapters, Gaia will orbit around the L2 point, at an approximate distance of 1.5 million kilometers from the Earth. This orbit implies that the ground station will have only a few hours a day of direct visibility of Gaia, which is a similar case than the LEO minisatellite studied previously (Geijo 2002, Geijo et al. 2004). The current baseline assumes Cebreros (Spain) as the only Ground Station (GS) used by Gaia. This location, at 40.27°N 4.22°W, will have a 35-meter antenna that will be operational in September 2005. Figure 8.1 (provided by U. Lammers) shows its polar mask, where we can see an important obstacle at the west –but still allowing communications at about 5° of elevation.

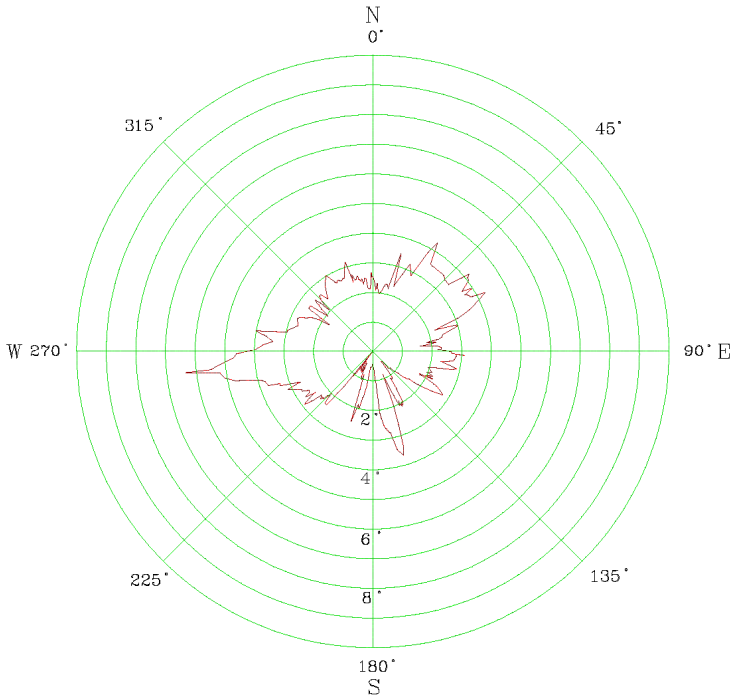


Figure 8.1: Polar mask of the future antenna in Cebreros (Spain)

The antenna will have to move from east to west for tracking Gaia, that is, from the raise of Gaia over the horizon until its set. The inclination of the apparent position of Gaia will never reach 90°, because of the high latitude of the ground station. It will also depend on the season of the year, e.g., during the summer in Spain it will be tracked with a lower elevation than during the winter, so the antenna will reach higher elevation angles during the winter in Spain. Anyway, it will never reach 90° and will have to point always to the south. This implies that the obstacle at the west really has to be taken into account –specially during the winter, when the antenna will get closer to 90°.

The current baseline for Gaia assumes that communication with the Ground Station will only be possible for elevation angles higher than 10°. Although we demonstrated that a satellite can transmit data up to 3°, our proposal for Gaia will be more conservative: our target is to reach the 5° elevation angle (thus also avoiding the obstacle shown in fig. 8.1). Nevertheless, an extension of "only" 5° per side (rise and set) can provide an important increase in the average telemetry capability: the extension of 2° in the minisatellite already implied an improvement of more than 10% in the average capability. The improvement will be even higher because of the high latitude of the ground station. Figure 8.2 (gently provided by U. Lammers) shows the increase in the daily contact times when switching from 10° to 5° minimum elevation angle, for three different ground stations (each at a different latitude). As expected, the case of Cebreros is where we can see the highest improvement (because of its higher latitude). Furthermore, the peaks in this figure correspond to the days when the daily contact times are shorter –hence this decrease in the minimum elevation angle leads to more uniform contact times.

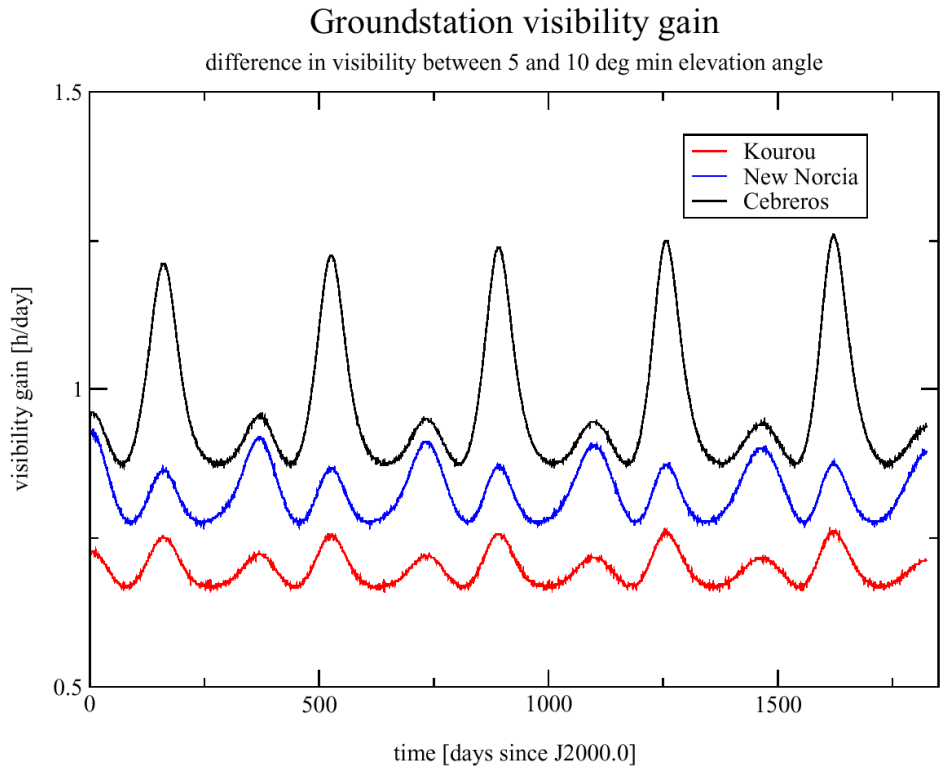


Figure 8.2: Improvement in the daily contact times for Gaia

Figure 8.3 shows the final contact times that we could achieve in these ground stations when assuming only 5° minimum elevation angle. The average gain in these daily contact times is of 1 hour in Cebreros, while in New Norcia it is 0.8 hours/day and in Kourou it is 0.7 hours/day. It is clear that the gain is more important when using ground stations located at higher latitudes. The method for making possible the communication at such low elevation angles will be the same as in that preliminary study (Geijo 2002): to increase the correcting capability of the error control codes included in the telemetry. The case of Gaia is a little more complicated, and its possible implementation will be discussed in the following section.

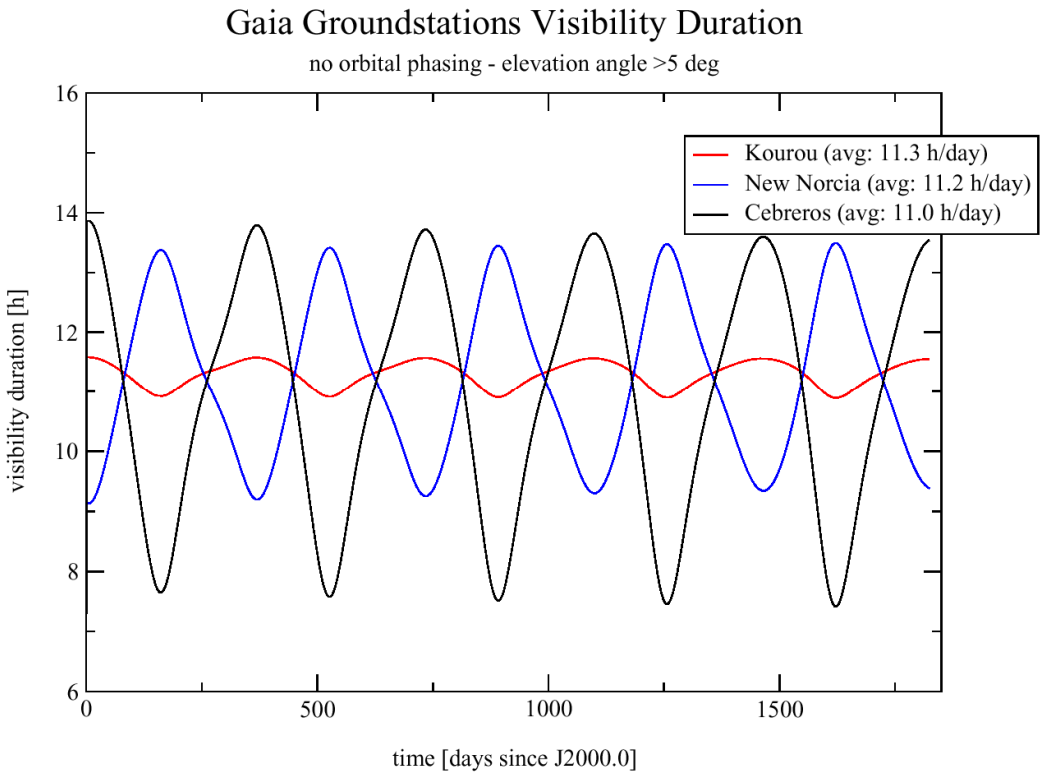


Figure 8.3: Daily contact times between several ground stations and Gaia

8.2. POSSIBLE IMPLEMENTATION

8.2.1. Implementation requirements for an adaptive system

The following are the main requirements envisaged for an adaptive implementation of this error control system, following the guidelines described in Geijo (2000).

1. **Bidirectional communication channel between the Ground Station and Gaia.**

   As seen in Geijo (2002) and Geijo et al. (2004), a satellite can adaptively improve the correcting capability of the codes included in the telemetry thanks to a set of "acknowledgements" received from the ground station. Therefore, an uplink with enough capability is required for this operation, reporting to the spacecraft the correct or incorrect reception of every telemetry frame. The case of Gaia, however, is more complicated because of its higher downlink capability. This higher transmission rate implies that the reception of much more frames must be acknowledged to the satellite, and therefore a much higher uplink capability is required. Specifically, the uplink required for an operation of this kind can be calculated in this way:

   - Every TF (transfer frame) received on ground will be acknowledged to Gaia (at least for low elevation angles, e.g., <15°).
   - The top transmission speed of Gaia will be close to $5 \cdot 10^6$ bps.
   - A Transfer Frame will usually be 10232 bits long (ESA 1988).

     → Up to 489 TF/s will be downlinked by Gaia to the GS.
   - The acknowledgement of a TF would require, at least, some of the fields of the *TF primary header* (ESA 1988), specially the *frame count* values. The minimum size of an ACK/NACK can be considered of about 32 bits.

     → About 16 kbps of uplink are required.

Taking into account that the baseline of Gaia only offers 2 kbps uplink, an alternative strategy should be developed.

2. **Additional on-board memory bank.**

   In order to make possible the re-transmission of a transfer frame that has been received with errors, all of the transmitted TFs must be stored in an extra memory buffer until they can be deleted (ACK from ground) or re-transmitted (NACK from ground). The size for this extra bank is not very large: assuming a maximum distance to Gaia of about 1.7 million km (for taking into account the Lissajous orbit), the round-trip time is less than 12 seconds. Then, with a maximum data rate of 5 Mbps, the maximum size required for this bank is about 60 Mbits.

3. **Inclusion of a feedback line from the telemetry module to the PDHU.**

   This is an on-board requirement, which will input the ACKs/NACKs received from the GS by the telemetry module towards the PDHU. The latter will be responsible of updating the extra correcting codes introduced in the telemetry.

4. **The GS must be capable of pointing towards low elevation angles.**

   The antenna must be able to point towards Gaia even if it is only 5° over the horizon. This is the most important of the requirements. If it is not possible, this alternative coding scheme should be discarded, although pointing losses may be evaluated and the benefits recalculated.

8.2.2. Recommended implementation

In the case of Geijo (2002) it was not mandatory to fulfill any telemetry standard, so the flexibility for introducing the adaptive correcting codes was complete. On the other hand, in the case of Gaia we have to fulfill the Packet Telemetry standard which implies, among other requirements (ESA 1988):

- Fixed TF size
- Fixed size of the error control field (Reed-Solomon codes)

Therefore, we cannot change the length of the TF field neither the R-S codes for adapting to low elevation angles (i.e., a lower quality channel). This implies that the extra correcting codes will have to be included as science data, within the *data space* of the transfer frames. This, at the end, requires a telemetry system of Gaia flexible enough for permitting the introduction of such data in the transfer frames.

There is another problem introduced by the requirements of Gaia: the limited uplink capability. Only 2 kbps are available in the baseline, and these will be mostly required for telecommand and other operations. Therefore, the basis of an acknowledgement for every TF (continuously adapting the correcting capabilities) is not realistic.

The alternative is to modify the correcting capabilities using a fixed scheme which, anyway, may be updated by the GS during every contact. This scheme should not affect the standard transmission range (i.e., above 10°), but operate only below this angle by following this protocol:

1. Contact start (5°): Maximum correcting capability assumed. Expected efficiency: >50%.
2. Elevation increasing (5°…10°): During this period the GS may send some reports of the channel quality to Gaia[26], which will adapt the correcting codes included in the telemetry. These reports may be based on statistics of the number of corrected (and not corrected) errors, thus recommending a given correcting capability to the PDHU of Gaia. The uplink required for this would surely not exceed 64 bps (e.g., transmitting a 64-bit flag every

[26] It implies that the GS must have the capability of generating such reports.

second). The correcting capability used should be exaggerated, because the transfer frames with errors cannot be re-transmitted (as they are not acknowledged from the GS).

3. Nominal elevation reached (>10°): Baselined transmission scheme, with the nominal efficiency of 86.7% (ESA 1988). An alternative option may also be studied: during this nominal phase, the GS may acknowledge to Gaia the several transfer frames received with errors (not only during the minimum elevation phase but also during the nominal phase), so that they could be re-transmitted. In this way, the probability of frame loss (PFL) of the mission would decrease significantly. It is important to note that the report of these errors from the GS to Gaia should imply the lowest possible uplink requirement (e.g., 100 bps), for not disturbing the nominal telecommand operation. Also, it implies that Gaia should have an extra memory bank with a large size.
4. Low elevation reached again (10°): The telemetry system of Gaia enters again in "safe mode", including the extra correcting codes in the data fields. Again, the GS will periodically report the channel status to Gaia, which will smoothly increase the extra correcting capabilities introduced in the data. As during the contact start, these correcting codes should also be exaggerated –for fulfilling the nominal PFL.

Following this scheme, the extra memory buffer indicated before will not be needed anymore (if the option proposed in item 3 is not considered), and the requirement on the uplink will be much more relaxed –even negligible.

Another possibility, if the global PFL of the mission decreases too much because of this scheme, would be the following: to take advantage of these low elevation angles for transmitting only "extra" data, such as imaging, false detection data, or low-priority (or outdated) data. The only requirement would be an extra memory bank for storing all these data, which would imply about 10 Gbits.

8.2.3. Benefits

Preliminary calculations indicate that the daily contact time between the Ground Station (Cebreros) and Gaia, in average, is about 10 hours. We have seen that the average improvement would be about 1 additional hour per day, which represents 10% more contact time. Furthermore, the variations in the daily contact time will be slightly smaller (which is also beneficial for the mission), making the final reliability of a high latitude GS like Cebreros closer to the reliability of low latitude GSs.

If we take into account an average efficiency of 65% (which implies a 75% over the 86.7% baseline) during the low elevation range (10° – 5°), the equivalent improvement is 7.5% additional telemetry capability. Recent studies indicate that Gaia will be able to transmit data at about 5 Mbps. With an average contact time of 10 hours/day (baseline) and 86.7% efficiency (ESA 1988), the average useful data rate would be 1.8 Mbps. With our proposal, the useful data rate would be increased to about 1.95 Mbps in average. This proposal was received with enthusiasm by scientists of the mission, and is currently being studied by the engineering teams.

Chapter 9

Telemetry CODEC Simulator

The science data fields to be included in the telemetry of Gaia, as well as their codification, the timing and the transmission schemes have already been defined in the previous chapters. The simulators that generate realistic telemetry data, such as GASS or GIBIS, are also available. The next step forward towards a complete simulation framework is to convert these simulated data accordingly to the codification and transmission schemes defined in the previous chapters. Here we define the main operation and implementation guidelines for a new software that will convert Gaia simulated data to realistic telemetry data, taking into account both the specific Gaia definitions (such as the timing schemes) and the flight-realistic standards (such as Packet Telemetry). We have called this new software *Telemetry CODEC* (coder/decoder), since it will both code and decode the data in order to verify its correct operation.

One of the objectives of the software tool introduced in this chapter is to verify the reliability and efficiency of our previous designs, such as the *Adaptive Time Data Codification* described in chapter 7. It will also make possible the calculation of telemetry curves and statistics in order to complement those already available (Lammers 2004), obtained with analytical and simpler galaxy models. Finally, it will be mandatory to have realistic binary data for the data compression simulations and this is, indeed, the primary goal of the software here described, namely to provide a solid basis for the simulations of data compression. Actually it is envisaged to also include data compression modules in this software. Finally, it is important to note that the TM CODEC will be a simulation software, not the final software to be integrated in the spacecraft or in any industrial demonstrator.

The chapter is organized as follows: section 1 lists the general requirements and specifications for the TM CODEC, as well as the conversion phases proposed. Section 2 describes two alternatives for implementing this software, and each of them will be described in detail in sections 3 and 4. Each of these two sections will include a description of the identified modules for the CODEC. Finally, section 5 proposes an implementation roadmap.

9.1. REQUIREMENTS OF THE TM CODEC

9.1.1. General requirements

We have conceived the Telemetry CODEC (or TM CODEC) as a complementary software for the several simulators available for Gaia. These simulators are, obviously, focused on the scientific results, so they offer outputs which are not flight-realistic telemetry –although some of them may offer the adequate data fields and formats. For example, GASS output is mainly used for GDAAS, so a realistic telemetry output is not mandatory (only realistic scientific contents).

On the other hand, realistic telemetry simulations are needed, in order to check the design of the codification and transmission schemes (chapter 7), and to verify existing estimates (Lammers 2004) on the amount of data to be transmitted by Gaia. These simulations should be as realistic as possible, and since there are already some excellent simulators available it is convenient to use their output data as a source for flight-realistic telemetry, rather than developing a full simulator. The main objectives for this software are the following:

1. Accept data from the available Gaia simulators (currently GASS, but also GIBIS and other simulators in the future), converting them to “digital”, flight-realistic data fields.

2. Convert or modify the data fields so that they correspond to the actual telemetry model (in the case in which the simulator is simplified and does not offer this possibility).
3. Packetize these data together and offer a single, standard telemetry stream. Separate outputs shall also be selectable.
4. Offer telemetry curves and statistics, in order to complement the currently available estimates based on smooth galaxy models, and to verify the telemetry-related designs (such as the *Adaptive TDC*, see chapter 7).
5. Make possible the integration of data compression software within its application framework.
6. Be flexible enough to adapt to most of the simulators and to possible modifications in the telemetry models or in the design of Gaia itself.

9.1.2. Coding steps

The TM CODEC appears as a large software application, performing several complex operations on large data sets and adhering to a given *telemetry model*. As in any other complex task, the best option (if possible) is to break it down to several smaller and simpler tasks. In our case, we have deployed the TM CODEC (from an operational point of view) in several steps, each of them described below. Stand-alone modules should implement these steps, thus making possible, for example, to include in the TM CODEC the resulting software of the forthcoming data compression study.

9.1.2.1. Quantization

The latest version of GASS offers telemetry data as a text file containing the several data fields. Some of them are coded as "analogue", this is, using the physical units instead of ADUs (Analog-to-Digital Units). This applies specially for all of the flux values, this is, the samples of the ASM, AF or BBP patches, which are coded as e^- without verifying the sample capacity. This first conversion step will be in charge of *quantizing* these data, this is:

- Verify that each value fits within a given range (e.g., in the case of an AF sample, about 300.000e^-). Otherwise, the maximum value (or zero, if negative) will be assigned, and an overflow (or underflow) flag will be raised for statistical purposes.
- Scale the value accordingly to the specified range, thus offering the value in ADUs.

Besides this conversion, the rest of the simulation output will be kept as-is (this is, the order will not be changed, and no data field will be added/suppressed). The output of this step will be binary quantized data. It is worthy to note here that this quantization process should be implemented in the simulator itself (e.g., GASS), although an external implementation also has some advantages, such as an easy evaluation of the quantization errors.

9.1.2.2. Conversion

Simulating Gaia is an extremely complex task, and even an excellent simulator may not reproduce all of its operating conditions. Also, the simulators are also implemented in a progressive way, so that some output modes may not be used at a given stage. For example, the current GASS version (2.2) does not use the full sampling scheme yet, this is, some window modes (specially for the brightest stars) are not used. On the other hand, it may be interesting to have simulated data containing a realistic case of the sampling scheme, this is, each star should be transmitted with the adequate window and sample sizes. The TM CODEC could modify (even falsify) this simulator output in order to offer realistic TM sizes –obviously taking into account that some of these data may not be used for scientific post-processing. Scaling and interpolation operations may be required to implement this. Also, it may be necessary to suppress some fields only used in simulations (such as the Source Id, Masana et

al. 2004), which would unnecessarily increase the amount of TM data. Finally, a *telemetry model adaptation* could also be performed in this step, e.g., adapting the output of a simulator to a more recent telemetry model. The output of this TM CODEC step will be binary flight-realistic data. This conversion process should also be intrinsically done in the simulator itself (e.g., GASS), so in future stages the TM CODEC shall not require this second coding step.

9.1.2.3. Packetization

After quantizing and converting, we already have realistic TM data but as *raw* data. Flight-realistic data must also fulfill the standards, and more specifically the Packet Telemetry standard (CCSDS 2000, ESA 1988). Its implementation will not only require a specific *conversion* step (adding the fields for the packet headers, etc.) but also complementary operations in order to obtain *calculated* fields. For example, the *source sequence count* field of a source packet (ESA 1988) would require a counter, and the *packet data length* field would require to calculate the total size of the packet.

During this packetization process, also the *Adaptive Optimized TDC* scheme will be implemented (see chapter 7). This is, the data fields will be coded in an optimized way, and the time slots will be calculated and generated. This will also require additional operations to be executed on the simulated data.

Finally, the several data sources (i.e., the several instruments) will also be multiplexed in order to generate a single data stream for all the science data. At the end, the output of this TM CODEC step will be source packets that can be directly fed into the communications system.

9.1.2.4. Pre-compression

Our proposal for the transmission scheme inverts the usual operation, packetizing the data *before* being compressed. This is done not only in order to apply the optimized codification and transmission schemes, but also in order to keep the individual star sets identified –so that the priorization process during transmission will be much easier and versatile. Therefore, at this stage the TM CODEC should also be able to simulate and test the compression process, integrating the necessary pre-compressors and compressors with the adequate configuration for Gaia data.

As already verified in some preliminary studies (Portell et al. 2002b), standard compression systems cannot offer the high compression ratios required for Gaia. On the other hand, applying simple pre-compression systems (Portell et al. 2001b) one can significantly improve the results. This is why we split the compression process in two sub-processes, making possible their separate study and optimization if required. The TM CODEC shall include these pre-compression operations in this step. Partitioning the telemetry stream in several uniform sub-streams (each with the same field sizes and similar data behavior) appears as an interesting, powerful alternative (Portell 2000), so the output of this step will surely be a set of sub-streams processed in one way or another. Forthcoming studies will determine in detail the optimal data compression system.

9.1.2.5. Compression

This step may be implemented as some kind of standard compression system, although the use of tailored implementations of them (e.g., using different symbol sizes) is envisaged. If a stream partitioning system is used, surely different concurrent compressors will run here, although their outputs shall be synchronized in order to offer a single stream. Whatever solution is selected, it would be interesting to integrate also this option within the TM CODEC framework. This would be the last coding step of the TM CODEC, so that the output stream of this will be the one to be fed into the communications system.

9.1.2.6. Decoding phase

The steps described in 9.1.2.1 to 9.1.2.5 correspond to the coding phase of TM CODEC, but there must obviously be a decoding phase as well. All of the coding steps will have an equivalent decoding step, so that decoding the data will execute the following:

1. Expansion: decompress the data, offering the several uncompressed sub-streams.
2. Re-combination: combine the several sub-streams into a single data stream containing the source packets.
3. Packet decoding: decode the source packets, restoring the time data fields and de-multiplexing the stream into the several instrument streams.
4. Interpretation: Undo the *conversion* process, trying to restore the original science data. This will be the only TM CODEC operation that may not be perfectly inverted, since the conversion tools may include interpolations or even field suppressions. This must obviously be taken into account: for example, if the TM CODEC is to be inserted in GDAAS, unilateral conversion operations must be avoided. This interpretation step shall be removed when the simulator itself implements the conversion process.
5. De-quantization: digital quantized data will be received, so this last step must offer the science data in their natural units. This step will also be removed if the simulator already offers digital quantized data. Actually, the final data processing system (even a future GDAAS version) shall work directly with digitized data.

9.1.2.7. Additional steps

With the steps described before we can already obtain accurate estimates on the telemetry volumes, data compression ratios, or even on the on-board occupation resources such as the solid state mass memory (SSMM). Furthermore, the compressed source packets offered by step 5 can be directly fed into the communications system, so in a principle we should not worry about what happens after that. Nevertheless, it could be interesting to obtain the actual telemetry volumes after generating the transfer frames (ESA 1988) including the correcting codes, or even to estimate the actual probability of packet loss (PPL) with a simulated communications channel. This would lead to two additional steps, *frame generation* (with the corresponding *frame decoding*) and *channel simulation*. With this, one could test the effect of some parameters on the transmission reliability (such as the minimum elevation angle required to establish the communication between Gaia and the ground station), or even an alternative channel coding as introduced in chapter 8 (Geijo 2002). These two additional steps are currently beyond our objectives and will not be studied in this work, although they appear very interesting (specially the frame generation) and should be included in the future.

9.1.3. Overview of the system

Figure 9.1 illustrates the global operation of the TM CODEC, indicating the several coding/decoding steps described in section 9.1.2. In order to illustrate how this software could work with other simulators (integrating their data into a single telemetry stream), and how it could be part of a larger simulation environment, GASS and the RVS Simulator are included as the data sources, and GDAAS is included as the science data processing software. A stream partitioning compression system using 3 sub-compressors is used as an illustrative example. We recall here that some of these steps could disappear in the future, thus integrating coding steps 1 and 2 within GASS or RVSsim, and decoding steps 1 and 2 within GDAAS.

It is worthy to note at this point that attitude and housekeeping data are ignored in the current design of the TM CODEC. They will suppose an amount of data that will be much smaller than science data, and this is the main reason why we have focused our efforts on the simulation and optimization of the latter. Nevertheless, they should be taken into account, so that future improvements in the TM CODEC may also include attitude and housekeeping data.

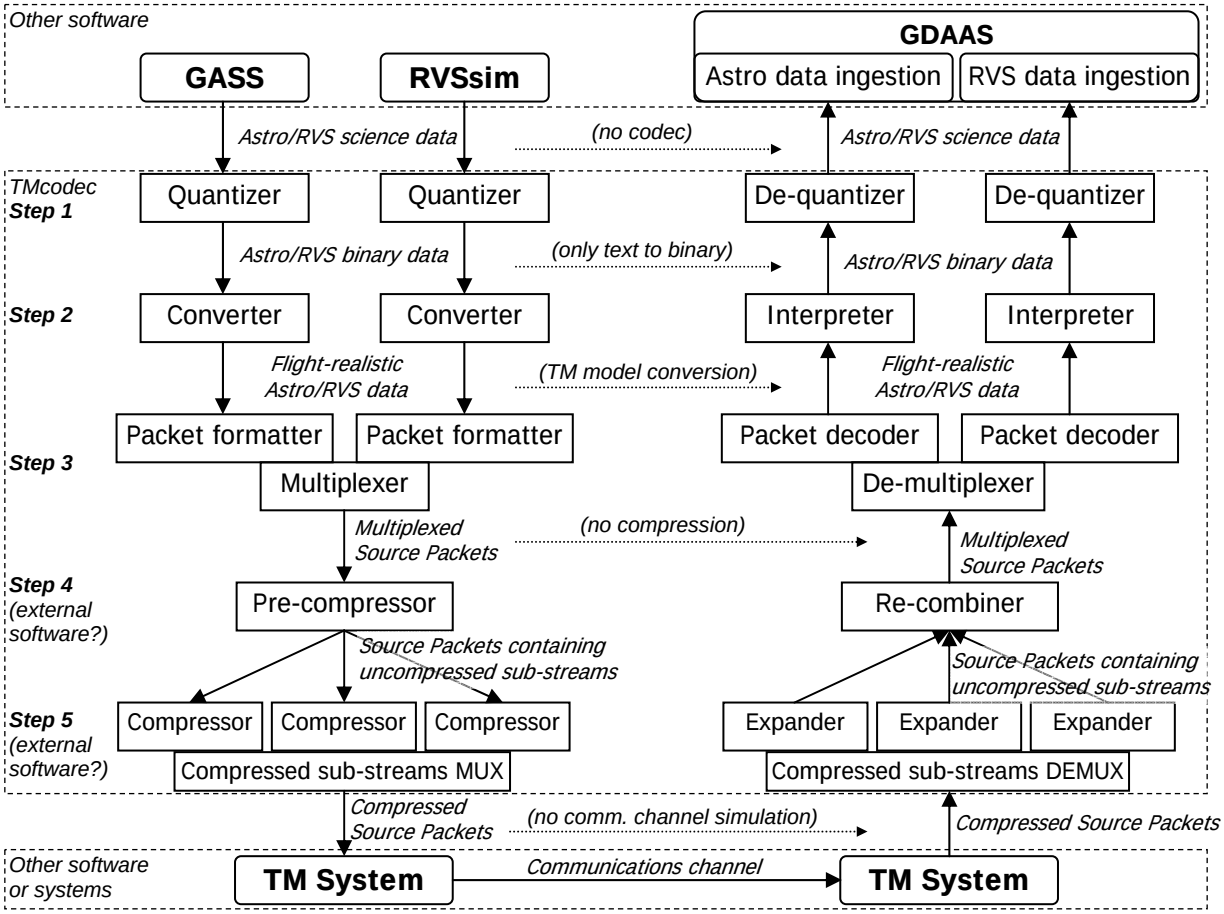


Figure 9.1: Overview of the TM CODEC operation

9.2. IMPLEMENTATION ALTERNATIVES

From a practical point of view, the Telemetry CODEC must be as modular as possible, so that any of its elements could be improved or even substituted without affecting the correct operation of the full framework. Even though, these modules could be implemented progressively, and each of them could be a separate stand-alone application. In this way, the coding (and decoding) steps could be verified separately, or even "switched off" in the TM CODEC framework if they are not required in some specific simulations. As an example, now GASS directly connects to GDASS prior to step 1, indicated as a dotted arrow with the "*no codec*" caption in Fig. 9.1. A very simple TM CODEC, only quantizing and approaching GASS data to flight-realistic formats, could be integrated in GDAAS after the 3rd step ("*TM Model conversion*" in Fig. 9.1), and so on. Note that we are using GDAAS as an example, but the inclusion of such a TM CODEC software in GDAAS is not defined yet. Also, the TM CODEC should be flexible enough to be integrated in other simulation environments. Finally, if necessary, the software modules implementing the TM CODEC could be external software – such as the resulting software for the optimal data compression system for Gaia.

9.2.1. Static implementation

Regarding the configuration of the TM CODEC, we distinguish two different alternatives. The simpler one is to develop a software specially devoted to the current TM models of Gaia, implementing its data management with a fixed code and specifying parameters (such as field sizes or sampling schemes) as, for example, a ".h" file in C or C++ language. This option would be simple in terms of coding effort (and a fast implementation is envisaged). On the other hand, any change in the global design or in the parameters of Gaia would imply the

modification of this code. Its application to other simulators would also imply a revision and modification of the code, or even a complete redesign of it. Nevertheless, this *static* implementation option is extremely interesting, since we can already achieve important results within a short time scale. We consider this option as a test bed for the TM CODEC. This will allow to explore later a more complete and flexible software to be eventually developed.

9.2.2. Dynamic implementation

The other option is to implement a generic data handling software, capable of performing any of the required operations in the TM CODEC pipeline and using any of the possible configuration parameters, including different TM models (i.e., format of the received science data). This software would be configured with some external files, so that modifications in most of the parameters would only imply an editing of this file, without the need to modify any code. This is our preferred option, which we call the *dynamic* option. Its counterpart is that as a generic, flexible and robust software application, its implementation will take much longer than the static option. We will describe both options in detail in the following sections.

9.2.3. Common implementation guidelines

Although there are different options to implement and configure the TM CODEC, some tools (or libraries) that will be required in order to manage the science data are common to both alternatives. We have already started developing them in the C++ language. We have chosen this programming language because its object-oriented capability becomes not only useful, but even mandatory due to the complexity of the TM CODEC. On the other hand, this language makes possible faster executions that other object-oriented ones. The following are the most relevant modules developed (or identified) for any implementation option:

- File management tools: as seen in the telemetry model definition (Masana et al. 2004), the several data fields have different bit sizes which not always correspond to an integer number of bytes. Since the C++ file system can only read and write bytes (not separate bits), a set of methods were developed in order to read (or write) an arbitrary number of bits from (or to) a file. These tools have the form of a C++ object, containing generic methods that unify and simplify the input/output (I/O) operations for both text and bit files. These methods even provide a *standard stream* I/O method, so that the TM CODEC does not necessarily have to work with files but redirect the data between processes. This will be really useful to reduce the execution speed of the simulations, and to avoid large intermediate files.
- Error management, display and logging tools: a set of methods to display any additional information (such as the progress of some operation), warnings, errors or even debug information, both to the screen or to a log file.
- Miscellaneous tools: non-standard functions such as the calculation of base-2 logarithm or base-2 exponential, offering the number of bits required to code some value (or the maximum value that can be coded with a given field size). Also other tools could be included here, such as the configuration of a global verbose level and method (so that modules will report more or less information, and to the console or to a given log file).

9.3. STATIC IMPLEMENTATION

Most of the modules of this implementation of the TM CODEC software are already working and have been successfully tested. As explained before, the static implementation is the simplest approach to obtain realistic telemetry data for Gaia, and despite of this we have already obtained very interesting results. These include telemetry curves, density curves based

on the *Field Density Index* (FDI, see chapter 4), and preliminary estimates on the data compression ratio.

9.3.1. Configuration of the CODEC

The most relevant parameters are defined in a C header file, *gaia_def.h*, including:

- Focal plane dimensions (number of CCDs)
- Number of TDI periods between CCDs
- Patch and window dimensions in every field (ASM, AF and BBP)
- Bit sizes for every data field
- Maximum sample and pixel capacity

These parameters are already available in the Gaia Parameter Database (de Bruijne 2003a), and future versions of the TM CODEC shall directly link to the *.h* files automatically generated by the database software rather than defining its own parameters. Regarding the data management, it is statically implemented as C functions in which we define the several operations to be performed for every data field. Therefore, any change in the TM model or in the data compression system will require a modification of this code. Nevertheless, this task does not imply a large effort, so small changes can easily be made.

9.3.2. Capabilities

These are the following operations that can be done with this simulation software:

- Conversion of GASS output data from ASCII (text) to binary data, with the field sizes indicated by the telemetry model (Masana et al. 2004).
- Simple conversions of the TM model. Examples:
  - Conversion of the TDI Offset Matrix from 18×10 values to 2×10 values (see chapter 7).
  - Conversion of AF windows to their adequate sizes, depending on the brightness of the source. For example, the window mode for the brightest sources is not implemented yet in GASS. The TM CODEC could do this.

  We must note that, despite of these capabilities, none of these conversions has been currently implemented in order to avoid "falsifications" of GASS data.
- Calculation of telemetry and density curves, offering the number of bits per second and stars per second with different integration times.
- Calculation of a flux histogram, in order to illustrate the typical star fluxes in a given simulation.
- Stream partitioning (data pre-compressor), for testing purposes.
- Data compression with standard systems (gzip, bzip2, etc.) and modified systems (adaptive Huffman and LZW with variable symbol sizes), for testing purposes.

Other capabilities will be implemented soon, such as the verification of the final output (after decoding the data), including RMS error calculations wrt the original data input.

9.3.3. System overview

The static implementation of the TM CODEC is intended to be a simplistic version of the final software, acting like a test bed for the most important procedures to develop –specially the data compression. Therefore, some of the steps defined in section 9.1 have been grouped in a single module, while others do not have an implementation yet. Figure 9.2 illustrates the actual

operation of this software, where "C" modules represent data compressors and "E" modules represent data expanders.

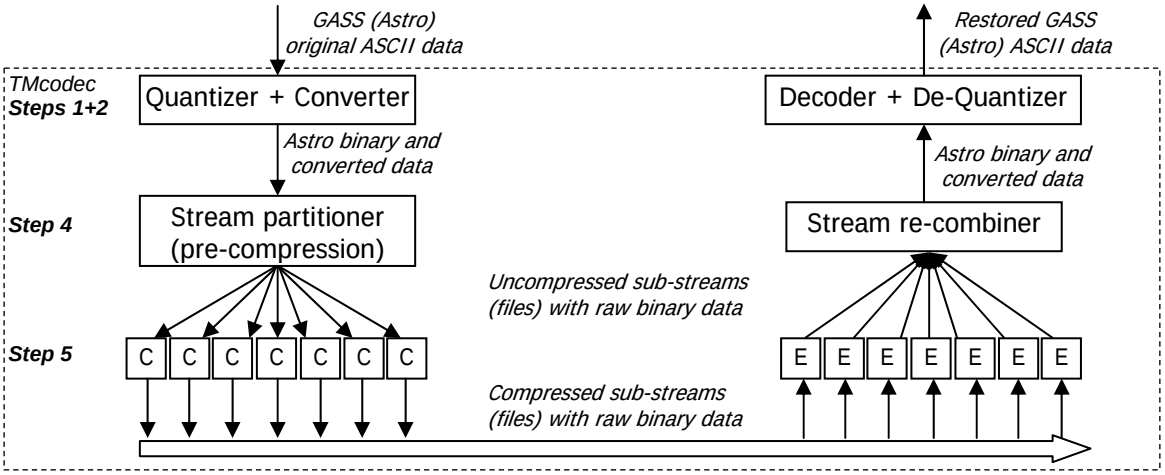


Figure 9.2: Static implementation of the TM CODEC software

As it can be seen, the generation of source packets and the multiplexing of compressed sub-streams are not implemented here, as well as multiplexing of data from different simulators. The main reason is that this approach is a simplistic one, just for testing our ideas. Another reason is the need for statistical analysis of Gaia science data, in order to know which data sub-streams represent a higher telemetry occupation, and which of them are easier to compress. Packet structures (with their corresponding overheads) could confuse the results, and multiplexing the compressed sub-streams would not make possible the independent study of their statistics in a simple way.

9.3.4. Modules

Here we list and briefly describe each of the modules of the static TM CODEC, implemented in C++.

9.3.4.1. Text-to-binary converter

This is the core of the TM CODEC, receiving the text files generated by GASS and generating binary (bit-level) files. It uses *gaia_def.h* to code every value with the adequate number of bits, so that the output size is much more realistic, thus indicating the real telemetry size generated by GASS. This module, *txt2bin.cpp*, is also in charge of performing the adequate modifications to the data, such as telemetry model conversion or sample scheme adaptation. It also generates text files reporting the FDI (Field Density Index) and TMC (Telemetry Curves) values for each given integration time, so that we can plot these curves. At the end, a *.bin* binary file is generated, containing all of the data received from GASS.

The *txt2bin.cpp* module also detects any underflow (negative values) and overflow events, reporting their statistics together with the maximum values for each data field in ADUs (Analog-to-Digital Units). It will be improved in future versions, offering an histogram of the values taken by every data field –which will be mandatory if we want to design an optimal data compression system. Finally, additional statistics such as the number of stars and data sets processed, as well as the average conversion speed in Mbps, are reported.

9.3.4.2. Data pre-compressor

Science data generated by Gaia (or by its simulators) contains data fields with very different symbol sizes, as well as different statistical behaviors. Therefore, since a standard compression

system usually operates with symbol sizes of byte- or word-length, they cannot take full advantage of the redundancies of the data. This is where the pre-compressor module becomes critical to achieve high compression ratios, since its function is to modify the data in such a way that can be better "understood" by standard compression systems. Even though, it can prepare the data for a non-standard compression system, such as standard systems with variable symbol sizes.

Although detailed studies on data compression are envisaged for the next months (including the design of an optimal data compression system for Gaia), here we wanted to obtain some preliminary results on this issue. More specifically, we developed *gaia-spap.cpp*, a stream partitioner and pre-compressor module which breaks the data down to separate files. The objective is to have files with more uniform data, so that even standard compressors could offer excellent results. These are the sub-streams generated by *gaia_spap.cpp*:

- Reference data, including ASM detection data and other flags
- TDI Offset data, with the values from the TDI Offset matrices
- ASM data, with the samples from the ASM patches
- Readout time data, with the readout times for all the AF and BBP windows
- Readout position data, with the readout positions for all the AF and BBP windows
- AF data, separating AF01-10 samples and AF11 data
- BBP data

This stream partitioning also makes possible the calculation of the telemetry occupation percentage for each type of data. Furthermore, *gaia_spap.cpp* can also modify these data, e.g., using differential coding rather than an absolute one. All in all, it represents a good test bed where we can implement different pre-compression techniques.

9.3.4.3. Data compressors and expanders

After the application of the pre-compression module, standard and adapted compression systems were tested on the resulting data. They include gzip and bzip2, as well as modified implementations of Adaptive Huffman and LZW taken from Portell (2000), among other systems. The results were satisfactory, significantly increasing the achieved compression ratios. Detailed results will be presented in next chapters, as well as an indication of the best pre-compressor and compressor combinations. All in all, our main objective here was validated, since the modular approach for the telemetry CODEC operates as expected.

9.3.4.4. Data post-expander

After expanding again the compressed data we obtain the files with the corresponding sub-streams of raw binary data. Another C++ module is being developed, which must re-combine these files in order to obtain the raw binary data in a single file and with the correct transmission order. This will be basically the "inversion" of *gaia_spap.cpp*, so both modules must be designed in a way that no data will be lost during the process.

9.3.4.5. Binary-to-text converter

This module, also under development, will undo most of the operations performed by *txt2bin.cpp*. It is important to note that not all the operations can be restored, such as the suppression or interpolation of fields. This module will also use the range definition indicated in *gaia_def.h*, so that de-quantization can be performed. At the end, a single text file will be generated which should contain the original GASS data whenever possible (e.g. when no conversion has been done previously).

9.3.4.6. CODEC manager

This is the front-end application for both coding and decoding processes. We will enter the original text file to this module and a set of optional parameters such as the data compressors configuration. It will code the data file and decode it back, offering final statistics such as the errors between the original data file and the resulting file. It will be used to validate all the process pipeline, so that we can be sure that there is no error in any of the intermediate modules.

9.4. DYNAMIC IMPLEMENTATION

Selecting the best options obtained with the test of the static TM CODEC, the dynamic (generic) TM CODEC will be developed. Here we describe the main guidelines to be used when implementing this, which will be done during the next months.

9.4.1. Configuration of the CODEC: XML files

The primary objective of the dynamic TM CODEC is to offer a generic software, valid for almost any simulator and any telemetry model, as well as many data compression configurations. The key for its flexibility is its configuration files, which will be in XML format. Several files will be used to configure all the TM CODEC steps:

- Simulator TM model: TM model used in the output files of each of the simulators. The final output of the TM CODEC, which shall be fed into the data processing system (such as GDAAS), will also use this format.
- Intermediate TM model: TM model used to generate the flight-realistic data stream.
- Multiplexed Source Packets TM model: it indicates the way to generate the several Source Packets, and how to combine the data from different simulators (using different TM models).
- Compressed TM model: both the pre-compression and compression operations are parameterized using this file.
- Global TM CODEC configuration: it indicates which XML files (i.e., TM models) must be used to perform a simulation for some simulator or set of simulators. Also global parameters of the simulation will be configured in this file.

The XML files containing the TM models will not only contain the TM model itself, but also indication of the special operations to be performed on the data. It may include counters, statistics, parameter export or file combination in the global environment, and array interpolation, scaling and data compression for specific data fields. For example, the GASSv2 telemetry model may be described including conversion scripts for retrieving GASSv1 data. This may be useful, for example, to estimate the telemetry occupation with a Gaia-2 design, but taking advantage of the large GASSv1 (Gaia-1) simulations already available. However, this must be kept as an illustrative example only, since there may appear unavoidable errors in the final estimates –e.g., due to different spin speeds or focal plane sensitivities.

9.4.2. System overview

While the static implementation operated almost in a sequential manner (without buffers), the dynamic philosophy stores large amounts of data in memory buffers, making possible many complex operations such as the determination of the *Adaptive MTO* (see chapter 7) or the determination of the total packet size (taking into account that it will be composed of several data sets), as well as complex data compression methods. The operation will be based on scripts following the TM model, so that these will be the main operating steps:

1. The input and output TM models will be loaded as a script (stored in RAM)
2. The several data fields of the output TM model (i.e., the script) will be linked to their data sources in the input TM model
3. Telemetry data will be loaded in blocks, following the input TM model
4. Data processing will be performed (scaling or interpolation operations, multiplexing, data compression, statistics, etc.) as indicated by the output TM model
5. Resulting data will be then written following the output TM model

This process will be done for each of the adequate steps of the global TM CODEC. Given the flexibility of our XML structures, steps 1 and 2 (quantization and conversion) of the TM CODEC can be grouped again without any problems. Figure 9.3 illustrates the usual operation of this dynamic TM CODEC, where the decoding steps have been omitted for the sake of simplicity. These decoding steps would be completely equivalent to the coding ones, except for different XML files used (which will be the "inverse" of those used during the coding process).

9.4.3. Modules

These are the envisaged modules that will compose the dynamic TM CODEC. Some of them have already been implemented as C++ classes, while others are being tested yet or even not implemented. We must recall that this is conceived as a large and generic software application (figure 9.3 gives an idea of this). Therefore, not all the modules must be implemented if not needed –such as the MUX (multiplexer) module, if only single-simulator essays are required. The modular design of the TM CODEC makes possible this progressive, user-requested development.

9.4.3.1. XML parser

This element is in charge of loading the XML file, processing it and offering high-level functions to retrieve the configuration for the TM model and operations. Standard (and free) libraries are available for this, although we preferred developing a simple XML parser offering the basic functions, which is enough for the preliminary tests carried on. Nevertheless, it should be substituted in the future by a standard library in order to make possible the interface with other standards, such as the DTD validation or even the integration in the forthcoming standard for TM definition. Annex A includes a sample file that defines the TM model for GASS v2, as well as the conversion scripts to retrieve data from GASS v1. The following are some of the tags used:

- `struct`: devised as a data container, easing the definition of complex data structures – or arrays of complex data structures.
- `field`: main data specification, indicating:
  - `name` of the field (a user-defined identifier)
  - `itemsize` (in bits)
  - `length` of the eventual field array. Also `lengthdefiner` can be used if this length depends on the value of another field.
  - `condition` for transmitting this field
  - `source`, `value` and `pattern`, indicating the data field (of the *input* TM model) from which the data is retrieved, as well as the eventual operations performed.

  All of these attributes can also be applied to `struct` (except for `itemsize`).

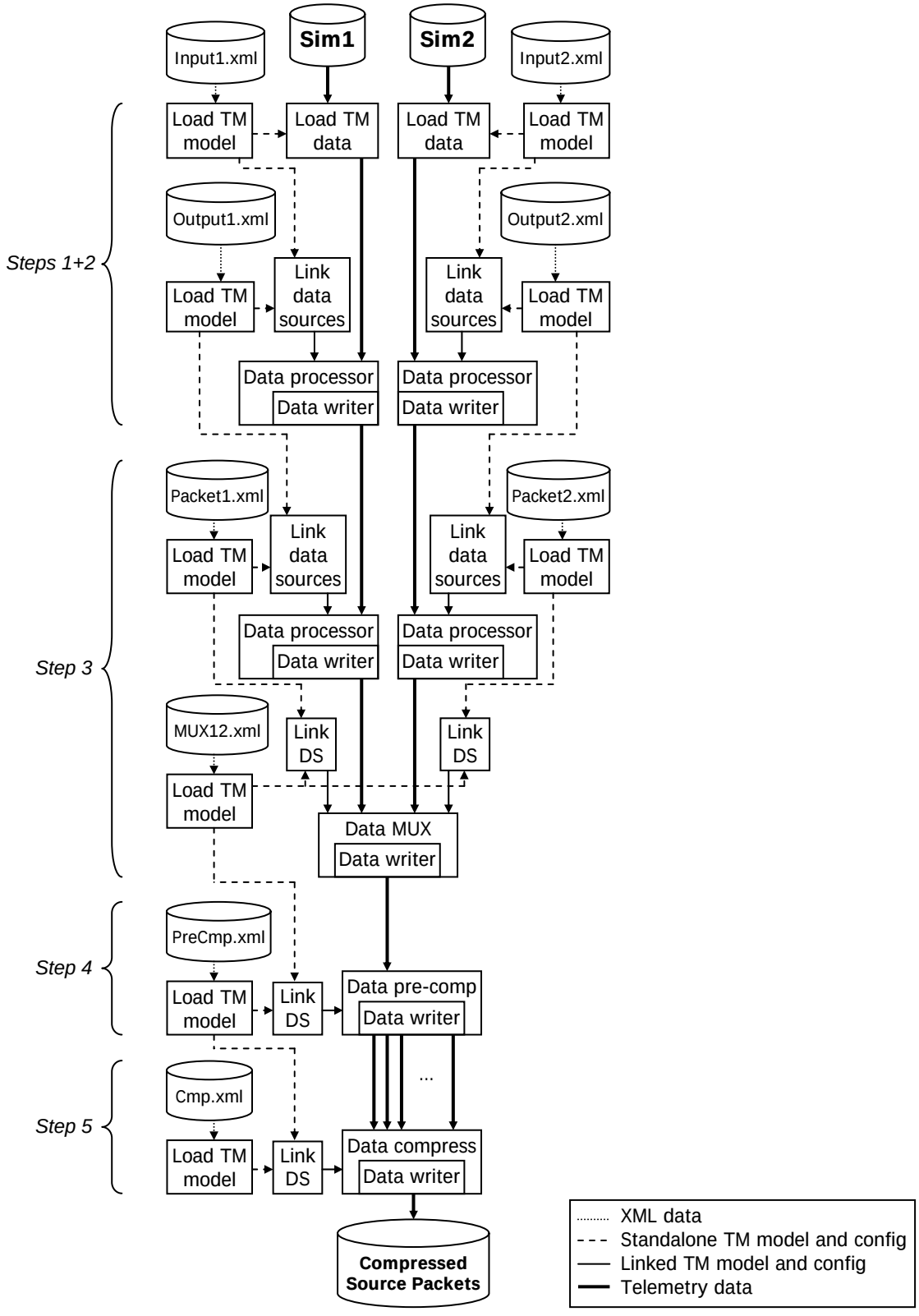


Figure 9.3: Operational diagram of the dynamic TM CODEC

9.4.3.2. Telemetry model loader

This module uses the XML Parser tool to load the TM model to memory as a "script" structure, including all of the required elements to correctly load (or write) the TM data. These elements include the several data fields to be "transmitted" (or "received"), grouped in structures if necessary, as well as their sizes in bits, transmission conditions or special operations. We have implemented a preliminary version of the TM model loader

(*tmmload.cpp*), creating a memory structure based on a vector of *telemetry model nodes*, each containing:

- Node name and identifier
- Field size in bits (if it is a field)
- Transmission condition:
  - Conditioning field name
  - Type of condition (*equal*, *different*, *greater* or *lower*)
  - Value for the condition
  - Relational operator (*and*, *or*), to make possible different conditions for a same node.
- Length (if fixed-length array)
- Length-definer node name (if variable-length array)
- Structure contents, which at the same time is another vector of *TM model nodes*.
- Data source:
  - Data source node name (and pointer)
  - Data source parts to retrieve (if the data source is an array)
  - Data source scaling (as a *pattern*, independent for each part retrieved)
  - Interpolation specification
  - Data source options (*MSB*, *LSB*, *DIV* or *MOD*)
  - Data source operation, making possible the combination of different data sources into the same node
- Pointer to the corresponding block in the TM data buffer.

9.4.3.3. Telemetry model to HTML converter

The TM model loader is one of the critical parts of the dynamic TM CODEC, since it builds a complex memory structure which will strictly indicate the operations to be done and the order in which they must be performed. Also, the creation of an XML file defining the operations that we want to do, although not being an extremely complex task, may lead to some errors that could not be detected by the parser. Therefore we devised *tmm2htm.cpp*, a C++ class that saves the TM model (the memory structure, not the XML file) in an HTML file, with a user-friendly format. In this way, we can verify with any HTML browser that we have correctly defined the TM model –and, during debugging phases, that our TM model loader operates as expected.

9.4.3.4. Data loader

The TM data loader is the next mandatory element in the dynamic TM CODEC. It uses the *TM script* (the memory structure created by the TM model loader) to load blocks of a simulated data file, such as a GASS telemetry file. This software module, implemented again as a C++ class, basically defines a large memory buffer of up to some tenths of megabytes (depending on the TM model, the operations to perform, and the TM data themselves). Afterwards, a basic set of tools are offered, making possible to load the TM data by blocks –such as data set by data set. These data will be recovered by the next modules using some high-level interface methods.

9.4.3.5. Data writer

If no special operation has to be performed on the TM data –except for a text-to-binary conversion (or vice versa) or redistribution in the TM model, the data writer can directly be executed to generate the output TM file. This module will simply contain a *data retriever* (retrieving the data from the TM buffer) and the high-level calls to the file writing methods described in section 9.2.3. On the other hand, if special operations must be executed, the data writer may also call data processing functions defined in other modules, such as the *data processor* or the *data MUX* defined below.

9.4.3.6. Data processor

This module offers several data processing tools that can be executed from the data writer. The resulting data could be saved either to another memory buffer (if more TM CODEC steps remain) or to an output file. This module will be specially dynamic, in the sense that it can keep growing with additional data processing tools as required by the users.

9.4.3.7. Data MUX

Only the use (and multiplexing) of data coming from several data simulators will require this additional module. Its main objective is to retrieve data from different buffers (or even files), using different TM models for each data buffer and combining them as indicated by another TM model. The multiplexed result will also be saved either to another memory buffer or to a file. Another module to be used in the decoding phase, the *DEMUX* module, will be in charge of inverting this operation –thus expanding the data depending on their type: Astro, MBP, RVS, etc. The difference is that now we can dynamically modify this operation, so if the final data analyzing system (e.g., GDAAS) is capable of receiving the data from several simulators at the same time, the DEMUX module could operate with simper operations –or could even be suppressed.

9.4.3.8. Data pre-compressor

The data pre-compression principles to be used in the dynamic version of the TM CODEC are the same than those described in section 9.3.4.2, for the static TM CODEC. The only difference is that the dynamic version takes the pre-compression parameters from an XML file, integrated with the flight-realistic TM model. Actually, some of the XML files illustrated in figure 9.3 could be combined, thus indicating the TM model, data processing operations and data compression configuration in a same file. During the decoding phase, an equivalent *data post-expander* module will be executed accordingly to the "inverse" XML file used to pre-compress the data.

9.4.3.9. Data compressors and expanders

Again, this module will be equivalent to that one in the static version (see section 9.3.4.3), except for its configuration with XML files.

9.4.3.10. CODEC manager

The TM CODEC manager will be the only module using a different specification of the XML files. Rather than TM models and operations, it will receive a set of CODEC steps to perform, each with the following parameters for both the input and output files:

- TM model (XML file)
- TM data file

- Data mode (binary/text)
- Read/write method (file, standard port, socket, etc.)

Then, this main TM CODEC module will be executed, calling the adequate sub-modules as required (i.e., depending on the operations requested in the XML files). It would be interesting to implement the TM CODEC Manager in such a way that different processes shall be created for every module, and that the several steps shall be interconnected with pipes or sockets. In this way, the execution speed would drastically increase, and multi-processor environments would automatically take advantage of this.

9.5. IMPLEMENTATION ROADMAP

The telemetry CODEC will be implemented in several phases, thus improving its compatibility and capabilities step by step and as required by the simulation needs:

- Phase 1: Basic operation with GASS Astro data, with static configuration.
- Phase 2: Inclusion of data compression for GASS Astro data, with static configuration yet.
- Phase 3: Migration to a generic, dynamic implementation, with XML-based configuration. This would imply several sub-phases, one for each TM CODEC module.
- Phase 4: XML development and operation test for MBP and RVS, extending to other simulators (GIBIS, RVS Simulator, etc).
- Phase 5: Inclusion of generic data compression, and inclusion of its optimal configuration for Astro, MBP and RVS within the XML files.

Phases 1 and 2 have already been implemented and tested successfully. Phase 3 has been partially implemented, since some of the dynamic modules are already operative or under development. Phases 4 and 5 shall be implemented in the near future. In each of these development phases, compatibility with the GaiaGRID environment (Ansari et al. 2004) shall also be tested, making possible its direct coupling with GASS. Also, a possible integration within GDAAS (as an intermediate stage between GASS and the telemetry extractor) will be tested.

Chapter 10

Telemetry CODEC Tests

After describing the main capabilities and operational guidelines of a Telemetry CODEC and implementing a static version of this software, a first campaign of tests has been carried out using data generated by GASS version 2.2. Although this version of the GASS simulator does not offer a completely realistic output, these tests offer interesting results which include star density curves, telemetry curves and flux histograms. Different sky regions with different star densities have been tested. Several objectives are pursued with these simulations:

- Verify that the *txt2bin* (and *bin2txt*) software operates correctly, offering a reliable TM CODEC –although in its static version. The performance of this software will also be evaluated.
- Obtain statistics of star field densities and telemetry rates, as well as histograms of AF peak fluxes. Different sky regions will be considered for this.
- Evaluate the consistency of the results when compared to the telemetry rate tests currently available.
- Offer an additional tool for testing and validating GASS results.
- Obtain a repository of GASS TM data, not only in its original (ASCII) version but also in binary version, as well as the restored version (i.e., ASCII including quantization errors). These data will be useful for data compression simulations, and for evaluating the effects of the quantization errors in GDAAS.

The chapter is organized as follows. Section 1 offers an overview of the simulation environment. Section 2 contains all the results obtained from the test campaign, including plots and numerical results. Section 3 shows an example of the restoration process with the TM CODEC and GASS data. Finally, section 4 offers our conclusions on these results. A description of the MatLab® software developed for plotting these results is included in annex C. All the C++ code developed for this has not been included in order to avoid an excessive document size. This software (either the source code or the compiled binaries for a given architecture) are available upon request to the authors.

10.1. SIMULATION ENVIRONMENT

10.1.1. TM CODEC

For these telemetry, field density and quantization tests we have used the static (*test bed*) implementation of the TM CODEC. Actually, in order to verify as much libraries as possible, a "hybrid" implementation of *txt2bin* (text-to-binary converter) was developed, in such a way that the telemetry definitions are taken in run-time from an XML file, rather than in compilation-time from a *.h* file. In this way, the user can modify many parameters of the TM CODEC without modifying and re-compiling the code. It also makes possible to test the XML parser library, as well as this XML-configured approach. We must note that only some parameters such as field sizes and ranges can be modified, but not the TM model structure.

On the other hand, the *bin2txt* (decoding, or binary-to-text converter) software has been kept as "fully static", this is, using the parameters defined in the *.h* file. The main reason is a simple matter of time, since we preferred not spending more time in the "hybrid" implementation –as this software version is a test bed. However, we must note the dangers of this hybrid/static

mixture: the user must be sure that the parameters in the XML file are coincident with those parameters in the *.h* file already included in the compiled software. The decoding software (*bin2txt*) would display warnings or errors during the process (and the resulting data will obviously be useless) if field sizes are wrong, but inconsistencies in the field ranges would not be detected. Fortunately, once the TM model and code ranges are fixed, the only parameters of the XML file that shall be usually modified by the user are related to the statistical analyses of GASS data, and these parameters do not have any counterpart in *bin2txt*.

10.1.1.1. Txt2bin features

- Text-to-binary converter, quantizer and data processor
- Version 1.0.1 (released 10-Dec-2004)
- Main features:
  - Operation as described in the previous chapter: quantization of the adequate data fields, conversion to binary, data analysis, etc.
  - Hybrid configuration: TM Model structure statically configured at compilation-time (*.cpp* main file), as well as the data analyses; TM field sizes and ranges, sampling scheme, physical dimensions and data analysis parameters configured in run-time with an XML file.
  - Elementary syntax tests: reference times are checked so that they do not exceed a nominal mission length, also verifying that they always increase. The used windowing modes are also checked (so that they are accepted by the current sampling scheme). Overflows in data fields that do not need quantization (and, hence, an overflow would have no sense) are also detected.
  - Statistics: FDI (Field Density Index) curve and 2 IFDI (Integrated FDI) curves, TMC (Telemetry Curve) and 2 ITMCs (Integrated TMC), all of them with selectable integration times. Histogram of AF01-10 peak fluxes (with selectable number of bins). Number of overflows and underflows in the quantized fields. Peak values (in ADUs) and quantization step (in e^-) for the most relevant fields. Percentage of use for each windowing scheme. Total observation time, number of sources and data sets, average number of sources and average data rate.
- Limitations:
  - TM files to be analyzed cannot be larger than 2 GB. It seems a limitation of the file system, although it should be investigated and solved if possible. For the moment, if larger simulations have to be analyzed it shall be done by parts, or the *txt2bin* software could be improved in order to automatically analyze (and convert) a set of files sequentially.
  - The conversion process is very slow. It seems a problem of the C++ I/O standard library: the bottleneck seems to be the readout of strings from a file and their conversion to integers or floats.
- Possible improvements:
  - Standard I/O (*stdin*/*stdout*) operation, in order to make possible its operation within GaiaGRID.
  - The effect of payload hardware limitations could be evaluated. In particular, the number of stars per second could be limited in order to simulate the VPUs and CCDs capability, and the amount of binary data generated per second could be also limited in order to simulate the capability of the payload data buses (and the mass memory size). Then, the exceeding number of stars and amount of data could be displayed or registered in files. In this way, for example, only a small part of the stars in Baade's window could be analyzed and converted (thus better simulating the real mission).

10.1.1.2. Bin2txt features

- Binary-to-text converter and de-quantizer
- Version 1.0 (released 10-Dec-2004)
- Main features:
  - Operation as described in the previous chapter: de-quantization of the adequate data fields, and restoration to an ASCII format (maintaining the original carry returns in order to ease a visual comparison).
  - Static configuration: TM Model structure defined in the main *.cpp* file, and TM Model parameters (field sizes and ranges, sampling scheme, etc.) defined in a *.h* file.
  - Elementary syntax tests: reference times are checked so that they do not exceed a nominal mission length, also verifying that they always increase. The window modes used are also checked (so that they are accepted by the current sampling scheme).
- Limitations:
  - The 2 GB upper limit also applies here, but now the user should take care of not generating an output file larger than this (the ASCII output file can be up to three times the size of the binary input file).

10.1.1.3. Libraries used

- XML Parser library:
  - Version 1.2 (released 18-Oct-2004)
  - Main features: elementary (sequential) parse of the file, retrieving the tag identifier, attributes and contents.
  - Limitations: no DTD validation.
- Gaia File library:
  - Version 1.2 (released 19-Nov-2004).
  - Main features: bit-level and ASCII-text routines for data input/output using files or stdin/stdout. Quantization option included (longints or floats). Text write capability.
  - Limitations: input/output process is slow, although it is currently being optimized.
- Gaia EDL (Error, Display and Log) Tools library:
  - Version 1.0 (released 18-Oct-2004).
  - Main features: display of errors, warning and user-defined texts depending on the verbose level (which can be selected as *none*, *basic*, *detailed* or *debug*). Progress reports and progress bars.
- Gaia Misc library:
  - Version 1.0 (released 18-Oct-2004).
  - Main features: miscellaneous tools, such as the selection of the default verbose level, mathematical tools for the management of binary numbers and ranges, etc.

10.1.1.4. Full package tests

All this software, packed as *Gaia TM CODEC (static) Release 1.0.1*, can be requested to the authors and has been successfully tested on the following platforms:

- Intel Itanium2 1.6GHz (SGI Altix 3700 Bx2), RedHat Enterprise Linux
- AMD Athlon XP 2600+, Gentoo GNU/Linux

- 2 x Intel Xeon 2.4GHz HyperThreading, Linux
- 2 x Intel Xeon 2GHz, RedHat GNU/Linux
- Intel Pentium-M (Centrino) 1.4GHz, Gentoo GNU/Linux
- Intel Celeron (Pentium-II) 333MHz, Gentoo GNU/Linux

Its operation under other OSs such as Windows or Macintosh has not been verified. An adequate C++ compiler is required, although pre-compiled packages can also be requested. Some platform definitions (such as HUGE_VALF or LONG_LONG_MAX) were found as undeclared in Debian systems, which will be solved in future versions in order to assure the maximum compatibility with any platform. Less than 32MB of RAM are needed for the execution of this software, and the average conversion speed (including data analysis) is about 1.6 Mbps in an Athlon XP 2600+. We are currently working on the improvement of this software, specially focusing on the I/O library. Although a new release has not been finished yet, a partial implementation has revealed a much faster execution speed –about 10 times faster than this stable release, as well as better compatibility with other platforms such as Tru64.

Finally, we must note that although this TM CODEC version is only a "test bed" for the final version, it has been developed to be reliable enough for being used in other environments such as GDAAS. That is, it is not an isolated software application just for self-testing purposes, but an application ready to be used as part of other larger simulation environments.

10.1.2. GASS (Gaia System Simulator)

The GASS version used for these tests is 2.2, which offers a TM output with most part of the current sampling scheme (Masana et al. 2004):

- Window modes 0 (faint) and 1 (mid-bright) are used. Mode 2 (brightest) is not used yet.
- Only AF11 mode 0 is used.
- Both BBP modes 0 and 1 are used.

It also uses the latest design of Gaia (sometimes indicated as "Gaia-2"), although there are some aspects that differ from the real design –and from the premises used in other telemetry simulations (Lammers 2004:

- The overall sensitivity has not been updated yet, which implies lower star densities than in the real case.
- The Galaxy model is not completely realistic, only single stars in the main H-R sequence are simulated, and the limiting magnitude is indicated in V instead of G. It also underestimates the amount of stars detected and measured.
- The Galaxy absorption model is also simplistic, but in this case it overestimates the star densities.

These under- and overestimates of the star densities more or less cancel out each other, offering final observations which are fairly realistic. Nevertheless, it is very important to note that these GASS v2.2 results, and therefore the statistical analyses obtained with *txt2bin*, cannot be considered as the real case but only as an acceptable approximation. TM data obtained from improved GASS versions shall be applied to the current version (or future versions) of *txt2bin* in order to obtain more realistic estimates.

This test campaign has been the first time at which GASS v2.2 has run at magnitude 20 (Mag20). It has caused some problems, since the amount of stars generated has been very large –as well as the file sizes, and, obviously, the simulation times. One of the parameters of GASS, the HTM level, had to be increased in order to avoid memory problems. With this, we realized that an excessively high HTM level increases too much the simulation time. These are some examples of this (when executed using CPUs of about 2GHz):

- 1 day for 15 days at magnitude 10 (requiring HTM level = 4).

- 5 days for 2 hours at magnitude 20 in a mid-crowded region (requiring HTM level = 9).
- 2 days for 8 minutes at magnitude 20 in a crowded region (requiring HTM level = 10).
- 3 days for 3 minutes at mag. 20 in a very crowded region (requiring HTM level = 11).
- 10 hours for 3 seconds in Baade's window (requiring HTM level = 12).

These results, furthermore, could not have been obtained using the standard execution of GASS, with which we found a lot of memory problems –implying too high HTM levels. Even though, some areas could not be simulated at all, such as the Baade's window. The solution to this was to manually increase the Java heap size, up to 256 MB or even 384 MB. This solution also makes possible the use of lower HTM levels and, hence, faster execution times.

We must note that the simulations generated during this campaign do not exactly coincide with those selected beforehand, but they are usually shorter. The main reason is the time limitation, since longer simulations would have delayed too much obtaining acceptable results in a reasonable timescale. Also, some simulations had to be aborted in order to avoid files larger than 2 GB. Finally, computer limitations (and some problems with the latest GASS distribution when executed in GaiaGRID) led to prohibitive simulation times. Nevertheless, the simulation results finally obtained completely fulfill our objectives, since the statistical studies have been carried out successfully and lots of binary data (to be used in forthcoming data compression simulations) are also available.

10.2. SIMULATION SCENARIOS AND RESULTS

In this section we describe the several tests undertaken both with GASS (to generate the input data) and, specially, the Telemetry CODEC. Since one of the objectives is to generate realistic binary data for the data compression study, different observation conditions shall be studied – as they may affect the achieved compression ratio. Furthermore, in this way, resulting star densities and telemetry rates could help in sizing on-board hardware. In order to cover the most typical observation conditions, the following scenarios have been selected:

- Gaia scanning the Galactic Plane (GP) perpendicularly, usually leading to a low average star density. This would be the typical case, even a little optimistic (in the sense that the star densities achieved would be lower than the average of the mission).
- Gaia scanning the GP tangentially, leading to high star densities. This would be the pessimistic case, i.e., leading to higher star densities than the mission average.
- Gaia scanning Baade's window, leading to peak star densities and telemetry rates which will surely overload the on-board hardware and communications systems.

Using an SWG web tool that computes the mission time at which Gaia scans a given area in the sky [IR.4.], approximate times for these situations were obtained:

- Around days 89-90 (wrt 2010.0) Gaia should scan the GP perpendicularly.
- Around days 119-120 Gaia should scan the GP tangentially.
- Around day 113 Baade's window should be observed.

Taking into account these time coordinates, we planned a set of "observations" (that is, simulated observations) and tests which are explained in the following subsections.

10.2.1. Magnitude 10 simulations: looking at the global picture

First of all, a long-term overview that includes all of the intended observation times has been simulated. More than 30 days had to be simulated (from day 89 to day 120), and therefore this test must have a low limiting magnitude –in order to avoid excessive simulation times and resulting file sizes. Magnitude 10 was selected, and large margins were included in order to

have a richer overview: 79 days were simulated in total, from day 54 to almost day 134. It generated a file of almost 2 GB, which was analyzed afterwards with *txt2bin*, including the "*-noout*" option. With this option the software only analyses the data, without converting and quantizing it, which avoids a useless and large output file and slightly accelerates the process. We stress that this output file is useless because binary files will only be used for data compression in the future; such a low limiting magnitude is absolutely unrealistic, and therefore achieved compression ratios would not be representative.

The console output of *txt2bin*, shown in figure 10.1, contains many additional data related to the simulation, while the detailed statistical analyses (FDI curves, TM curves and flux histogram) are saved to files for a better analysis with a MatLab application that we have developed. The parameters used for the statistical analyses (specified in the XML file) were a "real-time" integration of 2 minutes (for FDI and TM curves), 30 minutes for the short-term IFDI and integrated TM curves, and 6 hours (one great circle) for the long-term estimates. We also requested 100 bins for the flux histogram, which has been maintained for all of the simulations.

```
Analyzing /mnt/e_win/gassSims/data/tm/d55-134_m10.tm...
100% TM data processed  [ OK ]
[DI] Maximum code values in this TM file:
[DI] TDI offset -> 1005 ADUs (98% of code capacity). 1 ADU = 7.3314(TBD units)
[DI] Shape axis -> 63 ADUs (100% of code capacity). 1 ADU = 0.1270(TBD units)
[DI] Total flux -> 8388607 ADUs (100% of code capacity). 1 ADU = 1.0872e-
[DI] ASM sample -> 65535 ADUs (100% of code capacity). 1 ADU = 2.8992e-
[DI] Background -> 11 ADUs (8% of code capacity). 1 ADU = 1e-
[DI] StdAF flux -> 65535 ADUs (100% of code capacity). 1 ADU = 5.3407e-
[DI] AF 11 flux -> 65535 ADUs (100% of code capacity). 1 ADU = 5.3407e-
[DI] BBP flux   -> 65535 ADUs (100% of code capacity). 1 ADU = 5.3407e-
[BI] 54239459 overflows, 0 underflows.

[BI] Sampling scheme distribution:
[BI]    Window mode: Faint - 0%, Medium Bright - 100%, Bright - 0%
[BI]    AF11 mode: Normal - 100%, Narrow - 0%
[BI]    BBP mode: Normal - 79%, Narrow - 20%
[BI] Observation time:  1895.3508 hours.
[BI] Processing time:   44.0372 minutes.
[BI]    Processing speed ratio: 2582.3877 seconds/second.
[BI] Max. buffer size: 4.0000 KBytes.
[BI] 750752 sources in 704717 Data Sets
[BI] 375161 sources in Astro-1, 375591 sources in Astro-2.

In this simulation, Gaia has measured an average of 0.1100 sources/sec
```

Figure 10.1: Snapshot of the console output of txt2bin, after analyzing 79 days with a limiting magnitude of 10

As it can be seen, the simulator output contains information of the maximum values taken by several data fields, as well as their quantization step (indicated in Analog-to-Digital Units, ADUs). This information can be used for debugging purposes, e.g., to tune the code ranges for each data field. In our case, the maximum pixel and sample capacities indicated by the Gaia Parameter Database were taken, which are 190.000e^- for a pixel (ASM) and 350.000e^- for a sample (AF and BBP). The range for the Total Flux field was taken as 9.120.000e^-, this is, 48 ASM samples completely saturated. Should these ranges were wrong (or would eventually be updated in the future), the values in the XML file must be updated (as well as in the *.h* file, for *bin2txt*). In our case, these ranges (and ADU values) have been used in all of the simulations so the first part of the simulator output will be omitted in the next snapshots. Actually, this kind of output can be automatically omitted just indicating a lower verbose level: "`[BI]`" stands for "Basic Information" in the simulator output, while "`[DI]`" stands for "Detailed Information" and depends on a higher verbose level.

The rest of the simulator output is more scenario-dependant. It indicates the distribution of the sampling schemes, the total observation time and the total amount of sources and data sets. Also, simulation-related details are offered, such as the analysis (or conversion) speed and the

maximum buffer size used during the process. As it can be seen, since this simulation has a very low limiting magnitude, the buffer size was only 4 KB and about 43 minutes were analyzed every second.

Figures 10.2 and 10.3 show the most interesting results. Figure 10.2 shows the field density curves during these 79 days, using different integration times –as indicated in the figure legend. As one could expect, shorter integration times reveal large density peaks that cannot be detected with longer integration times. This is an interesting result, since it shows how these simulations can complement those already available (Lammers 2004) which use long integration periods. In this case, since the limiting V magnitude is only 10, very low star densities are obtained (ranging between zero and, at most, 1 star every 3.5 seconds). Nevertheless, the telemetry curve (figure 10.3) already shows peaks of about 3.5kbps (integrated during 2 minutes, in green), while averaged values only indicate about 2kbps.

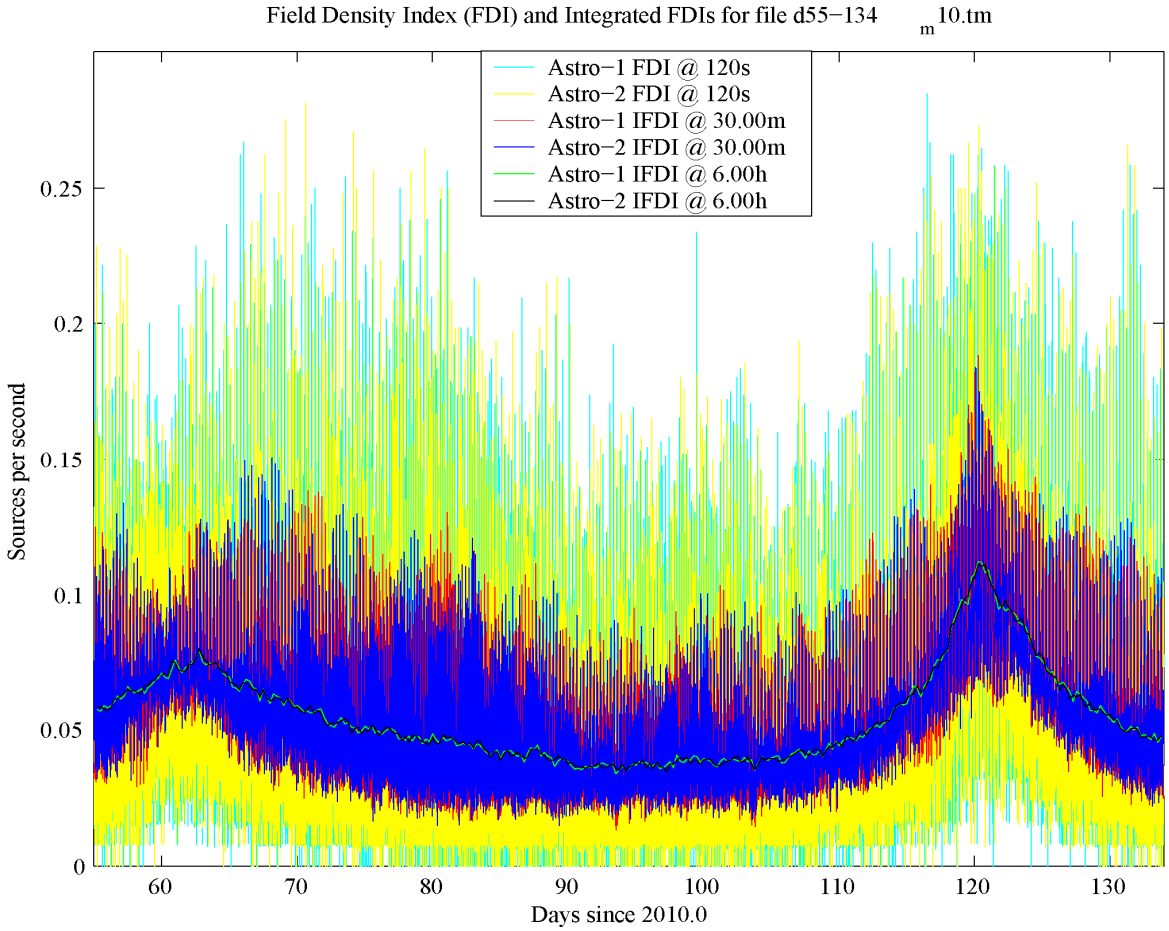


Figure 10.2: Star density curves from a 79-days test at 10th magnitude

The other interesting result that can be seen from an analysis of these figures is the evolution of the star density. As predicted by the SWG tool, around day 120 the density significantly increases, while around day 90 it is fairly low. Baade's window is not detected yet, since this low limiting magnitude excludes the typical star magnitudes that we can see there. Simulations with fainter magnitudes will reveal the exact situation of the window, while the other two typical cases (perpendicular and tangential scans of the GP) have already been verified. Finally, figure 10.4 illustrates the flux histogram obtained with this simulation. As it could be expected, a limiting magnitude of 10 implies that all of the peak fluxes saturate the pixels, and therefore all the occurrences are within the bin of the highest flux value (this is, 350.000e^-).

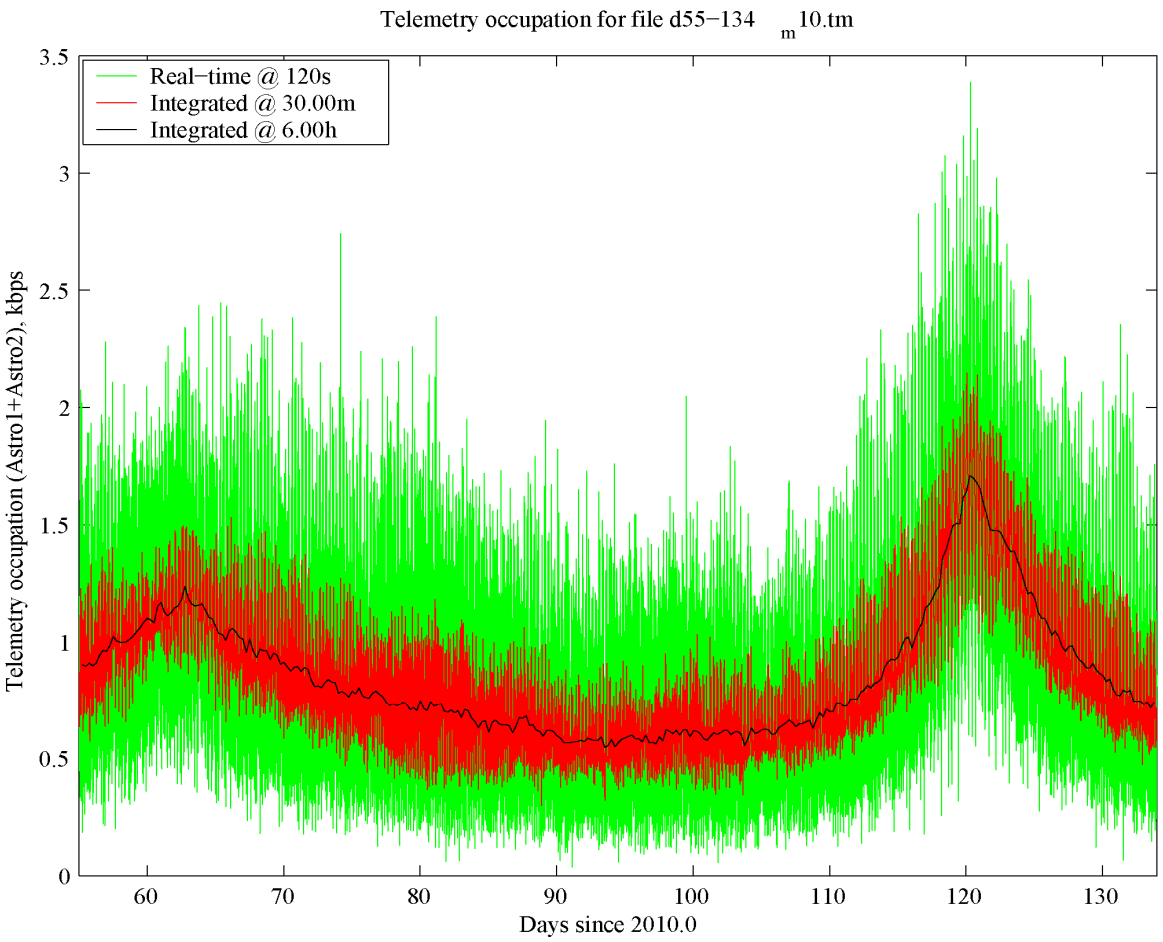


Figure 10.3: Telemetry curves from a 79-days test at 10th magnitude

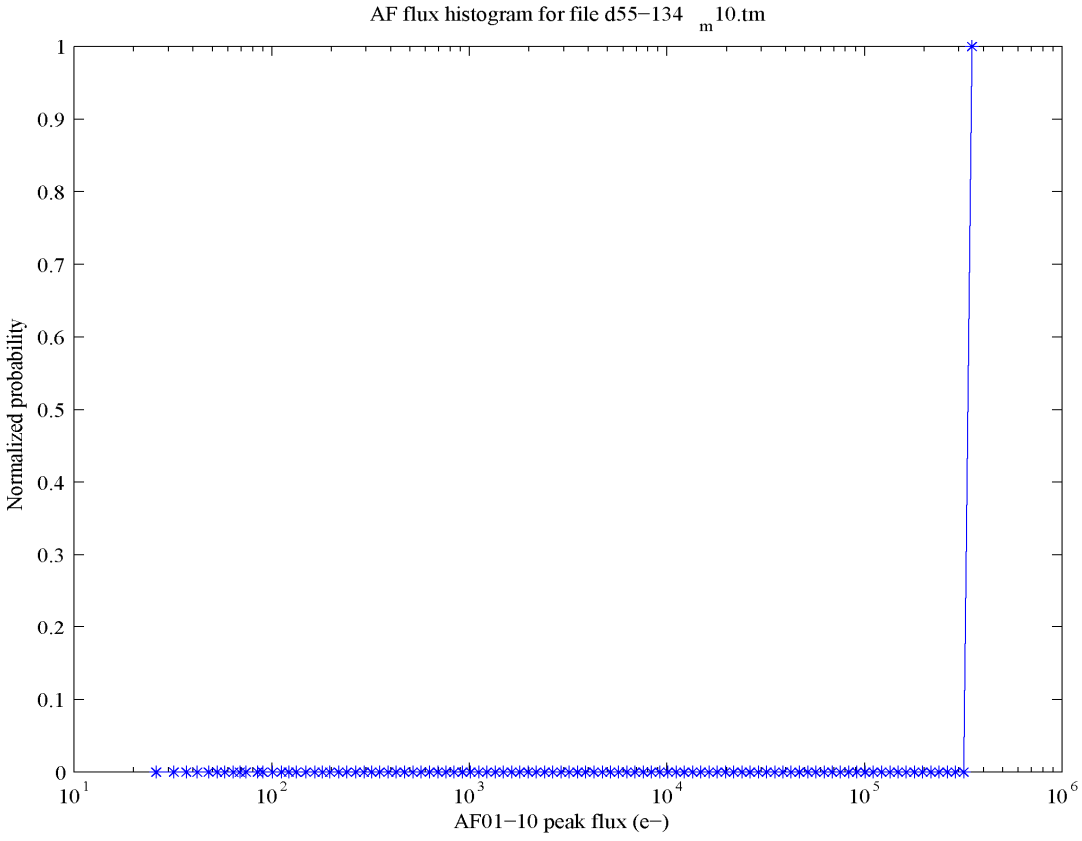


Figure 10.4: Histogram of the AF01-10 peak fluxes from the 79-days test at 10th magnitude

10.2.2. Magnitude 12 simulations: focusing on the targets

The nature of the GASS simulations (and of Gaia itself) reveals that even short periods of the mission will lead to huge file sizes when simulating up to magnitude 20. On the other hand, it would be desirable to simulate the adequate conditions for the forthcoming study on data compression, this is, to simulate a low-density area, a high-density area and an extremely dense area. The numerical determination of the star densities and telemetry rates under these extreme conditions will obviously be helpful as well. Therefore, we should know with a good precision the time intervals at which these events happen –i.e., crossing the GP perpendicularly or tangentially, and scanning the peak of Baade's window. For this, but also for studying the effect of different limiting magnitudes, we preferred to simulate intermediary limiting magnitudes before reaching the 20th magnitude. In this way, the "density peaks" and "valleys" will get clearer step by step. This subsection describes the results obtained up to 12th magnitude.

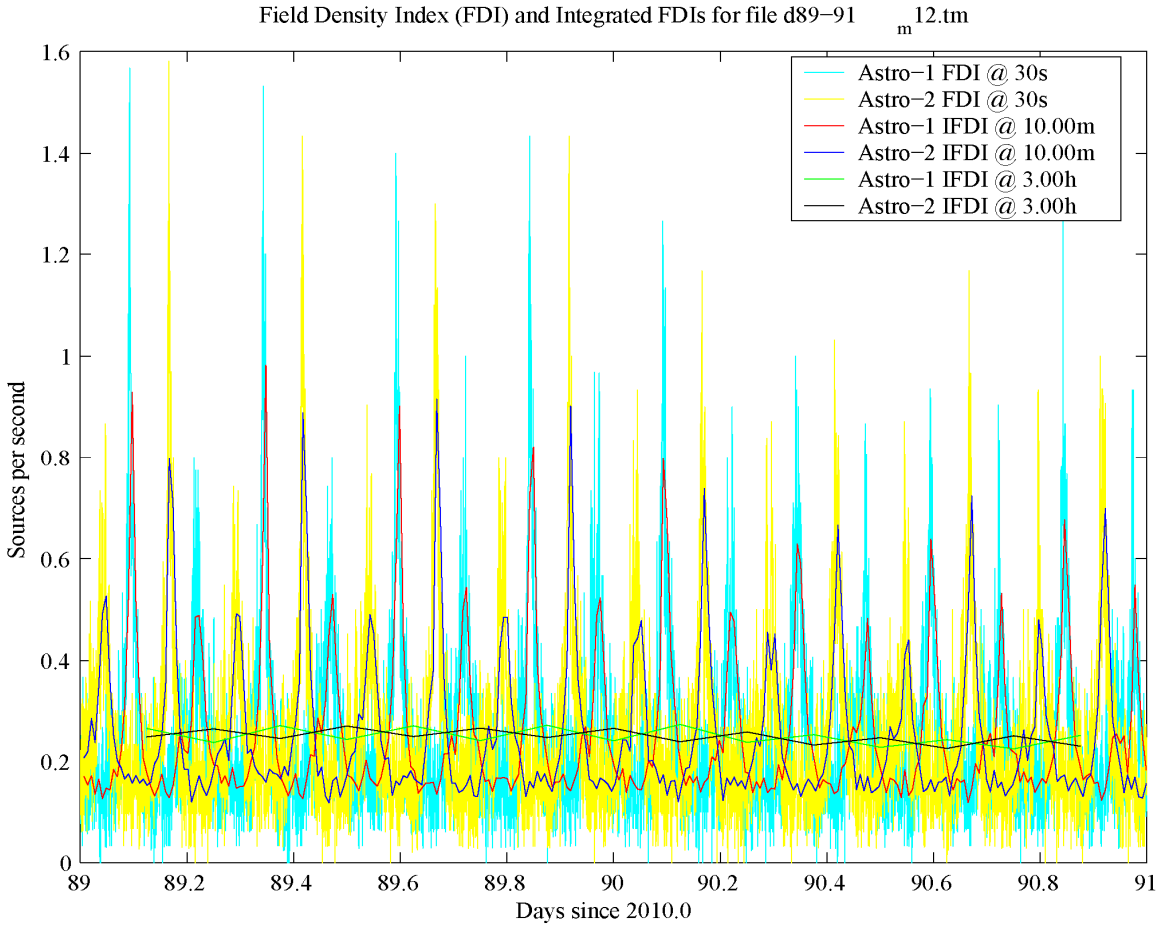


Figure 10.5: Star density curves from a 2-days test (8 GCs) at 12th mag. in a low density area

10.2.2.2. Low-density area

First of all we focused on days 89 to 91 which, presumably, should show an almost perpendicular scan of the GP. This should be reflected as a low average star density (and telemetry rate) but with large and rapid peaks (i.e., the crossing of the GP). Figures 10.5 and 10.6 illustrate how this situation has been successfully found. The star density is about 0.25 stars/s in average, this is, 1 star every 4 seconds, and the peaks (while crossing the GP) almost reach 1.6 stars/s, i.e., about 6 times the average density. In figure 10.5 we can clearly see the 8 Great Circles (GCs) scanned during those 2 days: 8 large peaks plus 8 lower peaks for every instrument. Also, day 89 reveal peaks slightly higher that day 90 (with which we can presume

that the GP is scanned more perpendicularly), so we will focus out efforts on this day when increasing the magnitude in the following subsections.

Figure 10.6, showing the telemetry rate curves during these two days, indicates an average TM rate of almost 4kbps with variations between 1 and 11kbps. These large variations confirm that the GP is scanned perpendicularly or almost perpendicularly. Observation periods around day 89.65 and 89.9 seem especially interesting because of the low telemetry rates, although day 89.15, 89.4 and 90.9 also offer interesting "valleys". We will study these areas with more detail. The dashed arrow in figure 10.6 indicates the lowest TM rate value when using a 10-minutes integration time.

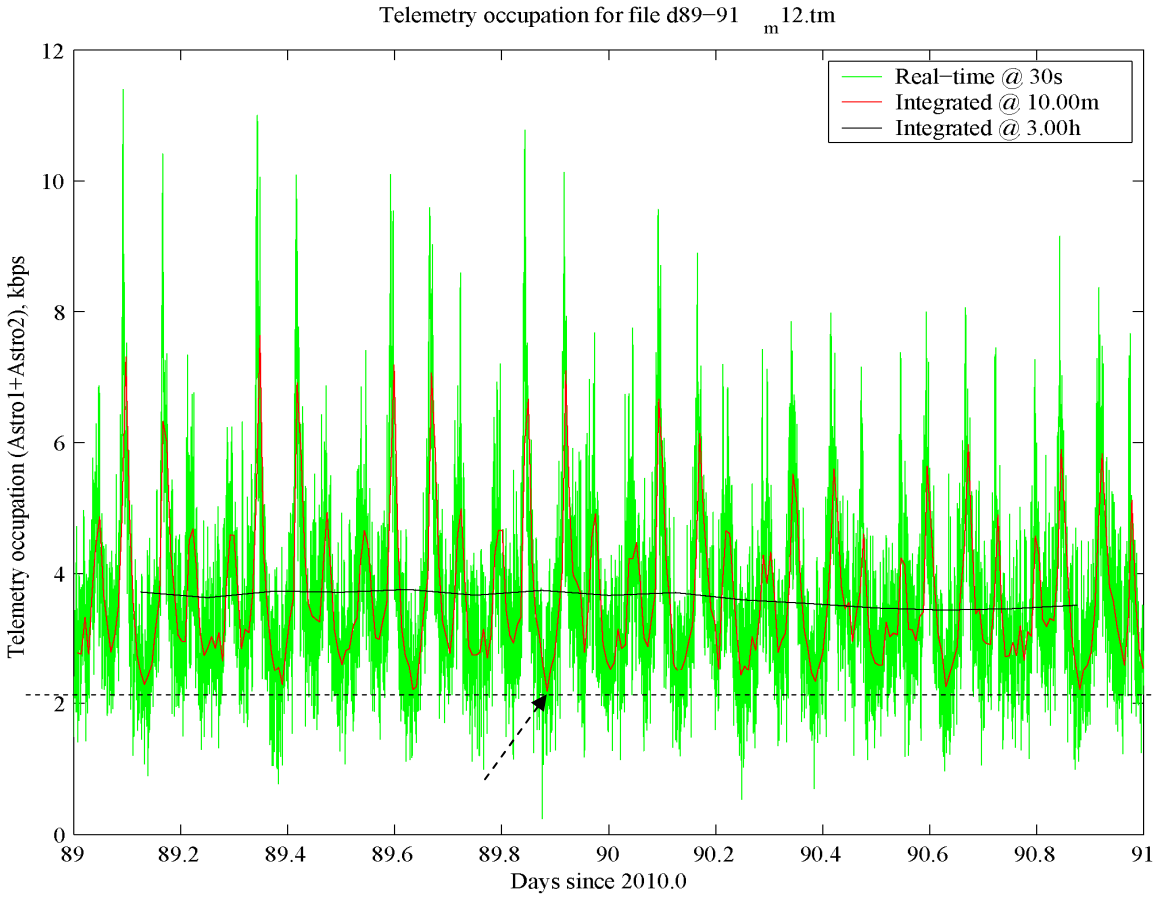


Figure 10.6: Telemetry curves from a 2-days test (8 GCs) at 12th mag. in a low density area

```
Analyzing /mnt/cdrom/d89-91_m12.tm...
100% TM data processed  [ OK ]
[BI] 2511294 overflows, 0 underflows.

[BI] Sampling scheme distribution:
[BI]    Window mode: Faint - 0%, Medium Bright - 100%, Bright - 0%
[BI]    AF11 mode: Normal - 100%, Narrow - 0%
[BI]    BBP mode: Normal - 80%, Narrow - 19%
[BI] Observation time:  47.9972 hours.
[BI] Processing time:   8.6266 minutes.
[BI]    Processing speed ratio: 333.8323 seconds/second.
[BI] Max. buffer size: 8.0000 KBytes.
[BI] 85804 sources in 66015 Data Sets
[BI] 42856 sources in Astro-1, 42948 sources in Astro-2.

In this simulation, Gaia has measured an average of 0.4966 sources/sec
```

Figure 10.7: Snapshot of the simulator output, after analyzing 2 days at 12th magnitude

Finally, figure 10.7 contains the snapshot of this simulation output, where it can be seen an increased memory buffer (8KB instead of 4KB), and a slightly higher average density than in the case of 10th magnitude, as expected. Although the relative processing time has increased, *txt2bin* is still able to analyze about 5 minutes per second of elapsed time.

10.2.2.3. High-density area

When focusing on days 119 to 121, during which Gaia should scan the GP almost tangentially, we also found an agreement between the simulation results and our predictions. Average star densities and telemetry rates are significantly higher, the GP crosses appear much wider, and the variations wrt average values are not so high. All of this can be seen in figures 10.8 and 10.9. In the first one we see an average star density of almost 1 star/s (4 times the average of days 89-91), with variations between 0.2 and 2.75 stars/s –so peak values are less than 3 times the average value. Here we can also see the 8 great circles, but now they appear much wider than before.

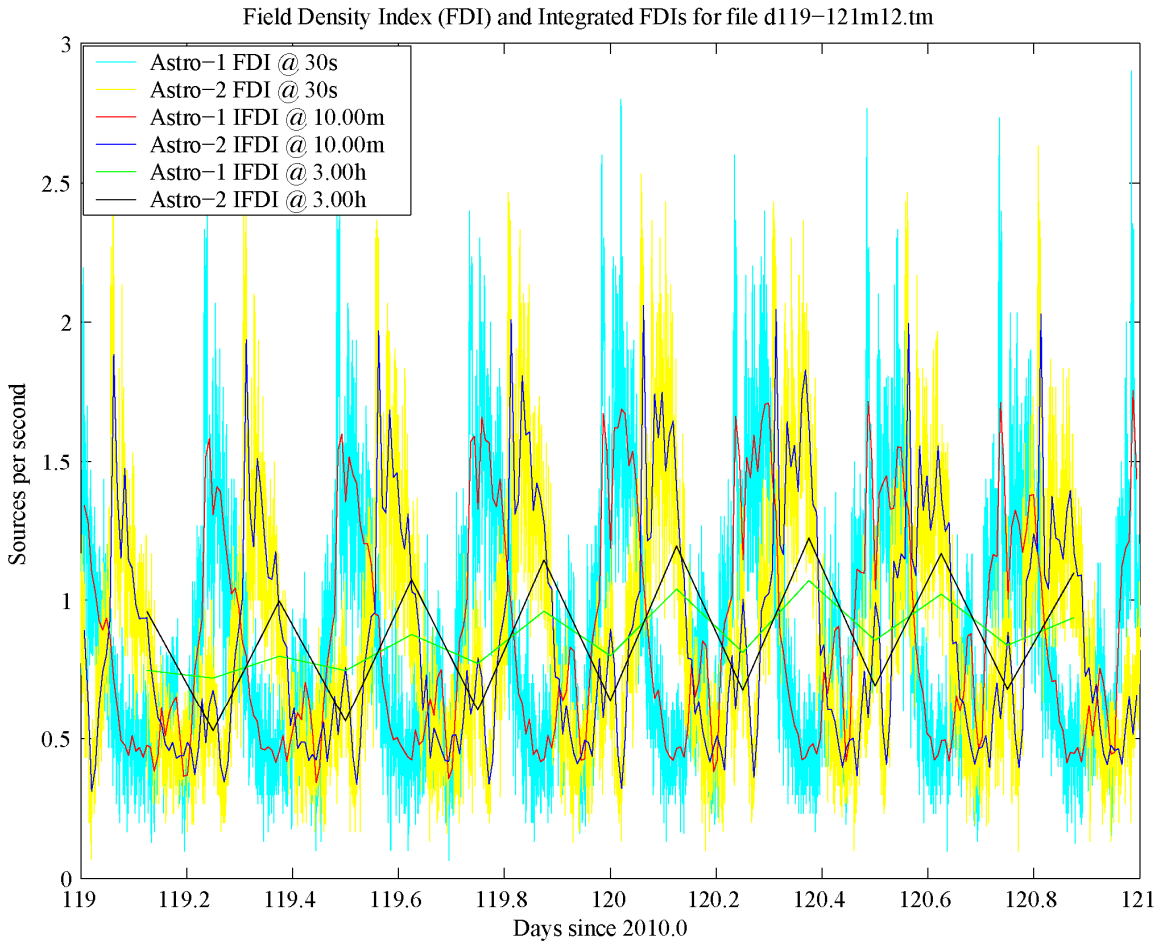


Figure 10.8: Star density curves from a 2-days test (8 GCs) at 12th mag. in a high density area

The TM rate curve shown in figure 10.9 offers yet a much clearer demonstration that we are scanning the GP almost tangentially. The 8 great circles scanned during these 2 days appear really clear, and the average TM rate is more than 10kbps –2.5 times higher than before. This average value shows a clear curve with a peak about day 120.2 to 120.4, so if we want high telemetry rates we should simulate this period. The TM rates vary from 3kbps to 23kbps, which is a smaller relative variation than before. Finally, figure 10.10 shows the flux histogram for this simulation. As can be seen there not all the samples are saturated, since a small curve from about 220.000e⁻ towards saturation (350.000e⁻) appears. Simulator output indicates an average of 1.73 stars/s (we recall that it is only 12th magnitude), a maximum buffer size of 16KB and a processing time of 83 seconds per second.

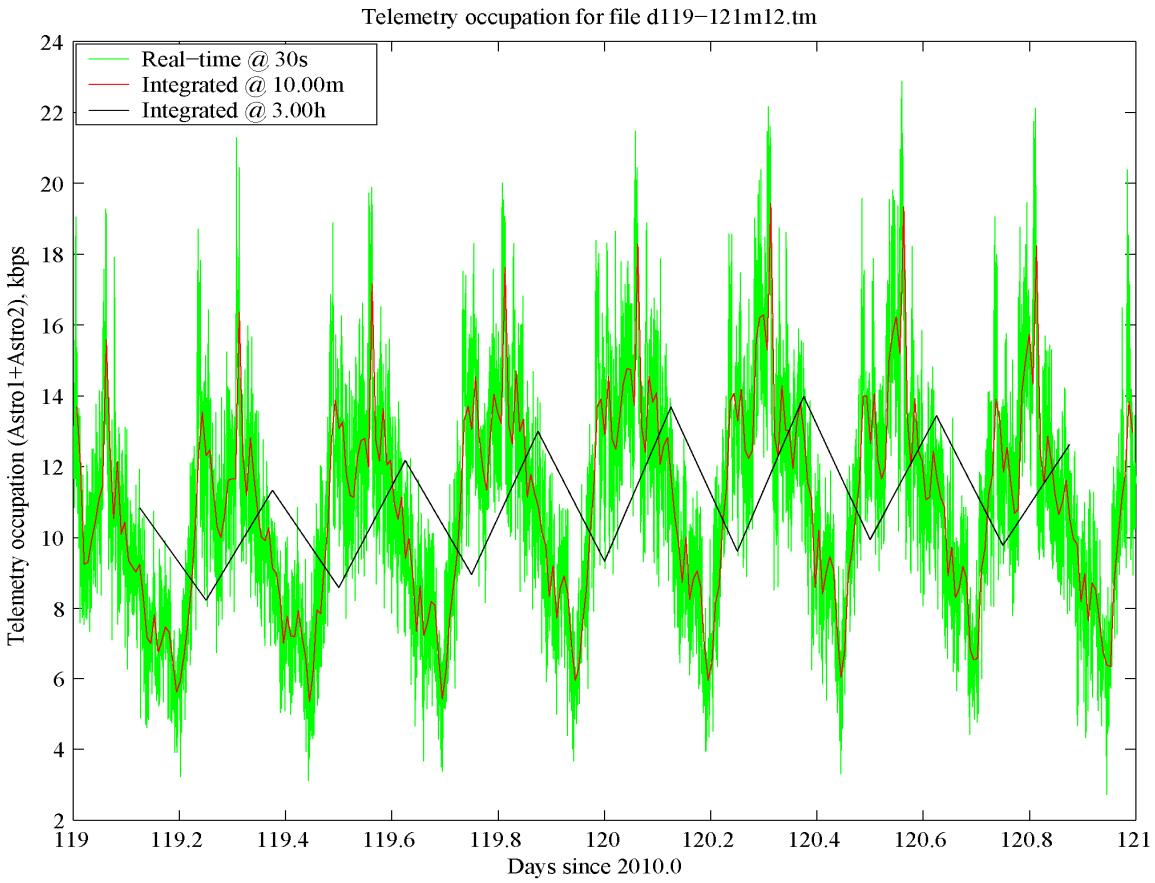


Figure 10.9: Telemetry curves from a 2-days test (8 GCs) at 12th mag. in a high density area

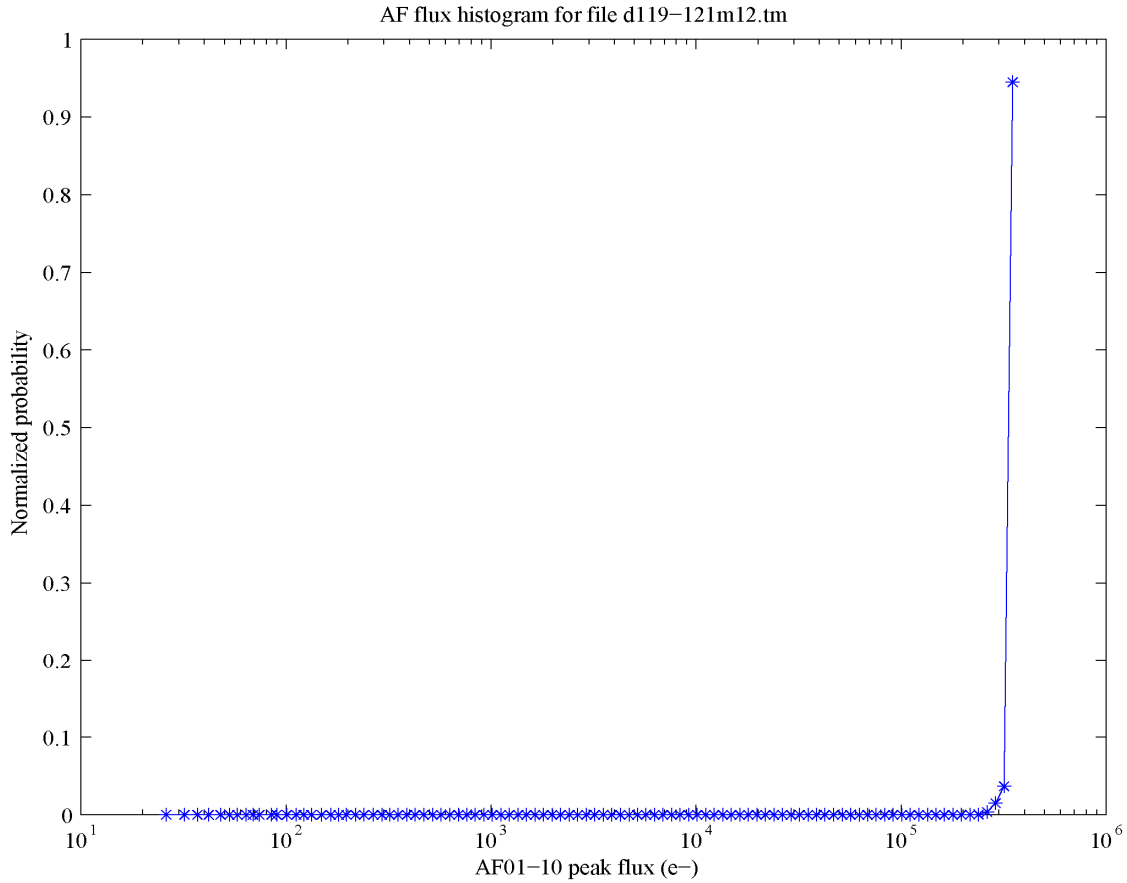


Figure 10.10: Histogram of the AF01-10 peak fluxes resulting from the 12th magnitude tests

10.2.3. Magnitude 16 simulations: looking for Baade's window

During the 12th magnitude simulations we did not try day 113 yet, since that magnitude would surely not include most of the stars seen in Baade's window. But now it is time to test that area, trying to determine a precise period where we could find a spectacular peak in the star density. Also, of course, we will improve the precision of the other two scenarios of interest: around day 90 and around day 120.

10.2.3.1. Low-density area

Since a low-density area such as day 90 still makes possible long GASS simulations at 16th magnitude without needing prohibitive execution times or file sizes, we preferred obtaining rich overviews of the area rather than focusing on short periods. This is the reason why we tried simulating more than 1 day, more specifically from day 89 to 90.35. Nevertheless, we started reaching our limits on file size and memory, so this simulation finally ended up at day 90. Anyway the simulation offered interesting results after being analyzed with *txt2bin*, as figures 10.11 and 10.12 illustrate. Figure 10.11 clearly shows again the several great circles scanned during day 89, and the low average star density (and large, rapid density peaks) that we can find in there.

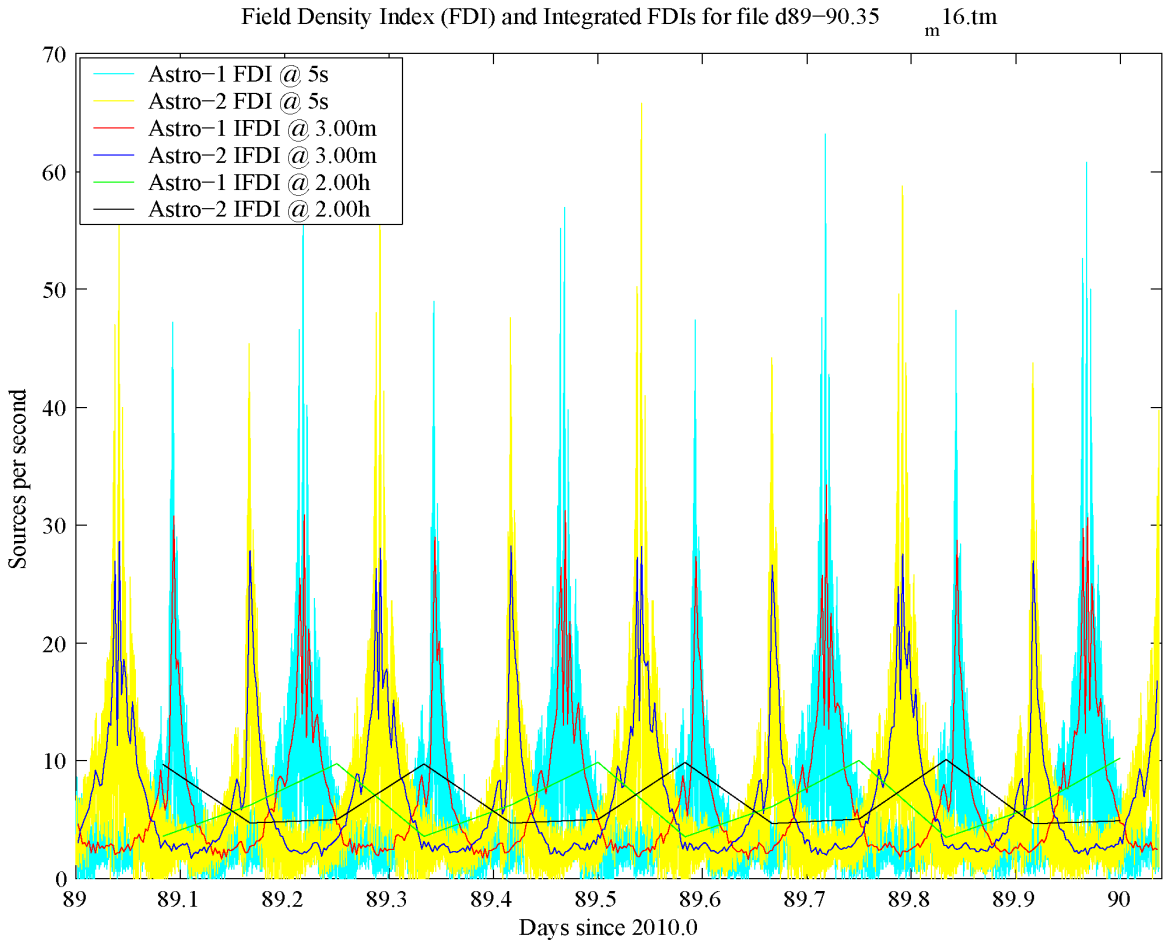


Figure 10.11: Star density curves at 16th magnitude in a low density area

The average star density that we obtained is about 7 stars/s per instrument, while the peaks reach about 65 stars/s per instrument. Figure 10.12, showing the telemetry curves, indicates an average of about 70kbps, with peaks up to 300kbps. It is worth noting at this point the importance of integration times, since TM rates averaged during 3 minutes reveal peaks of only 150kbps, while these peaks of 300kbps are shown when integrating during only 5 seconds. This difference should be taken into account when sizing the data transmission

interfaces of the payload. Finally, figure 10.13 contains a snapshot of the simulator output, where it can be seen an average star density of about 13 sources per second in this area (taking into account both Astro instruments).

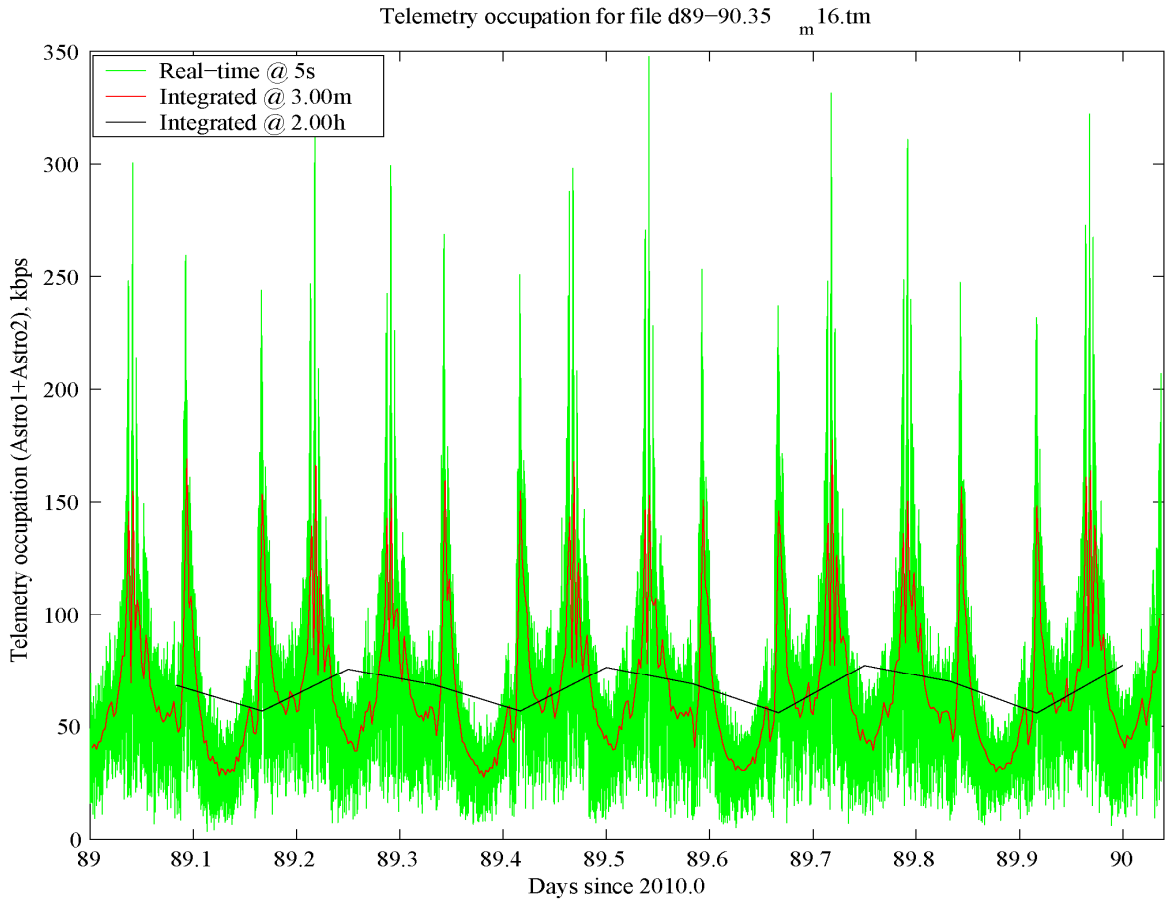


Figure 10.12: TM curves from a 16th magnitude simulation in a low density area

```
Analyzing /mnt/c_win/GASS SIMS/v2.2/tm/d89-90.35_m16.tm...
100% TM data processed  [ OK ]
[BI] 1096239 overflows, 0 underflows.

[BI] Sampling scheme distribution:
[BI]    Window mode: Faint - 0%, Medium Bright - 100%, Bright - 0%
[BI]    AF11 mode: Normal - 100%, Narrow - 0%
[BI]    BBP mode: Normal - 80%, Narrow - 19%
[BI] Observation time:  24.8953 hours.
[BI] Processing time:   42.6974 minutes.
[BI]    Processing speed ratio: 34.9838 seconds/second.
[BI] Max. buffer size: 64.0000 KBytes.
[BI] 1166330 sources in 87396 Data Sets
[BI] 575515 sources in Astro-1, 590815 sources in Astro-2.

In this simulation, Gaia has measured an average of 13.0137 sources/sec
```

Figure 10.13: Snapshot of the simulator result after analyzing day 89-90 at 16th magnitude

In order to refine our observation targets, and taking into account the relative easiness of simulating this area at this magnitude, we also generated two great circles in the two days being studied: one simulation covers day 89.3 to 89.55, and the other one covers day 90.3 to 90.55. The results are shown in figure 10.14 and 10.15, each containing the star densities and TM curves of the two simulations. Since the axes are difficult to read, the original plots have been rescaled in order to use the same vertical scale. In this way, we can see how day 89 is slightly “emptier” than day 90 and generates less telemetry, so in our analyses of a low-density area (or “optimistic case”) we will restrict to day 89.

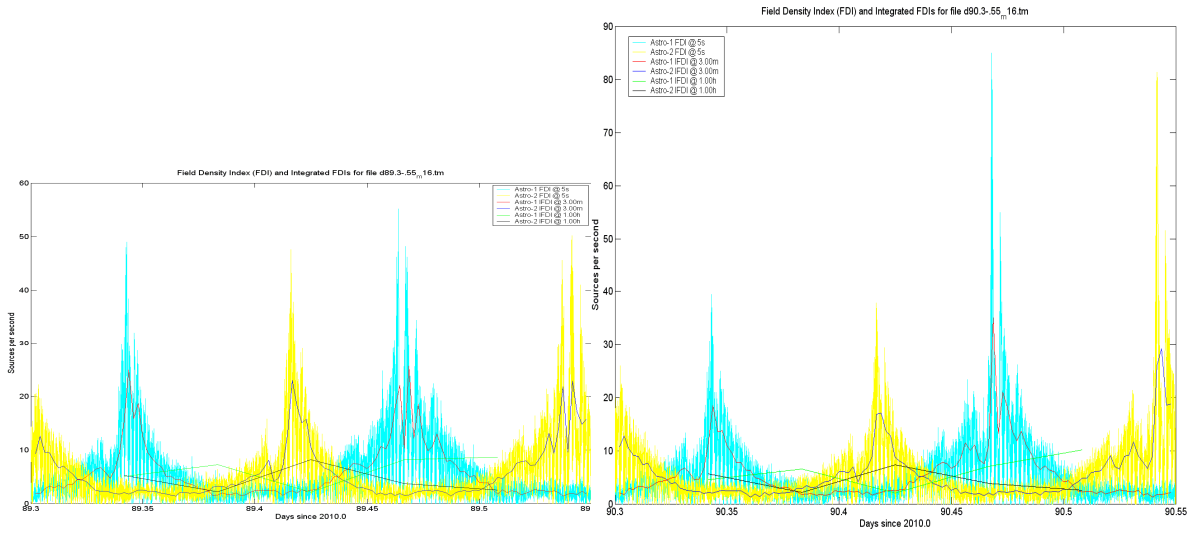


Figure 10.14: Field Density curves of one GC at 16th magnitude in day 89 (left panel) and 90 (right panel)

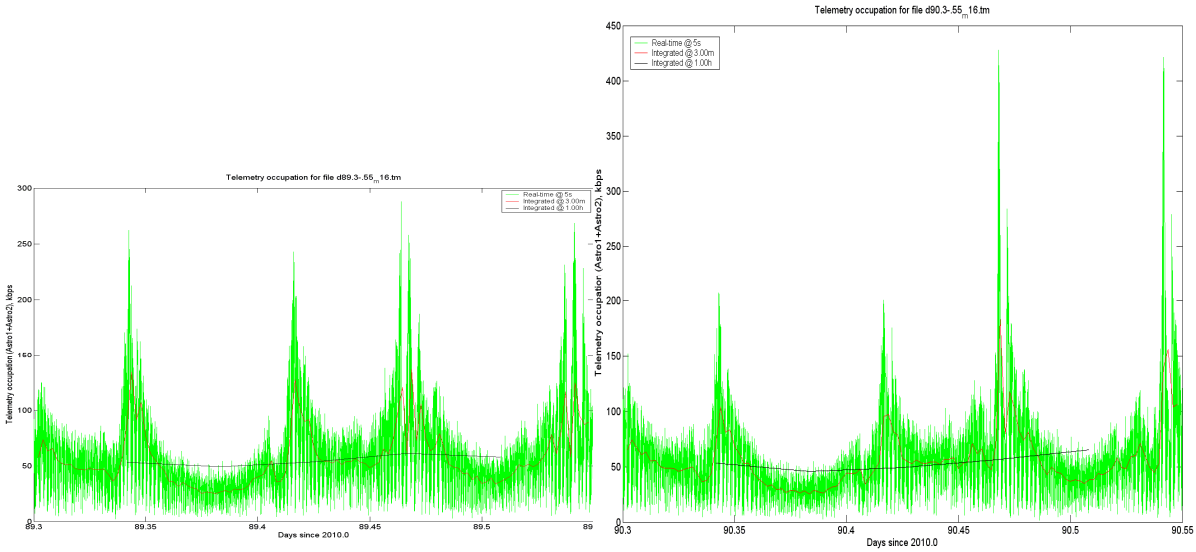


Figure 10.15: TM curves of one GC at 16th mag. in day 89 (left panel) and 90 (right panel)

Finally, we show in figure 10.16 the AF flux histogram obtained during the 1-day simulation. As it can be seen there, the peak has significantly moved towards the left, this is, towards lower fluxes (i.e., fainter stars, as expected). More specifically, the peak is about 14.000e⁻, while at 6.300e⁻ it stars containing some occurrences. We must note the peak at saturation level (on the right end of the histogram). The reason for this peak is that GASS v2.2 does not use the full sampling scheme yet, and more specifically the bright star windows (WYnn in Høg 2002b) are not included. Therefore, bright stars imply many saturated samples, which appear as a peak in the corresponding bin of the histogram. We will see in the 20th magnitude histograms how this peak significantly decreases, since the amount of faint samples will be extremely higher than the amount of saturated samples.

10.2.3.2. High-density area

While a low-density area such as days 89 and 90 makes possible the simulation of long periods of time even at 16th magnitude, moving to a high-density area such as days 119 or 120 rapidly increases the resulting simulation times and file sizes. Despite of this, we wanted to obtain a simulation as long as possible at this magnitude. Since at 12th magnitude we found out a density increase from day 119 to day 120, we preferred simulating the first one and, therefore, obtain a longer period to study. Fig. 10.17 shows the simulator output after analyzing this area,

where a spectacular increase (up to 132 stars/s, 10 times the density in day 89) in the average star density can be seen. The memory buffer has obviously increased as well, and the software now analyzes less than 4 seconds per second. We must note that the multi-window sampling scheme appears for the first time: GASS samples stars fainter than 16th magnitude with a different window mode, and here the report reveals a small use (less than 1%) of such windows. The BBP windows keep the 80% / 20% relation, basically for calibration purposes.

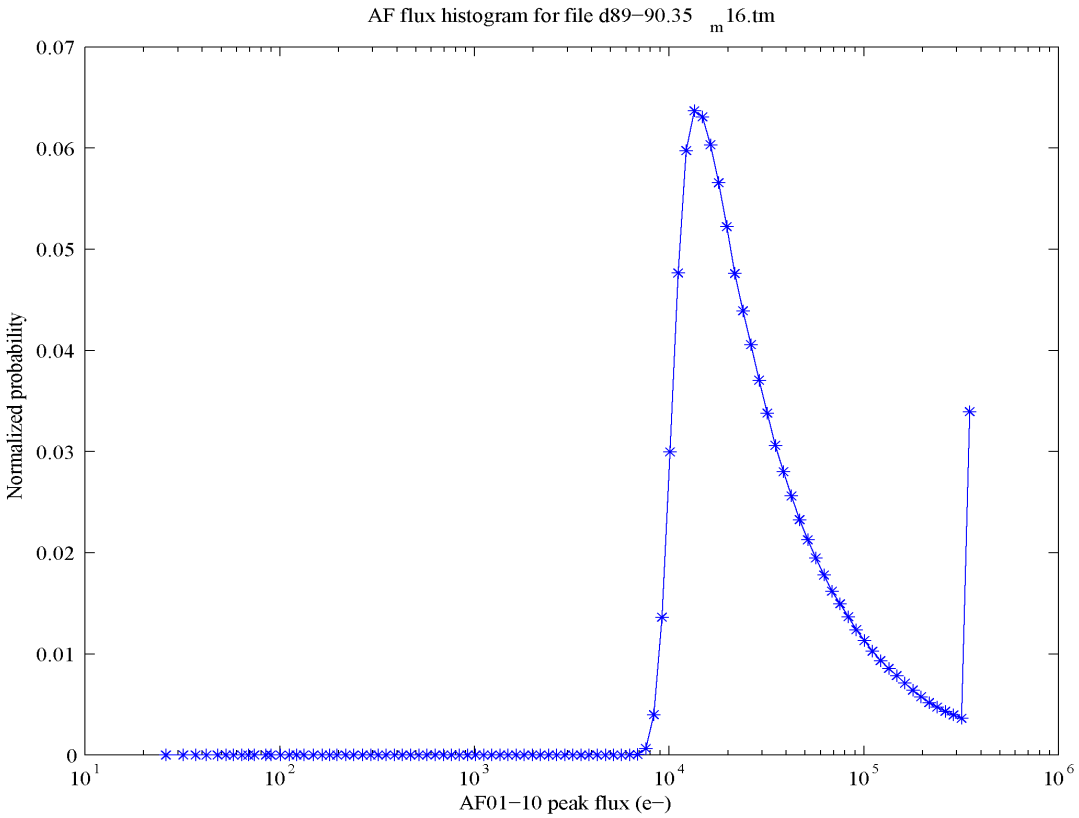


Figure 10.16: Flux histogram of the 16th magnitude simulation in a low-density area

As figure 10.18 (and also the simulator output) clearly reveals, this simulation finally lasts only 95 minutes. After this time, we started reaching file and memory limits. Despite of this short period, we can see in the density curves of figure 10.18 a scan of the galactic plane about day 119.06, where the star density increases even more, up to some 400 stars/s in Astro-2. It must be noted that the averaged star density for this instrument is only about 150 stars/s (2.6 times lower), so these differences must be taken into account when sizing the on-board hardware requirements.

```
Analyzing /mnt/c_win/GASS SIMS/v2.2/tm/d119.0-95min_m16.tm...
100% TM data processed  [ OK ]
[BI] 289971 overflows, 0 underflows.

[BI] Sampling scheme distribution:
[BI]    Window mode: Faint - 0%, Medium Bright - 99%, Bright - 0%
[BI]    AF11 mode: Normal - 100%, Narrow - 0%
[BI]    BBP mode: Normal - 79%, Narrow - 20%
[BI] Observation time:  1.5950 hours.
[BI] Processing time:   25.7222 minutes.
[BI]    Processing speed ratio: 3.7205 seconds/second.
[BI] Max. buffer size: 512.0000 KBytes.
[BI] 761145 sources in 5742 Data Sets
[BI] 260704 sources in Astro-1, 500441 sources in Astro-2.

In this simulation, Gaia has measured an average of 132.5575 sources/sec
```

Figure 10.17: txt2bin output after analyzing a dense 16th magnitude area

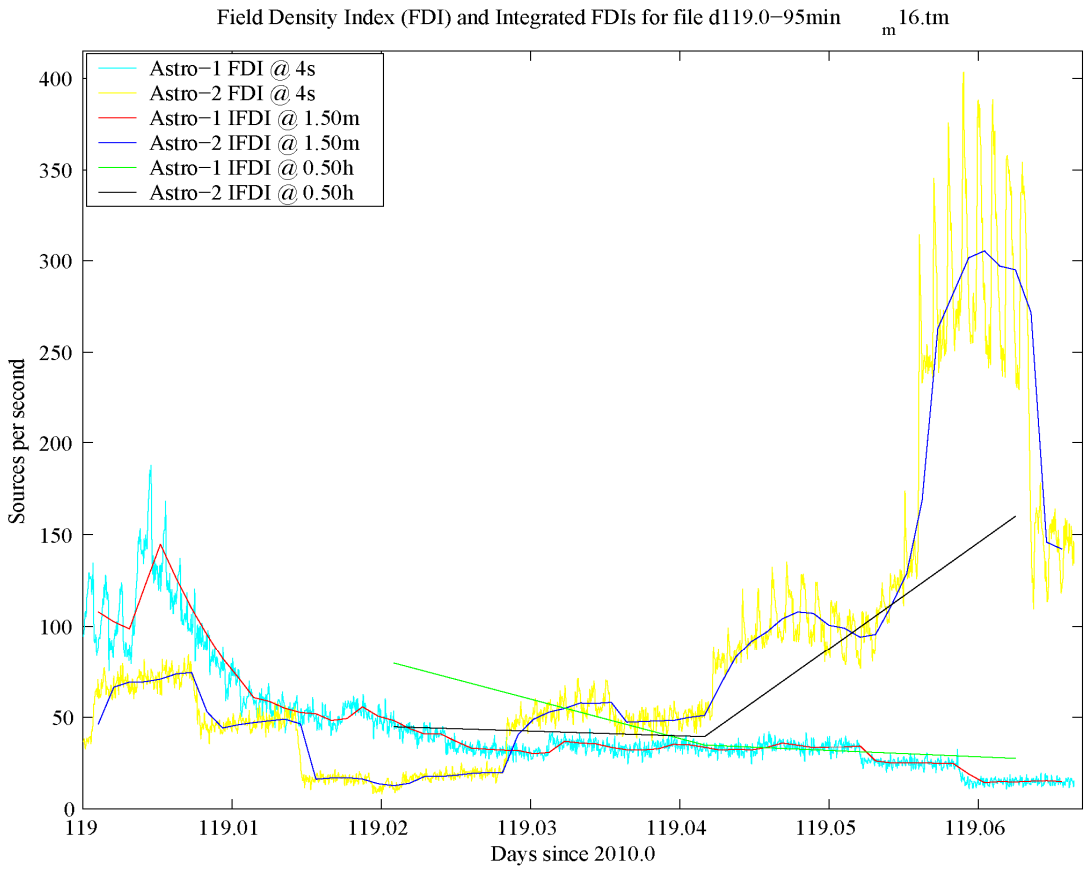


Figure 10.18: Star density curves at 16th magnitude in a dense area

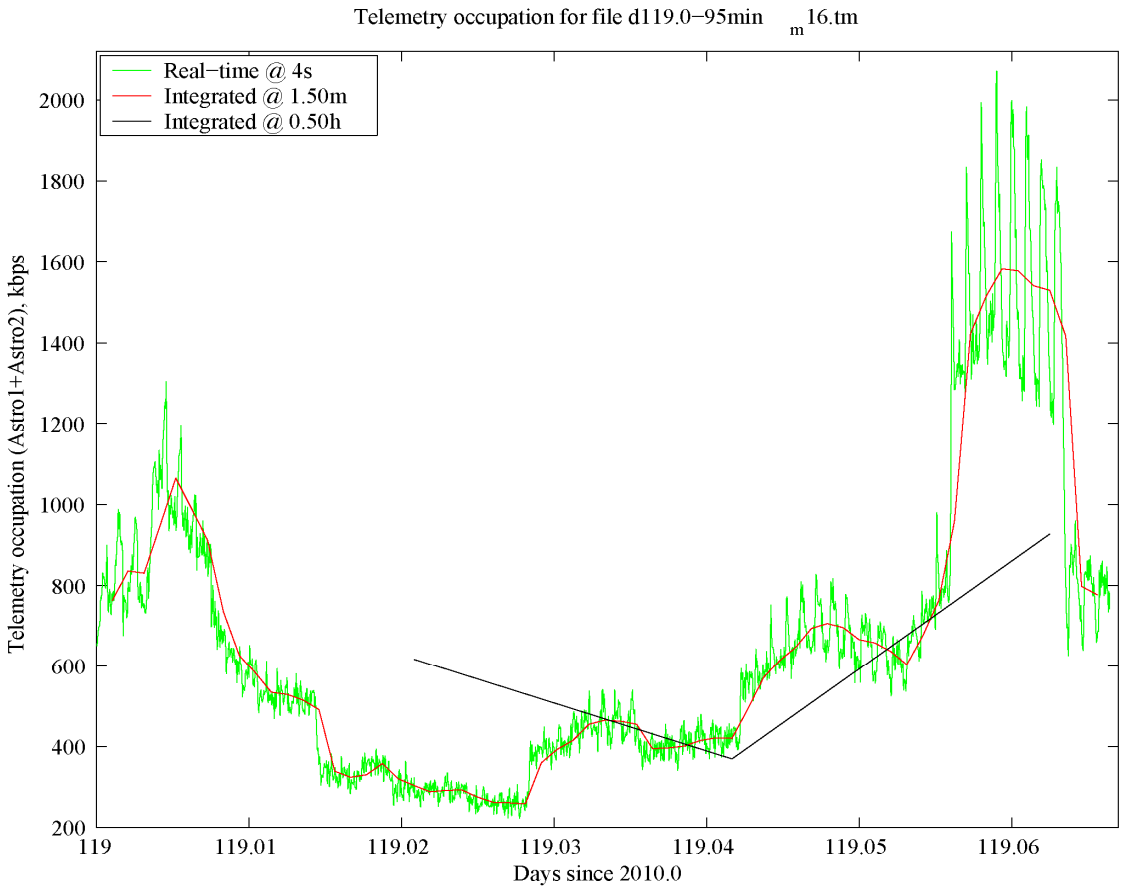


Figure 10.19: TM rate curves at 16th magnitude in a dense area

We also show the TM curves in this area in figure 10.19, where peaks of 2Mbps (with only 4 seconds of integration time) can be seen, while values averaged during 30 minutes are of only 900kbps. We must emphasize these high values in the telemetry rates, even taking into account that we are limiting our simulations to magnitude 16. The variations are also spectacular, since only 300kbps are generated with only 45 minutes of difference. It is curious to see that this 300kbps "floor" is similar to the highest value that we found in the 89th-day area.

Finally, figure 20 illustrates the flux histogram for this area. When compared to figure 16, we can verify the reason of the peak at saturation level: this area is much more crowded than before, and therefore the amount of faint stars (leading to the expected histogram) is much higher than the amount of saturated pixels. Therefore, the peak at saturation level significantly decreases.

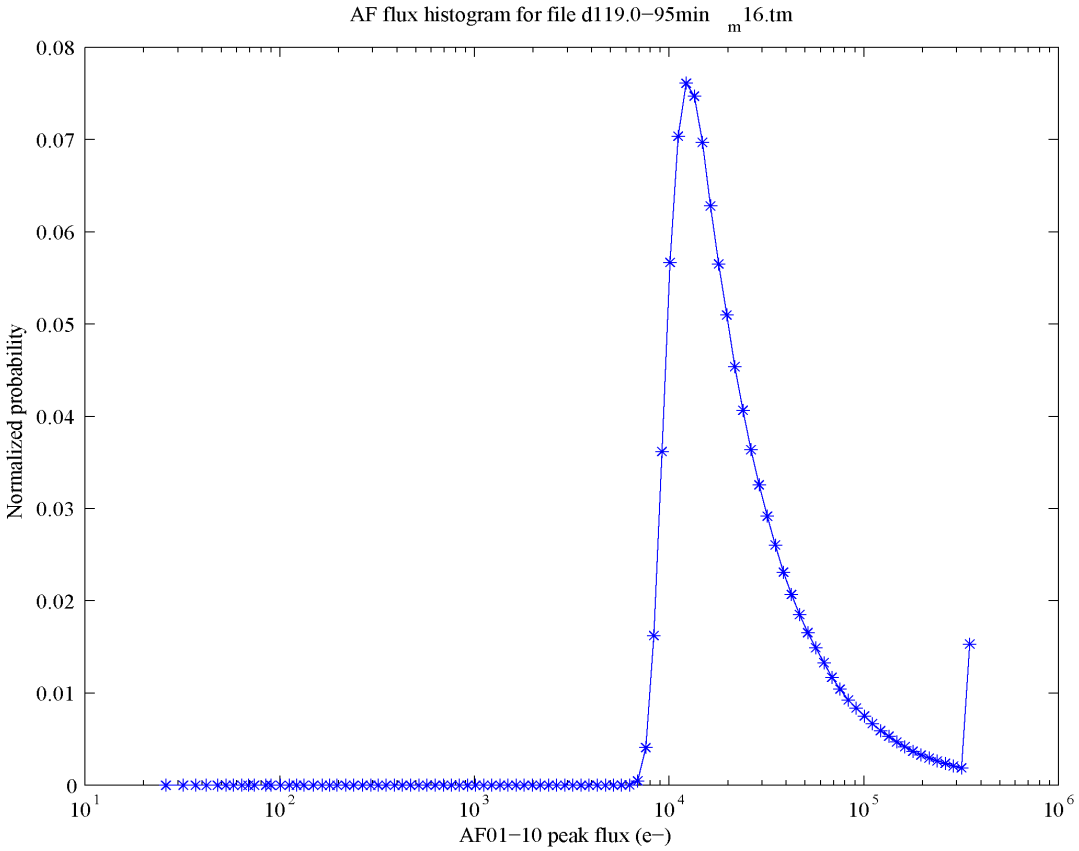


Figure 10.20: Flux histogram at 16th magnitude in a dense area

10.2.3.3. Baade's window

Given this faint limiting magnitude, we are finally able to see the peak at Baade's window. For this, we generated about 3 hours (half a GC) of telemetry starting at day 113, and the result can be seen in figures 10.21 and 10.22. There we can see an average density of about 10 stars/s, and an impressive peak of 700 stars/s when crossing the central area of Baade's window, observed by Astro-2 about day 113.057. Just as a curiosity, the small blue peak about day 113.115 is the galactic plane being scanned by Astro-1, reaching at most 30 stars/s.

The amount of telemetry data in this area is illustrated in figure 10.22, where a peak of 3.5Mbps can be seen, while the average is 50 to 100kbps. The small peak corresponding to the GP scan by Astro-1 is about 150kbps. We note at this point that the difference of the TM rate wrt those found in the dense area of day 119 is not so high. This, in turn, implies that we can safely assume that many stars are still being undetected due to the limiting magnitude. The next analyses at full (20th) magnitude should reveal the real densities and TM rates of such a crowded field.

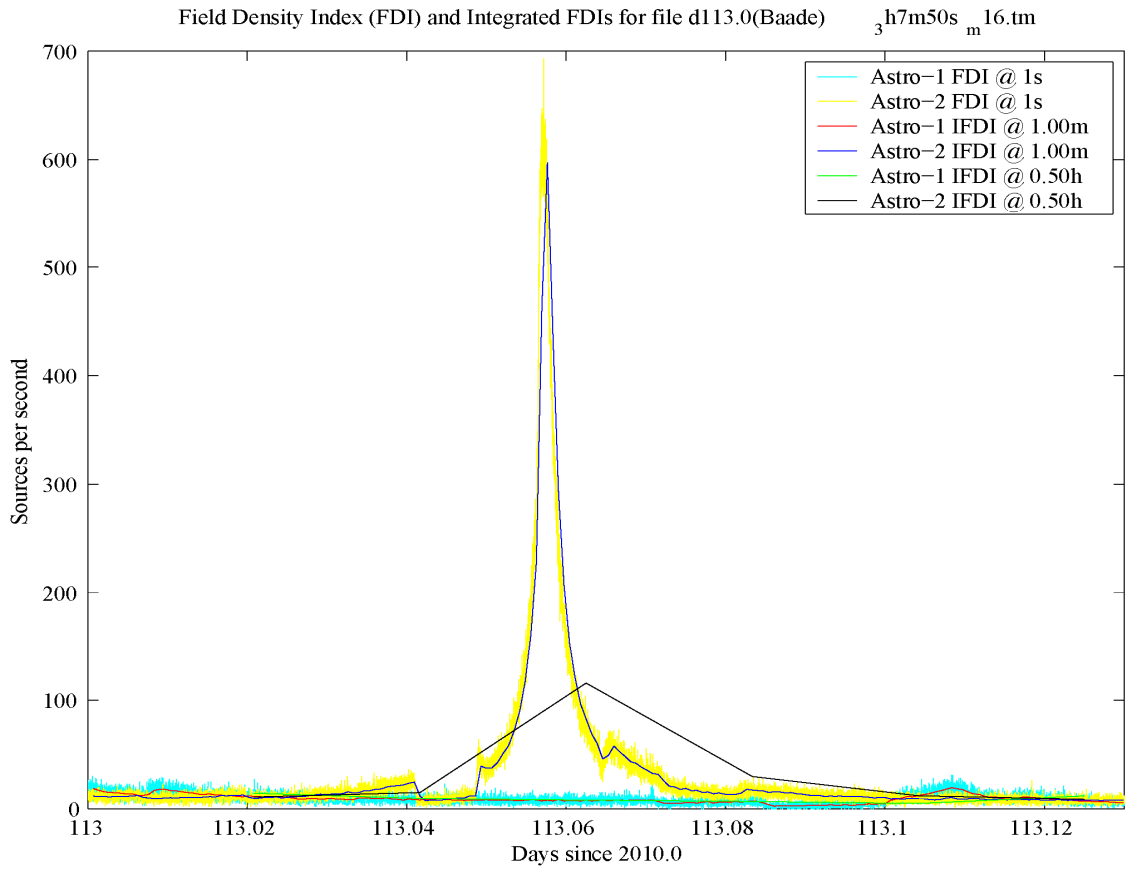


Figure 10.21: Star density in the Baade's window at 16th magnitude

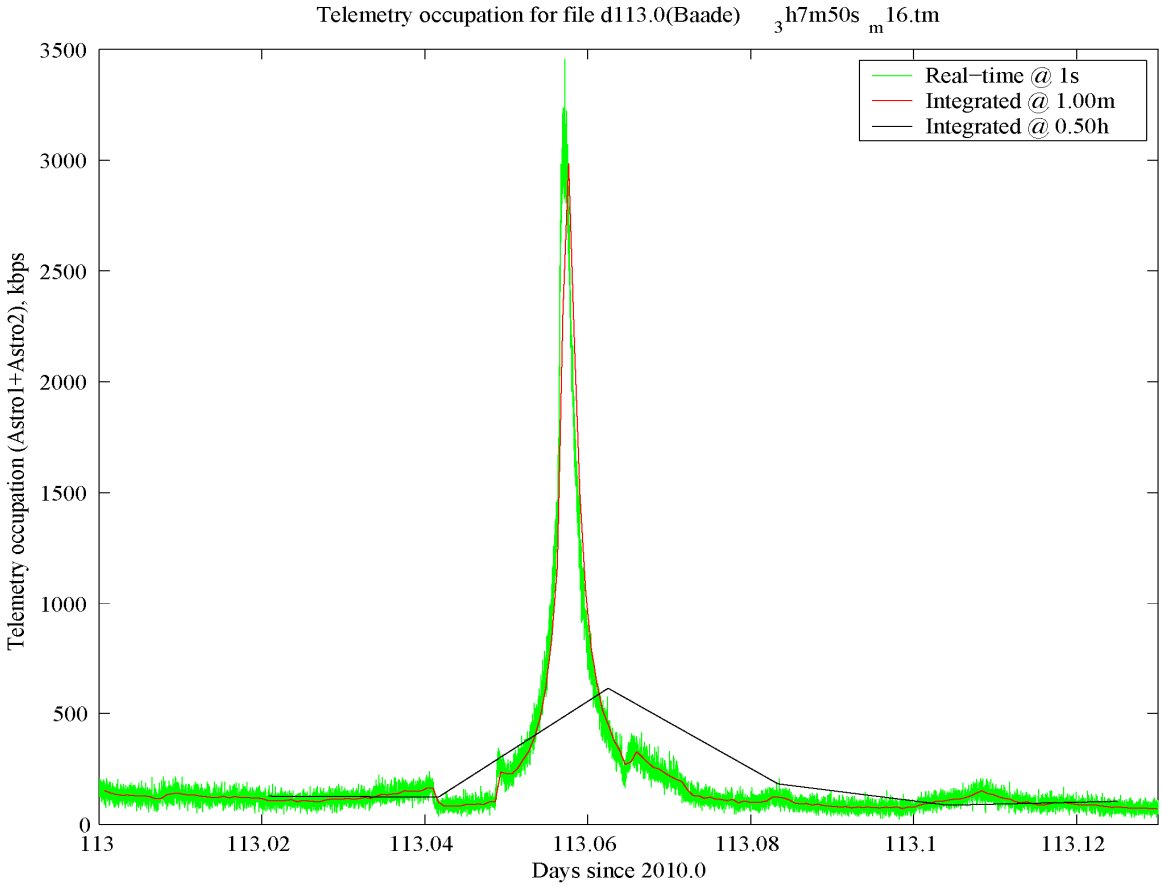


Figure 10.22: TM rate curve at the Baade's window at 16th magnitude

10.2.4. Magnitude 20 simulations: the real case

This last subsection will show the results of the simulations at 20th magnitude, for which we have also obtained the *.bin* (binary) files required not only for data compression studies, but also for verifying the decoding process.

10.2.4.1. Low-density area

First of all, we are going to simulate a typical region, with a low star density but with large and rapid density peaks. This will be the most optimistic case, and despite of this it will reveal the large stress under which Gaia will operate (due to large variations in the observation conditions). After studying the simulations at 12th and 16th magnitudes we have selected the last great circle of day 89, this is, day 89.75 to 90.0. During this GC very low star densities will be observed, which will be interesting in order to determine a "floor" for the data compression ratio –since preliminary studies reveal that the higher the star density, the best the data compression can be. Also, the average star density seems really low during this period, so this will be an interesting case for being considered as "the best case". In other words, Gaia must be absolutely capable of measuring this area to completeness; else, the problems in other sky areas would be unacceptable.

```
Converting data/tm/d89.75-.8_m20.tm to data/bin/d89.75-.8_m20.tm.bin...
100% TM data processed  [ OK ]
[BI] 51557 overflows, 0 underflows.

[BI] Sampling scheme distribution:
[BI]    Window mode: Faint - 95%, Medium Bright - 4%, Bright - 0%
[BI]    AF11 mode: Normal - 100%, Narrow - 0%
[BI]    BBP mode: Normal - 79%, Narrow - 20%
[BI] Observation time:  1.0236 hours.
[BI] Processing time:   1.1589 hours.
[BI]    Processing speed ratio: 0.8833 seconds/second.
[BI] 1.6185 Mbps written in average.
[BI] Max. buffer size: 4096.0000 KBytes.
[BI] 2177206 sources in 3685 Data Sets
[BI] 79310 sources in Astro-1, 2097896 sources in Astro-2.

In this simulation, Gaia has measured an average of 590.8293 sources/sec
Average data rate: 1.8324 Mbps.
```

```
Converting data/tm/d89.875-90.0_m20.tm to data/bin/d89.875-90.0_m20.tm.bin...
100% TM data processed  [ OK ]
[BI] 119745 overflows, 0 underflows.

[BI] Sampling scheme distribution:
[BI]    Window mode: Faint - 89%, Medium Bright - 10%, Bright - 0%
[BI]    AF11 mode: Normal - 100%, Narrow - 0%
[BI]    BBP mode: Normal - 79%, Narrow - 20%
[BI] Observation time:  2.1181 hours.
[BI] Processing time:   47.6642 minutes.
[BI]    Processing speed ratio: 2.6662 seconds/second.
[BI] 1.7706 Mbps written in average.
[BI] Max. buffer size: 2048.0000 KBytes.
[BI] 1582807 sources in 7625 Data Sets
[BI] 1001215 sources in Astro-1, 581592 sources in Astro-2.

In this simulation, Gaia has measured an average of 207.5813 sources/sec
Average data rate: 0.6641 Mbps.
```

Figure 10.23: txt2bin output after converting the 1st half (top panel) and 2nd half (bottom panel) of a low-density GC at 20th magnitude

Our first intention was to execute GASS for days 89.75-89-875 and 89.875-90.0, this is, breaking the GC down to 2 parts. Unfortunately the GASS simulator ran out of memory in both of these parts for reasons related to the HTM level. Our original selection was an HTM level of 9, but we verified afterwards that a level of 10 would have been enough for this case.

As a result, only 53 minutes in the first part and about 2 hours in the second part were generated. Despite of this, the results obtained with these simulations (and the binary files generated) will be really useful. Figure 10.23 shows the *txt2bin* outputs for these files, where it can be seen that only 2.6 seconds per second can be analyzed (and converted) for the best of the cases. The data buffer is now realistic, reaching up to 4MB, and the average star densities also correspond to more typical cases when observing at 20th magnitude. Finally, here we can see the real distribution of the sampling scheme: between 89% and 95% of stars are samples with the "faint star" window mode, which corresponds to the amount of stars fainter than 16th magnitude.

In figures 10.24 to 10.27 we show the resulting star density curves and TM curves for these areas. Figure 10.24 reveals Astro-2 as the most interesting instrument here, which starts with a very low density (about 30 stars/s) and reaches a peak of 3775 stars/s (after which the GASS simulator ran out of memory, with which we can presume that the real peak would be even higher than this). Astro-1 instrument keeps around a density of some 20 stars/s. All in all, this results in a telemetry rate from 200kbps to 12.6Mbps (figure 10.25). We note here again the importance of the integration times: when compared to "real-time" (1-second integration time), integrating during even only 1 minute implies a star density decrease of about 25% (from 3775 to 2858 stars/s), and a similar decrease in the TM rate (from 12.6Mbps to 9.6Mbps).

Figures 10.26 and 10.27 show the inverse case, so now it is Astro-1 reaching a density peak and leading to an out-of-memory error of GASS (but now about 2 hours could be generated). This case is more interesting, since this period also contains the outside part of the GC being observed by Astro-2, this is, a much smaller density peak. Figure 10.26 reveals a minimum star density of about 15 stars/s, a small peak in Astro-2 of 500 stars/s and a large peak of 2600 stars/s in Astro-1. The TM rate starts at about 150kbps, reaching a small peak of 1.9Mbps and a large peak of 8.7Mbps.

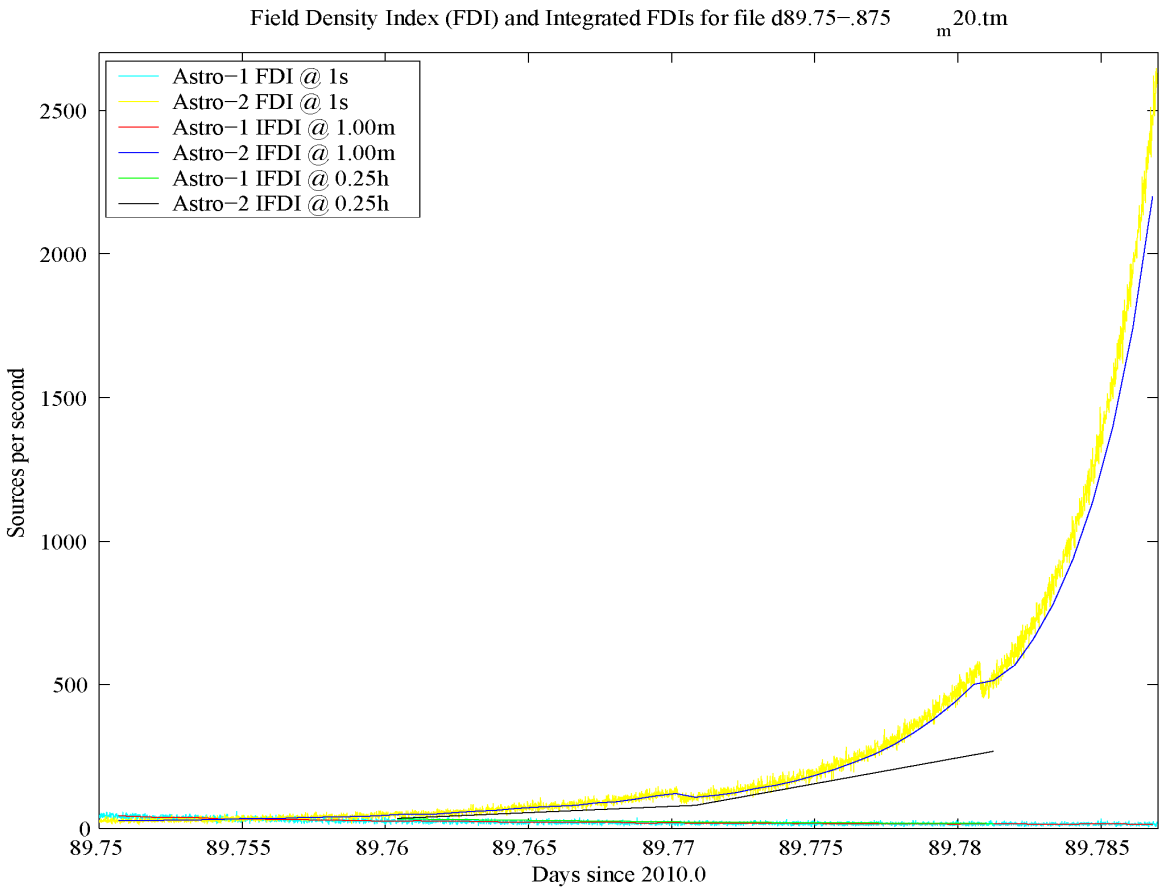


Figure 10.24: Star density curve of 53 minutes at 20th magnitude in a low density area

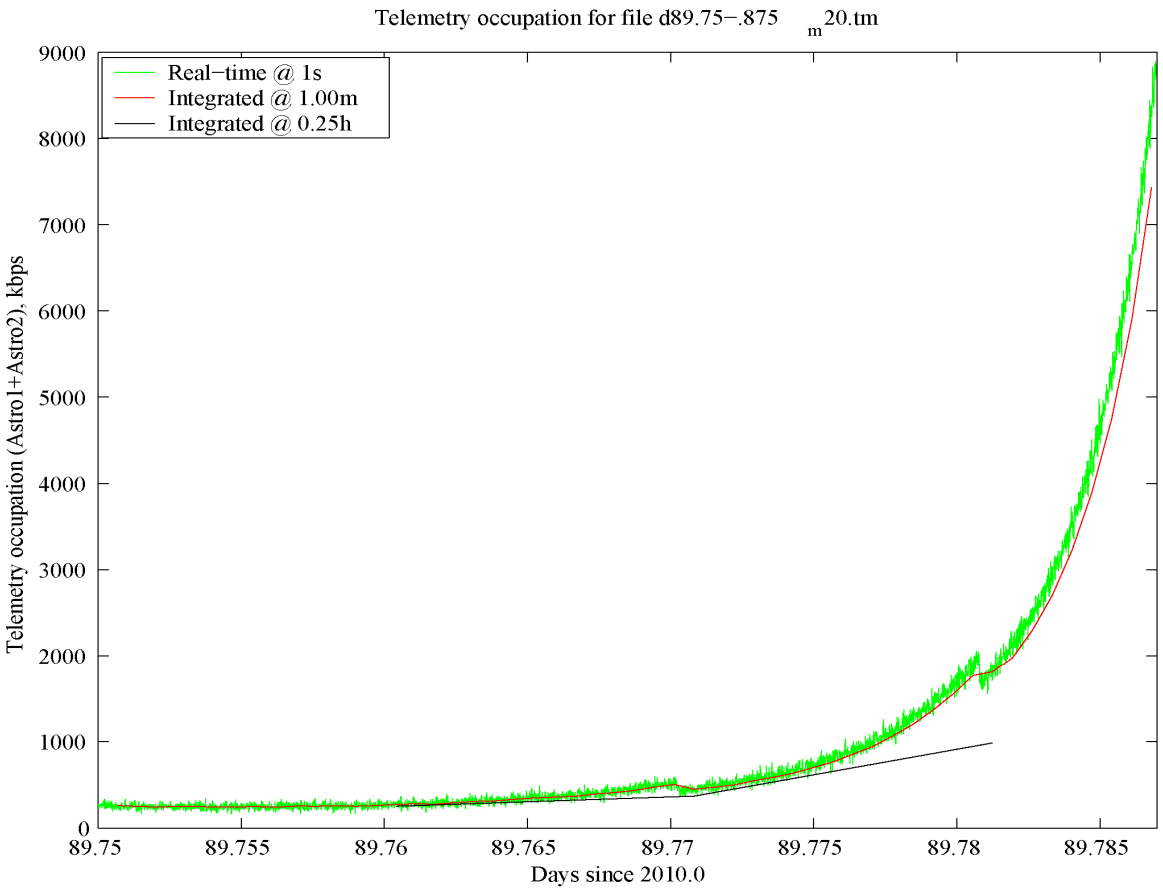


Figure 10.25: TM rate curve of 53 minutes at 20th magnitude in a low-density area

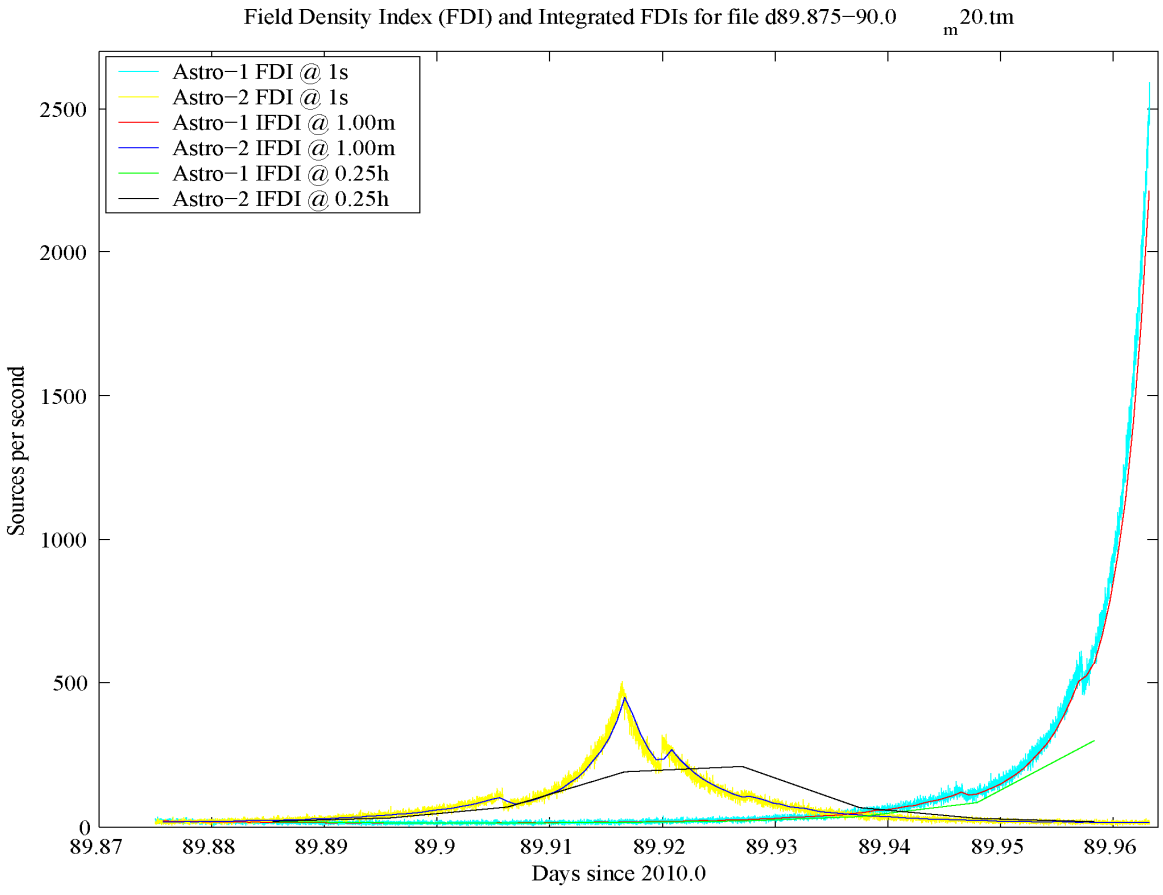


Figure 10.26: Star density curve of 2 hours at 20th magnitude in a low-density area

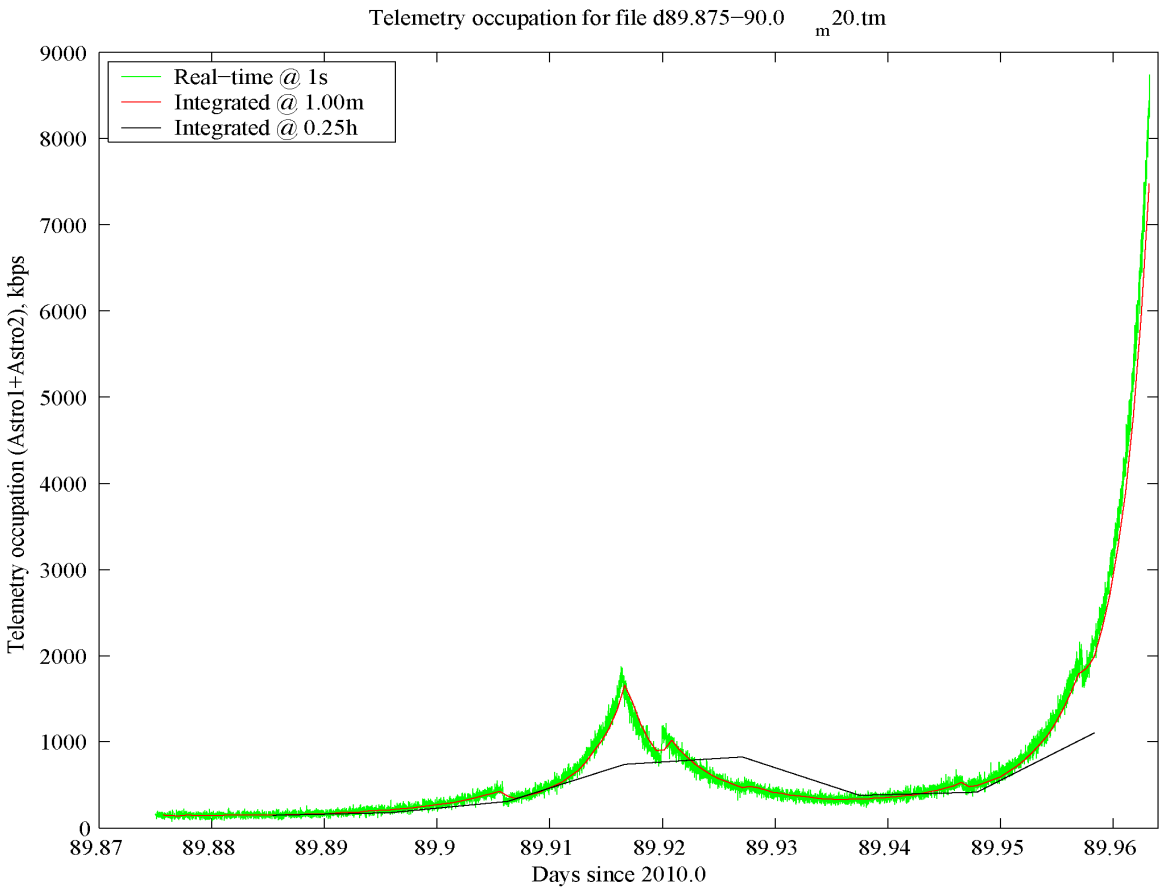


Figure 10.27: TM rate curve of 2 hours at 20th magnitude in a low-density area

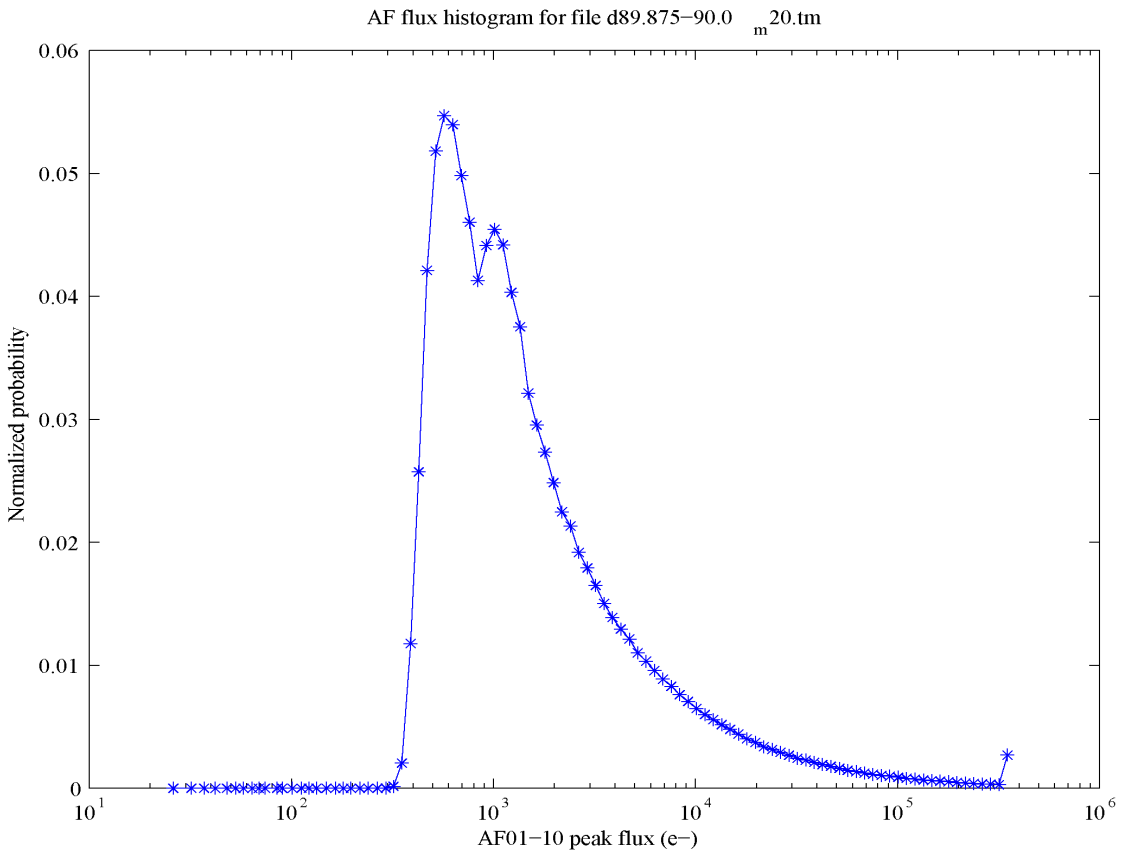


Figure 10.28: Flux histogram at 20th magnitude in a low-density area

Finally, figure 10.28 shows the AF flux histogram in this area. Again, we can see a peak at the saturation level (to the right), although this time is much lower (for the reason previously explained). On the other hand, now a "double-peak" appears close to the histogram maximum, which is due to the GASS instrument model leading to different fluxes along the AF CCDs. This effect is being studied by the GASS team.

10.2.4.2. High-density area

Here we will show the results of simulating a very dense area, around day 120 –which has been found as very crowded during the previous simulations at brighter magnitudes. More specifically, we have started simulating at day 120.2. Our intention, again, was to simulate a long period of time, but the final results exceeded our expectations: with only 3 minutes of mission the output file was already reaching our limit of 2GB. Therefore the final simulation is only 3 minutes 7 seconds long. We converted and analyzed this resulting telemetry file, offering a simulator output shown in figure 10.29. In this snapshot we can find impressive figures: more than 8400 stars are measured every second, generating more than 25Mbps in average. The internal memory buffer is 4MB again, and the processing speed has significantly decreased (now 1 second of the mission needs about 16 seconds for being converted and analyzed), although the final output speed (1.6Mbps) is not so bad. The percentage of faint stars gets even higher, leading to only 2% stars of 16th magnitude or brighter.

```
Converting /mnt/c_win/GASS SIMS/v2.2/tm/d120.2-3m7s_m20.tm to data/bin/d120.2-
3m7s_m20.tm.bin...
100% TM data processed  [ OK ]
[BI] 5615 overflows, 0 underflows.

[BI] Sampling scheme distribution:
[BI]    Window mode: Faint - 97%, Medium Bright - 2%, Bright - 0%
[BI]    AF11 mode: Normal - 100%, Narrow - 0%
[BI]    BBP mode: Normal - 80%, Narrow - 19%
[BI] Observation time:  3.1500 minutes.
[BI] Processing time:   50.2549 minutes.
[BI]    Processing speed ratio: 0.0627 seconds/second.
[BI] 1.6139 Mbps written in average.
[BI] Max. buffer size: 4096.0000 KBytes.
[BI] 1591075 sources in 189 Data Sets
[BI] 1509555 sources in Astro-1, 81520 sources in Astro-2.

In this simulation, Gaia has measured an average of 8418.3867 sources/sec
Average data rate: 25.7476 Mbps.
```

Figure 10.29: Summary of the data conversion and analysis process at 20th magnitude in a dense area

Figure 10.30 (with a forced contrast in order to better appreciate the plots) shows the field density curve in this region, where we can find again these impressive numbers: about 8000 sources are observed every second by Astro-1, while Astro-2 stays at about 425 stars/s. The trend of Astro-1, furthermore, looks as increasing, so this case seems not to be the worst one. Therefore, this is a clear example of a typical maximum that Gaia should be able to cope with. The TM rates (combining both Astro instruments as usual), which are shown in figure 10.31, show values ranging from 27 to almost 30Mbps, with a clear increasing trend. The averaging errors in this case only represent a 3%, so we can conclude that these averaging issues are not so relevant in areas where the star density varies smoothly.

Finally, figure 10.32 shows the AF flux histogram obtained from this observation. We can verify that the peak at saturation level is almost invisible, due to the huge amount of faint stars (about 1.5 million) in front of the saturated samples (about 5600, as the simulator output indicates). The "double peak" in the histogram maximum is also found here, although now it is smoother.

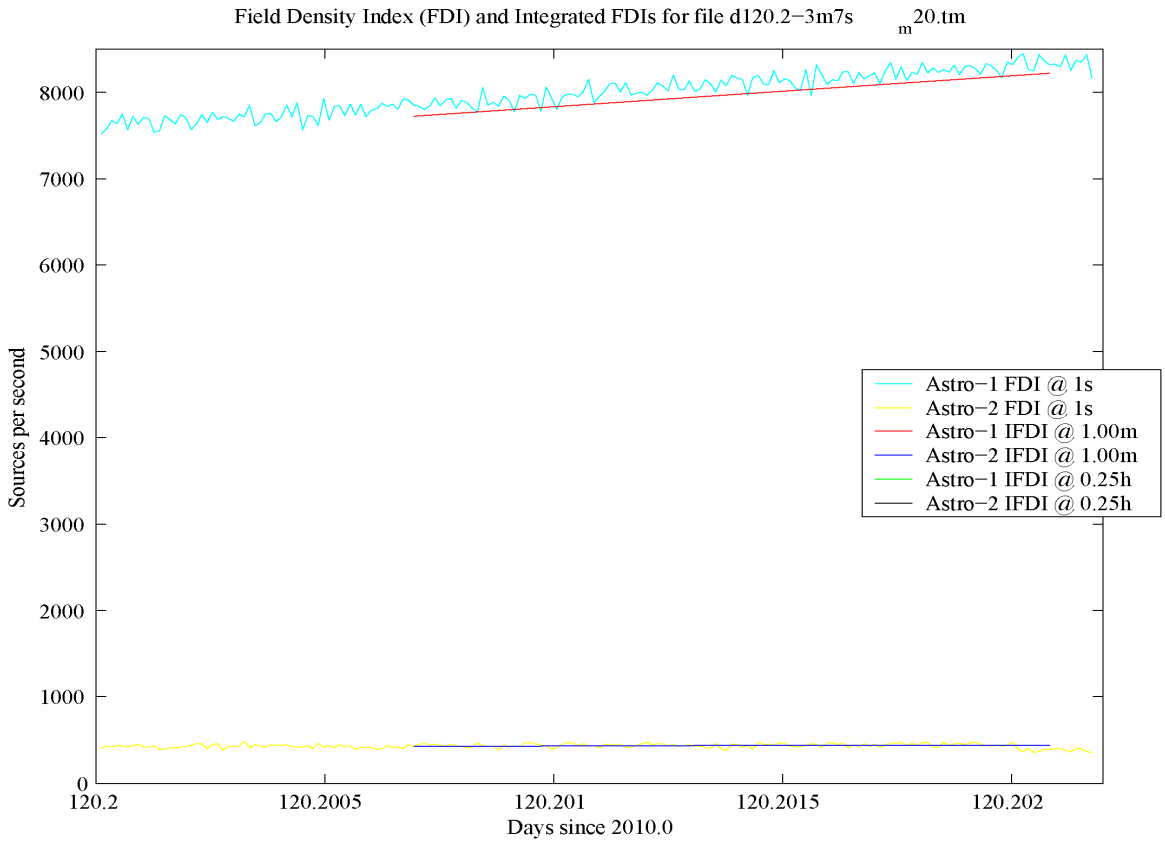


Figure 10.30: Star density curve of 3 minutes at 20th magnitude in a dense area

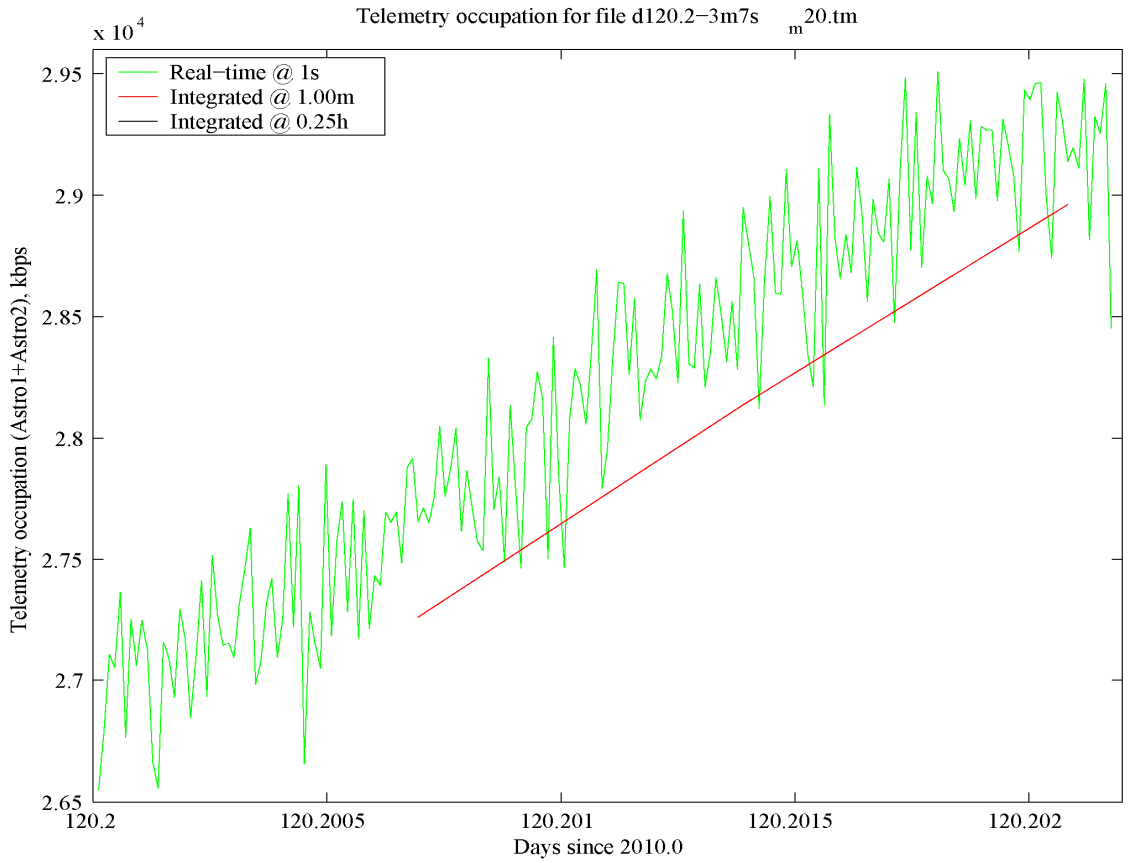


Figure 10.31: TM curve from a 3-minutes observation at 20th magnitude in a dense area

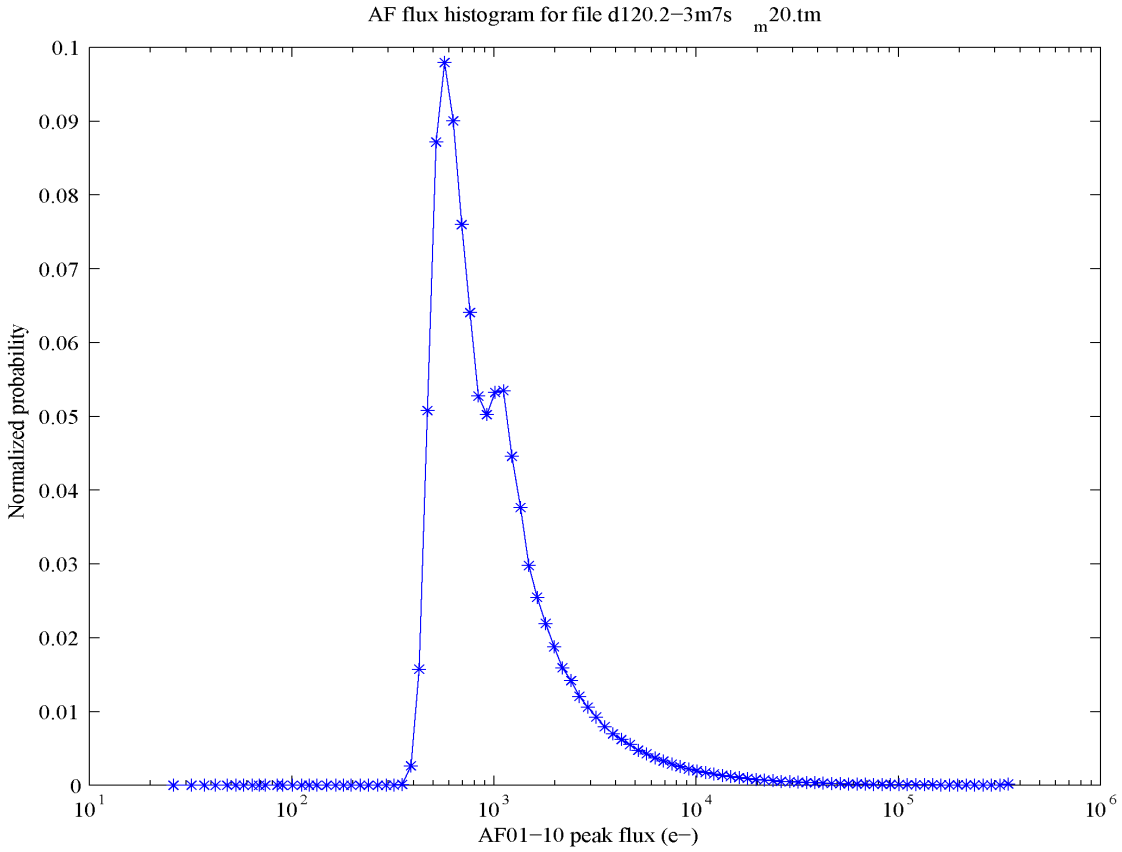


Figure 10.32: AF flux histogram from a 20th magnitude observation in a dense area

10.2.4.3. Baade's window

The last one of our tests is for the most extreme situation we can simulate: Baade's window at 20th magnitude. Simulations at 16th mag already showed a clear maximum, but we preferred doing a "step-by-step approach". As indicated in section 10.2.3.3, the maximum seems to be around day 113.057, so first of all we simulated starting at day 113.05 –where Baade's window starts to show up, as seen in figure 10.21. It made possible the simulation of 2 minutes and a half, so we could clearly see the trend. The results are shown in figures 10.33 and 10.34, where we can still find a moderate star density and TM rates –although increasing significantly. The contrast of figure 33 has been manually emphasized in order to better appreciate the plots. The average star rate obtained is 3700 stars/s (ranging from 3000 stars/s to almost 4500 stars/s), and the average TM rate is 11.3Mbps (ranging from 10.5Mbps to almost 15Mbps). Averaging errors are almost 10%, taking into account that even the long-term integrated curves are averaged only 1.2 minutes. It demonstrates the rapid changes of conditions in this area.

For the next step we moved towards day 113.056, which is really close to the maximum peak (at least as seen in the 16th magnitude test). The results demonstrate this, showing spectacular numbers both in star density and TM rate: almost 15000 stars/s and 45Mbps can be seen in figures 10.35 (again with emphasized contrast) and 10.36. Astro-1, on the other hand, only measures about 55 stars/s. These extreme conditions only made possible the simulation of 8 seconds, which supposed a processing time of 3 minutes and a half (so every second of observation required 27 second of processing and conversion time). Despite of this, a slightly increasing trend is still observed. An internal buffer of 8MB was required to process this amount of data.

Figure 10.33: Star density curve at the beginning of the Baade's window at 20th magnitude

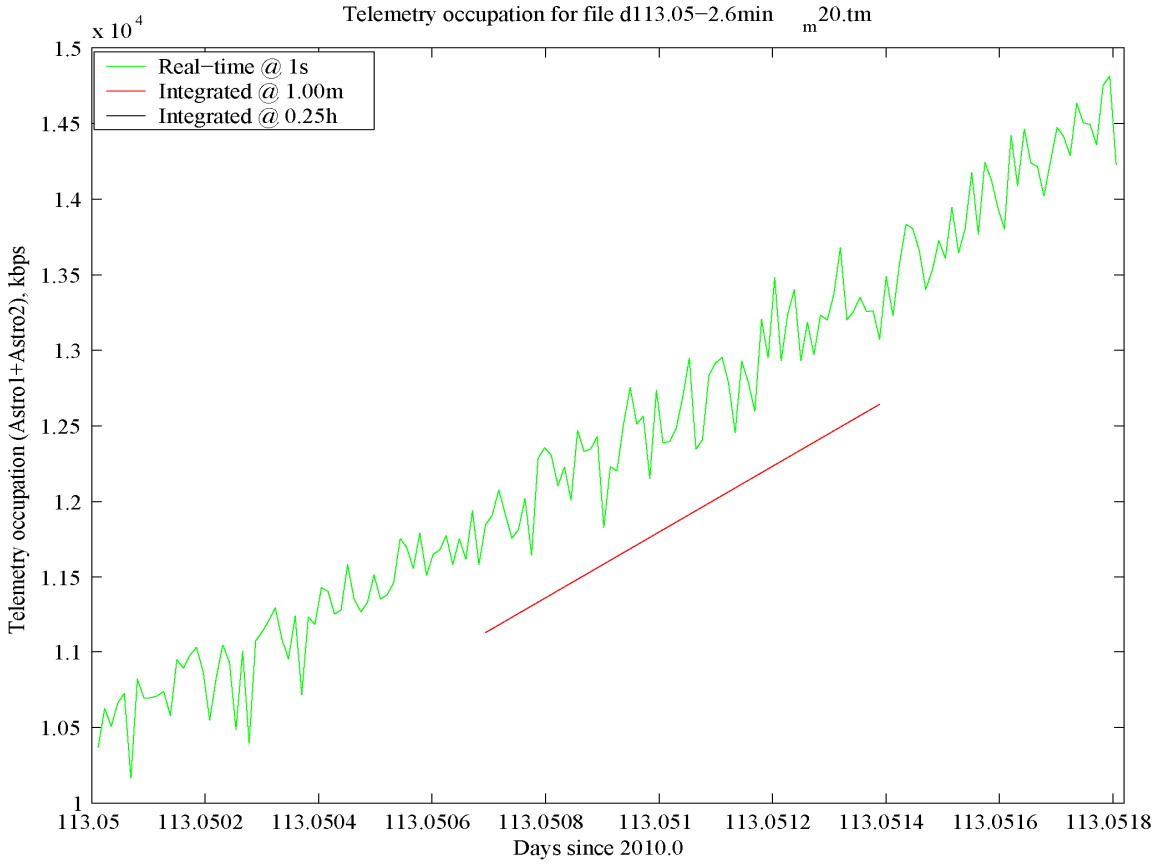


Figure 10.34: TM curve at the beginning of Baade's window at 20th magnitude

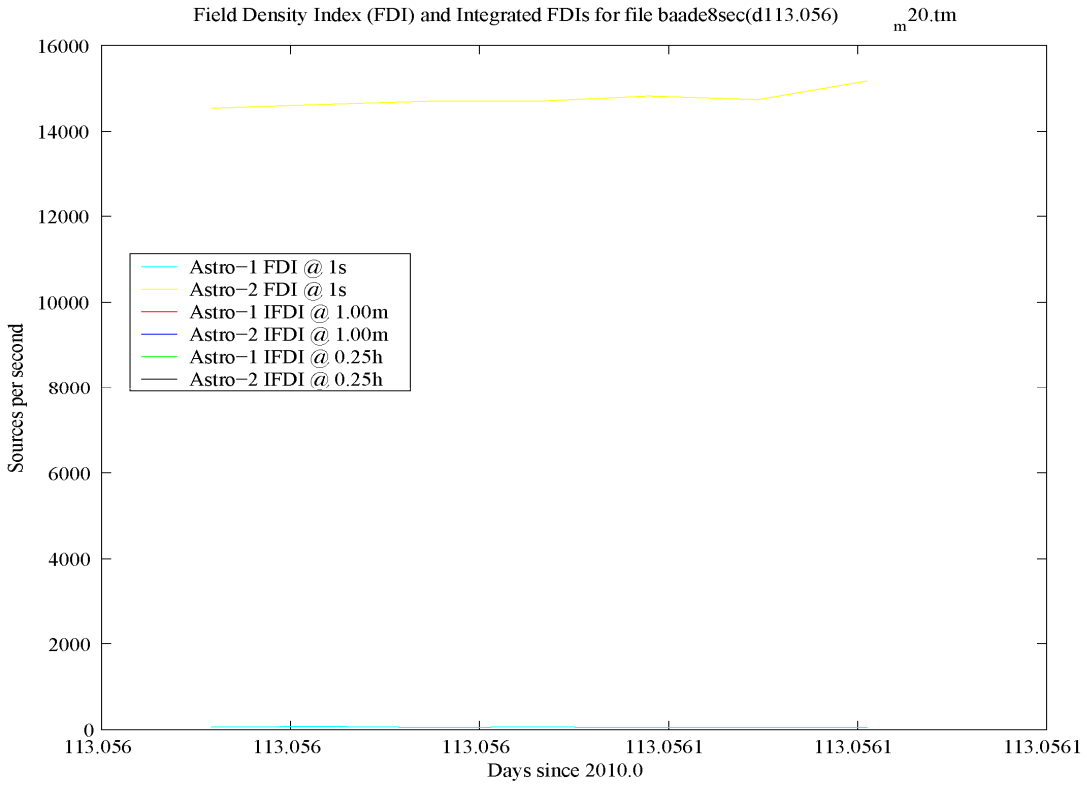


Figure 10.35: Star density close to the maximum peak of the Baade's window at 20th mag.

Figure 10.36: TM rate close to the maximum peak in the Baade's window at 20th magnitude

Finally, we tried simulating even closer to the peak of Baade's window. For this, we started at day 113.0565, but in 10 hours of running time GASS could only generate 3 seconds of this area. After being processed (and converted) with *txt2bin*, an average density of 20000

sources/s was found, leading to an average TM rate of 61.2Mbps. Despite of this, the last values saved to the FDI and TMC files are 20281 stars/s and 68Mbps, so it seems that we have not found the maximum peak yet. Nevertheless, these values are fairly representative of the most extreme operating conditions under which Gaia will operate. The processing time with *txt2bin* (which automatically used 16MB of memory) was 1.7 minutes, so one minute of observation time under such conditions required 35 seconds for being processed and converted, although the writing speed is about 1.75Mbps.

10.3. DATA RESTORATION WITH *BIN2TXT*

All of the calls to *txt2bin* were done with the *–noout* option, thus avoiding an unnecessary output file. Only with the 20th magnitude simulations we generated *.bin* files, which have been restored afterwards in order to verify the correct operation of the full software package. These tests have been completely successful, offering GASS TM files in their original format (ASCII) but containing the quantization and saturation errors. Therefore, this software package could be used within GDAAS in order to include such kind of errors in the astrometric error budget.

```
7758085000000000
4811 1135 5195 4416 5535 5504 842 6455 5103 6194
6387 2929 3621 65 6579 5880 5353 4369 1440 1650
6973 492 5493 4745 6463 1435 2292 6916 6716 622
(…more TDI Offset values…)
1775 7351 7010 73 7218 835 4701 971 6515 6773
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
1525

0002746984000013
0
1 0 0
99
0 0 0 2.489041872975779 1.136792832589638 86
2 1458 6
1162 17 (next, 5x5 ASM fluxes:)
401 102 124 105 404
102 112 240 126 108
124 240 1804 412 193
105 126 412 157 117
404 108 193 117 412
14208
11 11
9057 1460 (AF01 readout coords. Next, AF01 fluxes:)
663 756 796 1072 3805 8337 7428 2594 1140 1029 795 704
13957 1458
205 277 378 1519 5832 8412 4216 899 537 469 255 144
18856 1456
187 257 328 581 3173 7813 6977 2258 601 515 319 199
23756 1454
181 278 540 925 2711 6851 6826 2537 991 725 408 223
28657 1453
237 434 779 2017 5850 7618 3483 1042 799 488 280 164
33556 1451
189 279 569 1031 3433 7326 6275 2034 911 687 370 202
38457 1449
243 458 833 2156 6311 7248 3164 1045 815 460 226 172
43356 1447
192 287 562 1018 3335 7350 6060 2017 923 731 375 215
48256 1445
200 295 312 966 5060 8918 5156 1064 473 455 243 172
53156 1443
178 285 351 1671 6378 8927 3503 658 500 393 219 142
58029 1442
(…)
```

```
7758085000000000
4809 1136 5198 4413 5535 5506 843 6452 5103 6195
6386 2933 3622 66 6576 5880 5352 4370 1437 1650
6972 491 5491 4743 6466 1437 2295 6913 6716 623
(…more TDI Offset values…)
1774 7353 7009 73 7221 836 4699 968 6518 6774
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
1525

2746984000013
0
1 0 0
99
0 0 0 3 1 86
2 1458 6
1162 17 (next, 5x5 ASM fluxes:)
400 101 125 104 403
101 113 241 125 107
125 241 1803 412 194
104 125 412 157 116
403 107 194 116 412
14208
11 11
9057 1460 (AF01 readout coords. Next, AF01 fluxes:)
662 758 796 1073 3803 8337 7429 2596 1138 1031 796 705
13957 1458
203 278 379 1517 5832 8412 4214 897 539 470 256 144
18856 1456
187 256 326 582 3172 7813 6975 2259 603 513 320 198
23756 1454
182 278 539 924 2713 6852 6825 2537 993 726 406 224
28657 1453
235 433 780 2019 5848 7616 3482 1041 801 486 278 166
33556 1451
187 278 571 1031 3434 7327 6275 2035 913 689 369 203
38457 1449
246 459 833 2158 6313 7247 3162 1047 817 459 224 171
43356 1447
192 288 561 1020 3333 7349 6062 2019 924 732 374 214
48256 1445
198 294 310 967 5058 8919 5154 1063 475 454 246 171
53156 1443
176 283 352 1672 6377 8930 3503 657 502 395 219 144
58029 1442
(…)
```

Figure 10.37: Original (left panel) and restored (right panel) example of GASS TM file

Just as an illustrative example, we include a small sample of a GASS TM file, in its original version (left panel of figure 10.37) and in its restored version (right panel, after going through

the *txt2bin – bin2txt* process). We can verify how the errors are smaller than the quantization steps, which have been previously shown in figure 10.1. We note that the original and restored file sizes almost coincide (small differences could exist), while the binary file size decreases in a ratio of about 3. It is obviously not a "compression ratio", although this issue could help in the transmission of large simulations through the Internet. Finally, we note that the restoration speed with *bin2txt* is almost twice as in *txt2bin*, since it already reads binary data and does not perform any statistical analysis.

10.4. CONCLUSIONS

With these tests we have verified all of the specifications of the Static TM CODEC and, most importantly, its reliable operation when quantizing, converting to binary, and restoring to ASCII the GASS TM data files. We have tested its rich capability for statistically analyzing the data of observations from 10^{th} to 20^{th} magnitude under the most typical scenarios, including low-density areas, high-density areas and extremely dense areas (i.e., Baade's window). The following table summarizes the most relevant figures when simulating realistic conditions, this is, with a limiting magnitude of 20:

Observation scenario	***Scenario details***	***Typical densities (per instrument)***	***Typical TM rates (Astro1+Astro2)***
Best conditions (lowest density)	Minimum values during day 89	15 stars/s	150 kbps
Typical density	Average values during day 89	400 stars/s	1.6 Mbps
Typical short-term peak	Maximum values during day 89	3750 stars/s	12.5 Mbps
Sustained maximum in dense areas	Maximum values during day 120	8500 stars/s	29.5 Mbps
Extreme peak	Maximum values in Baade's window	20200 stars/s	68 Mbps

Table 10.1: Summary of the observation conditions tested with GASSv2.2 and the Static TM CODEC

The values shown in the second and third rows for day 89 have been obtained from a weighted average of the two tests executed: 1 hour starting at day 89.75 (with a higher average density) and 2 hours starting at day 89.875 (with a lower average density). We note at this point that the codification of the simulated TM data has not been changed at all, i.e., the optimized codification and transmission schemes described in chapter 7 have not been applied. All of the data generated by GASS has been kept "as-is", even maintaining the Source ID field. The main reason is that, in this way, the resulting data can be used for GDAAS, e.g., in order to quantify the astrometric errors due to quantization. Future studies shall deal with an optimized codification scheme, determining the telemetry savings when compared to this raw, non-optimized codification.

Part IV:
Data Compression

Chapter 11

Preliminary Data Compression Systems

In the previous chapters we have not only described the PDHS as a whole, but we have also developed detailed proposals for the operation of some of its modules, including the timing and transmission schemes and a powerful software tool to simulate this coding process. Up to now we have been treating the data compression as a "black box", using only estimations of its performance but without entering into any details. This chapter describes a set of proposals for a pre-compression system, focusing not only on Astro but, more specifically, on the flux data generated by the ASM, the AF and the BBP. The reason is that, as previously seen, these are the data that represent the highest telemetry occupation and, hence, the data to be best compressed.

It is very important to note that this is a *preliminary* study on the data compression problem in Gaia, and that it was done at the very beginning of this work (this is, in 2001) in order to be used as a *feasibility study*, so that we could determine if this problem had some feasible solution. The reason is that the data compression has always been our final objective, so we had to be sure of its feasibility before starting any detailed work. We have included it here (rather than in the beginning of the thesis) for the sake of a better structured document. The important point to note here is that this work was done *before* any redesign in Gaia, this is, with the baseline design described in ESA (2000). We refer the reader to that document (also known as the *Gaia Study Report*, or the *Red Book*) for more details on this old design. Nevertheless, let us summarize its most relevant features:

- Two different focal planes were used for Astro: Astro-1 and Astro-2, each with its corresponding telescope. Hence, no image combination was done (i.e., every focal plane measured the images from only 1 telescope).
- Both focal planes were identical, each containing:
  - 4 ASM strips (ASM0-ASM3). ASM1 has been the only one inherited (now having one for each FOV, this is, ASM1 and ASM2). ASM3 is the one implemented now by AF01 (this is, confirmation). ASM0 did the same work as ASM1 but only for bright stars, and ASM2 was there for redundancy.
  - 17 AF strips (AF01-AF17). AF17 operated like the current AF11 (this is, with an extended sampling). AF01-16 were reduced to the current AF01-10.
  - 5 BBP strips as in the current design, but their sampling was different in Astro-1 than in Astro-2.
- The spin rate was twice fast as the current one, this is, 120 arcseconds per second. Therefore, the average stars per second was also higher.

Due to this higher scan rate and to the higher amount of measurements (17 AFs rather than 11), the requirement on the data compression ratio was about 7 (instead of 3-4, which is the current one). Nevertheless, this larger amount of repeated measurements also helped in compressing better, thanks to the higher implicit redundancy in the data.

The purpose of this chapter is to introduce the problem of the data compression in Gaia, illustrating it with some preliminary proposals which already offer interesting results. All these proposals can be considered as *pre-compression* systems, after which some standard data compression system should be applied –such as Huffman, LZW or Rice. Section 1 of this chapter shows the general characteristics of flux codification, as well as some of the requirements. Sections 2 to 5 show the different codification schemes for flux data, with a global summary and comparison in section 6. Finally, in section 7 we show some results obtained (Portell et al. 2002b) when simulating these preliminary systems.

11.1. GENERAL FEATURES OF FLUX DATA CODIFICATION

The most important type of data in the Gaia mission is the astrometric data, specially those coming from the Astrometric Field. This kind of data can be classified as *flux data*, that is, the number of photons received in a CCD during a TDI period. This number of photons, coming from a source (or cosmic rays), is converted to electrons (or counts) by the CCD electronics. A CCD will have a *dynamic range*, that is, it will be able to offer a maximum number of counts (*saturation level*) and a minimum useable number of counts. The latter (the lower limit, usually coinciding with the resolution) will be at least 1 electron (e^-) because of the quantized nature of the measurements. However, the ADC (Analog-to-Digital Converter) used in the output of each CCD will play an important role in the calculation of the final resolution, which is obtained from the Read-Out Noise (and the ADC contributes to the RON). The procedure is the following: more bits used in the ADC leads to a better digital resolution (or *quantization step*), which leads to a lower quantization noise and therefore to a lower contribution to the total RON (Read-Out Noise). Then, the final resolution is taken as equal to the RON.

The reason why flux data is converted to digital format is its easier management. This analog to digital conversion will need a *codification scheme* in order to assign a unique binary code to each and every one of the values obtained from the output of the CCD. Several parts of the astrometric focal plane generate flux data:

- Astrometric Sky Mapper (ASM): 5×5 patch and total flux of the source detected
- Astrometric Field (AF): PSF profile or LSF (patch with 6 samples) measured in AF01 to AF16, and wide field patch (30 samples) in AF17.
- Broad-Band Photometer (BBP): LSF in 5 spectral bands, for BBP1 to BBP5.

However, the most important data generation will be located in the AF, where a total of 96 flux data will be generated in AF01 to AF16. Therefore, it is very important to find an optimal codification scheme, in order to reduce this large amount of data. Hence, this chapter will be centered specially in AF01-AF16 flux data codification (and compression), although codification schemes for other parts (ASM, AF17 and BBP) will also be proposed.

11.1.1. Assumptions

First of all, a dynamic range should be defined for the CCDs. The maximum charge for an AF CCD was 330.000e^- in this "Gaia-1" design (although in Annex 1 we show a slightly different value), so this will be taken as the upper limit. For the lower limit, a resolution better than 1e^- at the output of the ADC will make no sense. Valid resolutions should be obtained taking into account the typical fluxes from faintest sources, as well as the required precision for the final results of the mission. As shown in ESA (2000), in page 172, the baseline ADC codes the flux data in 16-bit words. Taking advantage of the whole dynamic range, the least significant bit (LSB) will refer to about 5e^- in the CCD. Therefore, any codification scheme offering a resolution of 5e^- will be considered valid.

The second assumption for this chapter is a constant conversion all over the focal plane, with no added gain selection (apart from a selectable gain already introduced in ESA 2000). Therefore, the charge of a CCD will be pre-amplified with a selectable gain of 3μV/e^- or 6μV/e^-, and then converted to a 16-bit digital value. An improvement to this resolution, specially useful for very faint sources, could be obtained with a second amplification stage, inserted between the analog output stage of the CCD and the input to the ADC. This second stage could increase the amplification of low signals, e.g., doubling a signal lower than 165.000e^- and therefore also doubling the resolution. This option should be further studied, and will not be considered in this work (although it could be easily combined with the codification schemes shown here). It is very important to note that this option was not included in the baseline. Finally, there are some assumptions made to obtain the estimated compression ratios. These estimations assume that:

- A perfect synchronization exists between the TDI and the satellite attitude.
- Neither cosmic rays nor charged particles hit the CCDs.
- Flux data readout is performed when necessary, ignoring TDI offsets.
- CCD response is ideal (i.e., quantum efficiency is 1).

This means that the numbers shown in this chapter are the "best possible results" achievable with these encoding techniques. The real compression ratios will be slightly lower, and should be found via simulation. In the next chapter we will describe more realistic results using improved models and simulations.

11.1.2. Requirements

The astrometric data, calculated from the flux, is the most important data of the mission. Therefore, it is imperative to get the best quality and thus no errors are allowed while reading and coding flux data, because flux data precision affects directly to the precision obtained in the final results of the mission. Because all of this, no errors or precision losses are allowed in the codification scheme to be selected for Gaia, and therefore lossy compression techniques will not be considered here. This is the main requirement for the codification scheme of the flux data, and this is the ultimate reason of some solutions or variations proposed in the following codification schemes: data integrity must be assured at 100%. Thus, if a coding requires some assumptions on the data to be codified, it will also need a "security system" leading to a not-so-high compression rate (or even to an expansion) when the data is not within the expected margins.

Another requirement for the coding scheme is the compression ratio: some studies assume a final compression ratio of 2 for the data generated in the AF, and a compression of 3 for the data generated by the BBP. Therefore, if the selected coding scheme already leads to this compression ratio (with no data loss) it will be considered a valid scheme. Otherwise, the remaining compression required should be obtained by the communications system, which could be difficult. However, some studies like a preliminary version (based on "Gaia-1") of Masana et al. (2004) indicated a required compression ratio of 7.5, so efforts should be directed towards obtaining this ratio.

11.1.3. Possible solutions

There are many codification schemes that could be applied to the flux data, many of them including some kind of data compression. The easiest solution is a simple full 16-bit encoding, which guarantees both data integrity and constant data length but leads to a maximum size of data generated. Other solutions assume some kind of knowledge about the data (the shape of the PSF, the total expected flux...) but need some kind of "security system" in order to avoid loss of data if the *real* data is not within the expected margins.

11.2. FULL 16-BIT ENCODING

11.2.1. General description

This is the simplest encoding scheme, with no data loss and a perfect operation without the need of any assumption about the data to be encoded. However, this scheme is also the most "expensive" solution in terms of telemetry and amount of data generated. Its operation is based on a linear conversion of the counts obtained by the CCD, with the selectable gate phases and selectable output gain as the only parameters available. This is, these two systems will be used in order to avoid saturation at pixel-level and therefore to take a –more or less– fixed upper limit of about 330.000 e^-. Then, the number of charges measured in a sample will be linearly

coded, taking 330.000 e^- as the dynamic range. As an example, if a sample read out leads to 170.000 e^-, the code obtained will be the following:

$$code = num\_counts \cdot \frac{max\_code}{max\_count} = 170.000e^- \cdot \frac{2^{16}-1}{330.000e^-} = 33.760$$

The resolution obtained with this encoding scheme is uniform for all the range:

$$lsb = \frac{max\_count}{max\_code} = \frac{330.000e^-}{2^{16}-1} \cong 5e^-$$

This resolution is really good for very bright sources, but with faint sources the quantization error represents a high percent of the total number of measured counts (specially for the PSF axes, where the number of counts is still smaller). Although this is the baseline, some kind of improvement could be implemented in this encoding scheme in order to increase the resolution for fainter stars: for example, scaling the conversion factor (increasing it) when a very faint star (mag≥18) is detected in the ASM, and therefore increasing the resolution for fainter stars. However, this would imply a problem if a cosmic hit occurs when measuring that star: the number of counts would fall out the reduced range. Also, the main problem would not be solved: this codification implies a large amount of data generated.

It is important to note that this codification scheme can be considered as the basis for all the other codification schemes: they all can be obtained by just adding some operations or processes to the data obtained with this simple 16-bit encoder.

11.2.2. Assumptions

No assumptions are needed in this codification scheme if the baseline is kept. If some kind of compensation is implemented in order to increase resolution for faint sources, then it will be assumed that no "flash" or cosmic ray will be received when using that higher amplification. If so, that cosmic ray or interference could fall out the reduced dynamic range, and therefore some data could be lost. In order to avoid this, data read-out range should be verified and some flag could be added to the data in order to know if the dynamic range has been changed.

11.2.3. Operation

1. Obtain the analog CCD data, whether in 3μV/e^- or 6μV/e^-.
2. If desired (and if the source is faint enough), amplify this measurement in order to increase the resolution. If so:
   a) Calculate the desired (or needed) compensation, from ASM flux data. Because the default resolution is about 5e^-, a compensation gain higher than 5 is not needed, and therefore the gain of this compensator will be between 1 and 5.
   b) Verify the range of the data read. If the range exceeds the ADC saturation level (330.000e^- or equivalent), then the original data (with no compensation) will be used instead.
   c) Calculate a flag in order to indicate the real dynamic range used (with or without compensation, and the compensation used).
3. Convert the analog data to digital data, using a 16-bit encoder and a dynamic range corresponding to 330.000 e^- without compensation (with compensation the equivalent charge will be smaller, up to 66.000e^-).
4. If compensation is used, add the flag to the data. This can be done by adding the flag to a whole patch, or by adding the flag to each and every one of the samples (basic data). Our recommendation is to add one flag per patch.

If faint source compensation has to be used, and in order to avoid an excessive data generation due to the flags, our recommendation is to use a single compensation gain. In this way, the flag will have to be only 1 single bit: compensation applied or not. For the limit of the compensation and its value, our recommendation is a gain of 4 applied to sources with a global flux lower than 50.000e^-. This leads to a margin of 32.500e^- for fluctuations (up to 82.500e^-, the dynamic range available with a compensation of ×4), and a quantization noise lower than 0.1% for bright sources (>50.000e^-), and lower than 1.3% for a source as faint as 100e^-.

This scheme can be applied to the whole focal plane, not only to AF01-16, although some values should be verified: 330.000e^- dynamic range and 6e^- RON are valid for the CCDs of the AF, but CCDs of the BBP or the ASM will have different values.

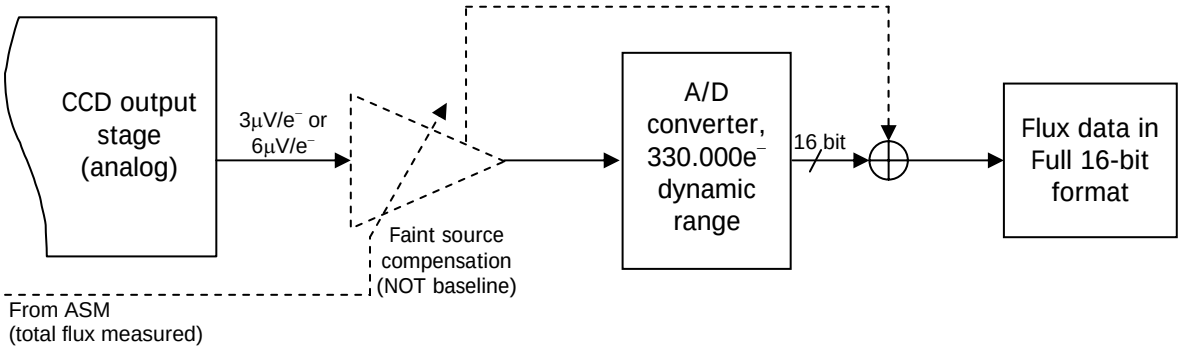


Figure 11.1: Full 16-bit encoding scheme

11.2.4. Data frames

Figure 11.2 shows a draft of the frames generated with this coding scheme for every source detected, also including bit sizes.

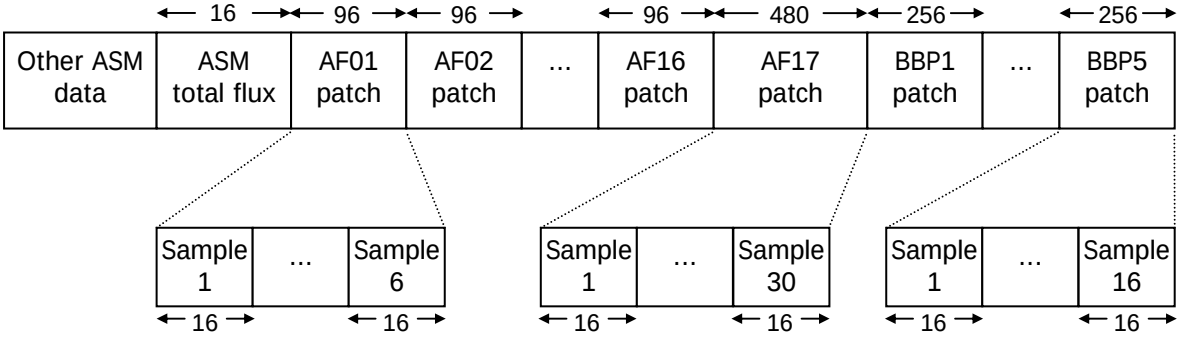


Figure 11.2: Data frames generated with a full 16-bit encoding

This scheme, valid for Astro-1, is also valid for Astro-2 except for a change in the number of BBP samples (10 instead of 16). Also, if compensation is used, a 1-bit flag has to be added at the beginning of every patch (just before every sample #1). Finally, the other ASM data (and telemetry data) is described in chapter 6.

11.2.5. Performance estimations

The data generated with this coding scheme lead to fixed lengths of data fields:

- ASM total flux: 16 bits
- AF01-AF16: 16 CCD columns with 6-sample patterns, 16 bit per sample. Hence, 16×6×16=1536 bits

- AF17: 30-sample pattern, 16 bit per sample. Hence, 30×16=480 bits
- BBP (Astro-1): 5 CCD columns of 16-sample patterns, 16 bit/sample. Hence, 5×16×16=1280 bits
- BBP (Astro-2): 5 CCD columns of 10-sample patterns, 16 bit/sample. Hence, 5×10×16=800 bits

TOTAL: 3312 bits (Astro-1) or 2832 bits (Astro-2), for measured flux data of 1 source.

If compensation is used, then 1 bit should be added per pattern:

- ASM total flux: 16 bits
- AF01-AF16: 16 columns of 6-sample patterns, 16 bit each, plus 16 flags (for 16 patterns). Hence, 16×6×16+16=1552 bits
- AF17: 30-sample pattern, 16 bit per sample, plus 1 flag. Hence, 30×16+1=481 bits
- BBP (Astro-1): 5 columns of 16-sample patterns, 16 bit each, plus 5 flags. Hence, 5×16×16+5=1285 bits
- BBP (Astro-2): 5 columns of 10-sample patterns, 16 bit each, plus 5 flags. Hence, 5×10×16+5=805 bits

TOTAL: 3334 bits (Astro-1) or 2854 bits (Astro-2), for measured flux data of 1 source.

The physical resources (i.e. the electronics, etc.) needed to implement this coding scheme are minimal, but the telemetry and the data resources are maximal. Therefore, the telemetry consumption using a full 16-bit coding will be used as a reference, i.e., the performance of the encoding schemes described in the following sections will be always referred to this one.

11.3. ADAPTIVE ENCODING

11.3.1. General description

The most important fraction of sources to be observed by Gaia corresponds to very faint sources; hence, an important part of the 16-bit dynamic range will not be used in most of the cases. Moreover, the ASM takes a first global measure of every source during the detection phase, and so when a source is being scanned by the AF (and the BBP) its approximate global flux is already known. Therefore, as already proposed in Vannier (2000), one can take advantage of this previous knowledge to adapt the dynamic range to the expected flux for that source. In fact, this is the same idea used for the *faint source compensation* described in the previous section (for the full 16-bit encoding), but used for data compression and not for resolution improvement (although both can be done this way).

11.3.2. Assumptions

For an optimal operation of this coding scheme, it is assumed that there will be no important fluctuations in the detected flux for a given source along the focal plane. That is, the flux (and the pattern) measured for a given source should be (more or less) the same from the ASM0 to the BBP5. This assumption is specially important for the first version of the coding scheme (*original "adaptive" encoding*): if the assumption fails, some data could be lost. An example is the hit of a cosmic ray while measuring the source, and therefore an important increment of the flux detected while measuring that source. However, this could be avoided by using some other version of the coding scheme: in this way, big fluctuations in the flux detected will lead simply to some more data generated.

11.3.3. Original "Adaptive" Encoding: weakly lossy encoding

The original proposal described in Vannier (2000) is a modification of the "faint star compensation" previously described, increasing the resolution but also using less bits for the codification.

11.3.3.1. Operation

1. Obtain the ASM flux data.
2. Depending on ASM flux data, the analog data obtained from the CCD will be scaled in one way or another:
   a) Magnitude<15 (Charge>10.000e^-): ×1 (no changes are made).
   b) 15≤Magnitude<18 (10.000e^->Charge>750e^-): ×33 (ADC dynamic range corresponds to 10.000e^-).
   c) Magnitude≥18 (Charge<750e^-): ×440 (ADC dynamic range corresponds to 750e^-).
3. Also depending on ASM flux data, the number of bits to be used will be selected:
   a) Magnitude<15 (Charge>10.000e^-): 16 bits (resolution: ~5e^-).
   b) 15≤Magnitude<18 (10.000e^->Charge>750e^-): 12 bits (resolution: 2.4e^-).
   c) Magnitude≥18 (Charge<750e^-): 9 bits (resolution: 1.5e^-).
4. CCD analog data is converted using the number of bits selected in (3) and the compensation (gain previous to ADC) selected in (2).

This coding scheme leads to a better resolution as well as some degree of compression for faint sources (up to 16/9=1.7 compression ratio). However, this is not really a lossless codification technique: suppose that a source is detected at the ASM with a global flux of 9.000e^- (and, therefore, a dynamic range of 10.000e^- is selected). If some cosmic ray (or whatever interference or noise) falls over the scanning zone of that source on the AF, and if this cosmic ray has enough energy, the readout flux in that zone could be higher than 10.000e^- and therefore saturate the coder. So this codification scheme cannot be considered for Gaia, if no "security system" is included in order to avoid data loss. Moreover, the *faint source compensation* is not within the baseline.

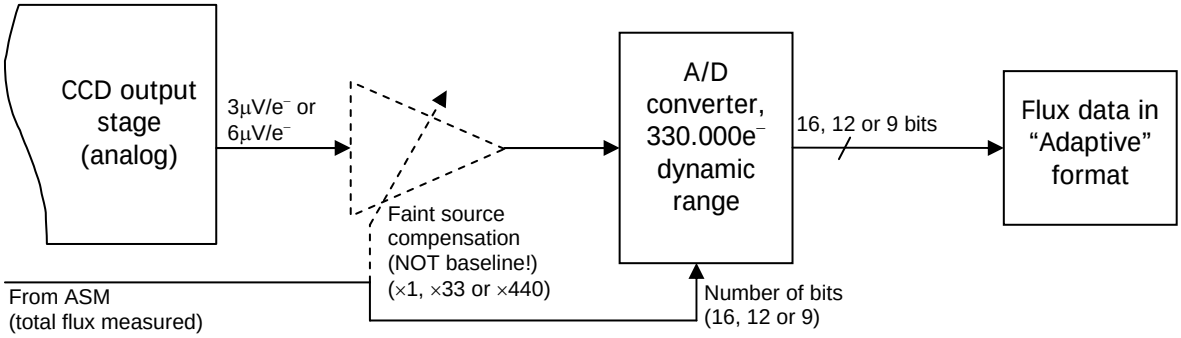


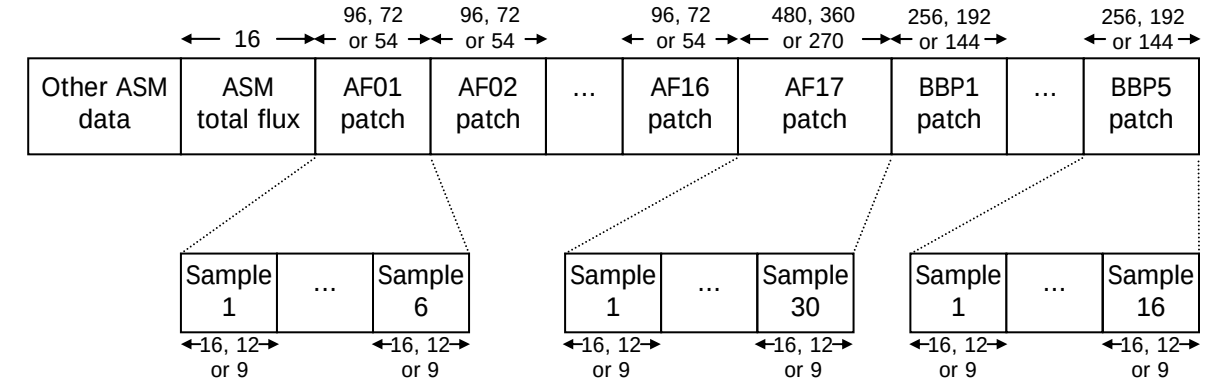
Figure 11.3: Original "adaptive" encoding scheme

This encoding scheme is also valid for all over the focal plane (AF and BBP), except for the ASM (where flux data has to be coded in full 16-bit format). It is important to take into account that the insertion of a "compensator" before the ADC can lead to implementation problems, also adding the need of an ADC with variable resolution. This coding scheme could also be implemented by maintaining always the same resolution (i.e., with no resolution compensation) and with a fixed 16-bit ADC. With a fixed scheme like this, compression tasks

can be done by simply taking the correct number of bits at the output of the ADC. That is, if 9-bit coding is required, only 9 LSBs of the ADC output will be read, and the rest 7 MSBs will be ignored. In this way, resolution will not be improved at all but compression will be done. This variation on the scheme can also be applied to the other coding schemes explained in the following sections.

11.3.3.2. Data frames

In the following scheme, the sizes for every sample will be determined by the total flux data of the ASM, following the rule described in 11.3.3.1 (step 3). The global structure of the data frame is the same as obtained with a full 16-bit encoding: only field lengths are different. This scheme, as well as all the data frames schemes of the following sections, is again valid for both Astro-1 and Astro-2, just changing the number of BBP samples (10 for Astro-2 and 16 for Astro-1). The other ASM data (and other telemetry data) is also described in chapter 6, valid for every data frame description.



Minimum frame size (ASM total flux+AF+BBP, in Astro-1): **1870 bits** (**56.4%** of Full 16-bit Encoding frame size).

Maximum frame size: **3312 bits** (**100%** of F16E size).

Figure 11.4: Data frames generated with an original "adaptive" encoding

11.3.3.3. Performance estimations

The data fields generated with this encoding scheme do not have a fixed length, but they depend on the flux of every source. Some sources will be coded with 16 bits per sample, while others will be coded with 12 or 9 bits per sample. Therefore, performance estimations are not so easy to find (its real performance should be obtained by simulations, as well as the data integrity results). However, a first estimation can be found if one takes a look at the expected star distribution shown in table 3.12 of ESA (2000):

G [mag]	Fraction of all stars	
<15	4%	4%
15 – 16	4%	26%
16 – 17	8%	
17 – 18	14%	
18 – 19	25%	70%
19 – 20	45%	

Table 11.1: Star density as a function of the magnitude (Galaxy model)

With this distribution, average (weighted) data sizes for every field can be found:

- ASM total flux: 16 bits (unchanged).

- AFxx or BBPx sample: 0.04×16 bit + 0.26×12 bit + 0.7×9 bits ≅ 10 bits

Therefore, total size can be found easily:

- ASM total flux: 16 bits
- AF01-AF16: 16 columns of 6-sample patterns, ~10 bit each. Hence, 16×6×10=960 bits
- AF17: 30-sample pattern, ~10 bit per sample. Hence, 30×10=300 bits
- BBP (Astro-1): 5 columns of 16-sample patterns, ~10 bit each. Hence, 5×16×10=800 bits
- BBP (Astro-2): 5 columns of 10-sample patterns, ~10 bit each. Hence, 5×10×10=500 bits

TOTAL: 2076 bits (Astro-1) or 1776 bits (Astro-2), for measured flux data of 1 source. Compression ratio (wrt full 16-bit encoding, no compensation): **~1.6**

The resources needed to implement this coding scheme are more or less the same than those needed for a full 16-bit coding. All operations can be executed sequentially, and no special calculation is needed.

11.3.4. Modified "Adaptive" Encoding: Lossless encoding

The idea of an "adaptive" encoding is an interesting way to take advantage of the star density as a function of the magnitude (as seen in table 11.1). However, one can improve the compression ratio if the resolution is not improved. Furthermore, since there is an important probability of data loss in this algorithm (e.g., due to cosmic rays), some kind of verification must be implemented. In this improved adaptive algorithm these two features have been included: the main idea has been kept (three levels of compensation and variable length codification), but the number of bits used (i.e., resolutions) have been decreased and hence a better compression ratio is obtained, also including a flag in order to guarantee data integrity. Furthermore, a margin has been included (decreasing the compensations) in order to avoid false "thresholds" on the limit of a resolution interval. Details are described in the following subsections.

11.3.4.1. Operation

1. Obtain the ASM flux data.
2. Depending on ASM flux data, the analog data obtained from the CCD will be scaled in one way or another:
   a) Magnitude<15 (Charge>10.000e^-): ×1 (no changes are made).
   b) 15≤Magnitude<18 (10.000e^->Charge>750e^-): ×30 (ADC dynamic range corresponds to 11.000e^-, so a 1.000e^- margin is included).
   c) Magnitude≥18 (Charge<750e^-): ×400 (ADC dynamic range corresponds to 825e^-, so a 75e^- margin is included).
3. Also depending on ASM flux data, the number of bits to be used will be selected:
   a) Magnitude<15 (Charge>10.000e^-): 16 bits (resolution: ~5e^-).
   b) 15≤Magnitude<18 (10.000e^->Charge>750e^-): 11 bits (resolution: 4.9e^-).
   c) Magnitude≥18 (Charge<750e^-): 8 bits (resolution: 2.9e^-).
4. CCD analog data is verified:
   a) If the charge measured for that CCD (for the whole pattern) remains within the predicted margin plus a 10%, then the "adaptive" coding is used and it is marked by

assigning "1" to the flag for that pattern. For example, if ASM flux was 9.999e⁻ (or lower) and the flux read for that CCD is 10.999e⁻ (or lower).

b) Else, full 16-bit coding is used for the whole pattern and it is marked by assigning "0" to the flag. For example, if ASM flux was 670e⁻ and the flux read for that CCD is 900e⁻.

5. CCD analog data is converted using the final number of bits and compensation selected in (4).

This coding scheme leads to a better compression ratio, but offers a lower resolution (the resolution is improved just for the fainter stars). However, data integrity is absolutely guaranteed here, no matter what could happen on the focal plane during measures (cosmic rays, etc.). Also, it is important to note that faint source compensation is not within the baseline.

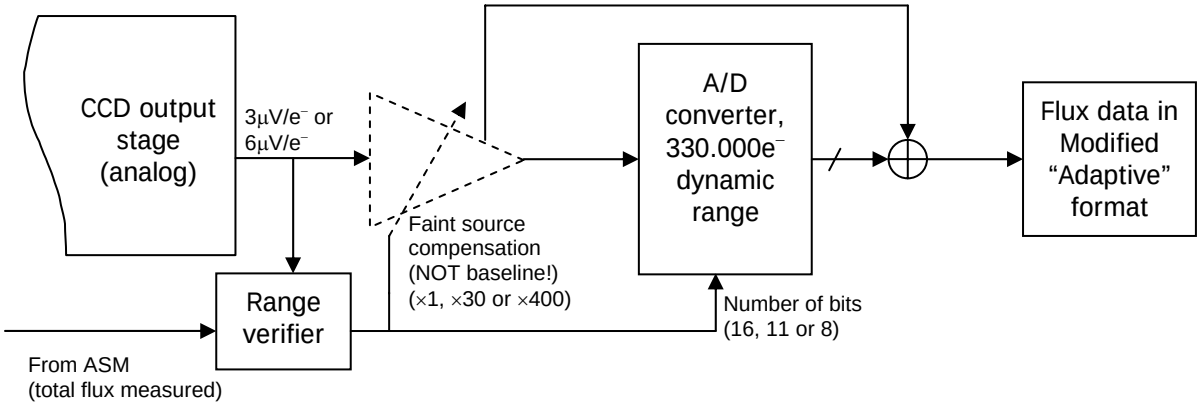
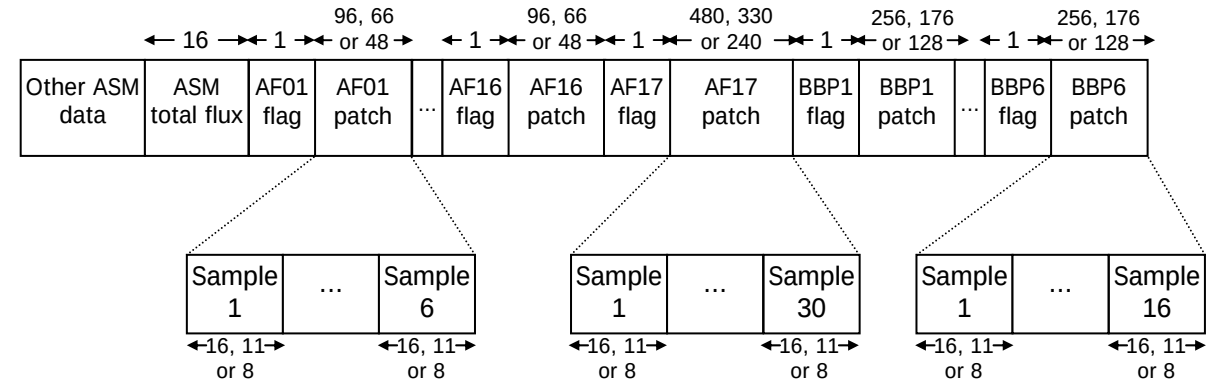


Figure 11.5: Modified "adaptive" encoding scheme

This encoding scheme is also valid all over the focal plane (AF and BBP), except for ASM (where flux data has to be coded in full 16-bit format).

11.3.4.2. Data frames

The following figure describes the data frames generated using this encoding scheme. Field lengths in bits are shown.



Minimum frame size: **1686 bits** (**50.9%** of F16E size).

Maximum frame size: **3334 bits** (**100.7%** of F16E size).

Figure 11.6: Data frames generated with a modified "adaptive" encoding

11.3.4.3. Performance estimations

As for the original "adaptive" coding, the data fields generated with this encoding scheme do not have a fixed length: they depend on the flux of every source. Some sources will be coded with 16 bits per sample, while others will be coded with 11 or 8 bits per sample. Also, a 1-bit flag will be added at the beginning of every pattern data field. Using the star distribution shown in Table 11.1, average data sizes can be estimated:

- ASM total flux: 16 bits.
- AFxx or BBPx sample: 0.04×16 bit + 0.26×11 bit + 0.7×8 bits = 9.1 bits
- AF01 to AF16 pattern: 6×9.1 bits + 1 bit (flag) = 55.6 bits
- AF17 pattern: 30×9.1 bits + 1 bit = 274 bits
- BBP pattern (Astro-1): 16×9.1 bits + 1 bit = 146.6 bits
- BBP pattern (Astro-2): 10×9.1 bits + 1 bit = 92 bits

Therefore, total size (average) can be estimated as follows:

- ASM total flux: 16 bits
- AF01-AF16: 16 columns of 55.6-bit patterns. Hence, 16×55.6=889.6 bits
- AF17: 274 bits
- BBP (Astro-1): 5 columns of 146.6-bit patterns. Hence, 5×146.6=733 bits
- BBP (Astro-2): 5 columns of 92-bit patterns. Hence, 5×92=460 bits

TOTAL: 1912.6 bits (Astro-1) or 1639.6 bits (Astro-2), for measured flux data of 1 source. Compression ratio (wrt full 16-bit encoding, no compensation): **~1.73**.

However, this compression ratio could be lowered if many cosmic rays hit the CCDs. The physical resources needed to implement this are not an important restriction: just a "range verifier" has to be added, and it does not need to do many calculations. All operations can still be kept in a sequential order (first come, first served).

11.3.5. Fully Adaptive Encoding

A fully operative and reliable coding scheme has been described in the previous sections. However, the compression ratio obtained with an "adaptive" coding is still very low. The reason of this low ratio is, to a big extent, the limited adaptation capability of the described coder: only three levels of adaptation are allowed, and the adaptation rules are extremely fixed.

One solution to this problem is a *fully adaptive encoding*, that is, a *really* adaptive encoding, generating always the exact number of bits needed. A way to do this is the following: first of all, the CCDs of the whole focal plane have to be read for a given source (ASM, patterns of the AF, and patterns of the BBP). Then, a *typical* flux for that source should be determined, that is, the approximate flux measurement that has occurred more times along the focal plane. Also, the *maximum* flux read along the focal plane should be determined.

Once typical and maximum fluxes are obtained, one can easily find the number of bits required to code these values. Then, each and every one of the patterns read for that source will be coded with the *typical* bit length or with the *maximum* bit length. Of course, flags will be needed: two at the beginning of the data frame, in order to indicate *typical* and *maximum* lengths found (and used), and 1-bit flags for every pattern, in order to indicate whether a typical or maximum length is used for that pattern. This coding scheme leads to coding lengths perfectly adapted to the flux of every source. Also, interferences like cosmic rays are included in a better way: full 16-bit encoding will not be needed anymore if something fails and the range falls out of the expected one. Finally, this is the only coding scheme valid for all the flux data generated all over the focal plane, including the ASM (patch and total flux received in detection).

11.3.5.1. Operation

1. Read all the flux data for the source: ASM, AF and BBP.
2. Obtain maximum length:
   a) Analyze all the flux data of all CCDs and obtain the maximum flux received.
   b) Find the number of bits needed to code that value. This will be the maximum length.
3. Obtain typical length:
   a) Generate a histogram of all the fluxes received.
   b) Find the typical flux level threshold, i.e., the level above which an important part of measurements are included. For example, the level above which a 90% of the flux measurements are included.
   c) Find the number of bits needed to code that value. This will be the typical length.
4. Add two flags at the beginning: one indicating the maximum length, the other indicating the typical length. 4 bits will be enough for every flag (maximum code length: 2^4=16 bits).
5. Analyze all the flux patterns sequentially. For each one of them:
   a) Decide if all the values of that pattern can be coded with the typical length. If so, generate a 1-bit flag indicating a typical length coding, and then code the values by just taking the *typical_length* Least Significant Bits (LSBs).
   b) If some value exceeds typical length coding capability, generate a 1-bit flag indicating a maximum length coding, and then code the values with maximum length (i.e. taking the *maximum_length* LSBs).

An important detail of this coding scheme is that no resolution improvement is achieved, that is, a constant 5e^- resolution is used for all sources. Although resolution enhancement could be implemented, its implementation would be more difficult, as well as the adaptive algorithm. Moreover, compression ratios would be lower. Figure 11.7 shows an operational scheme for this encoding technique. However, its operation is easier to understand with the flux diagram shown in figure 11.8.

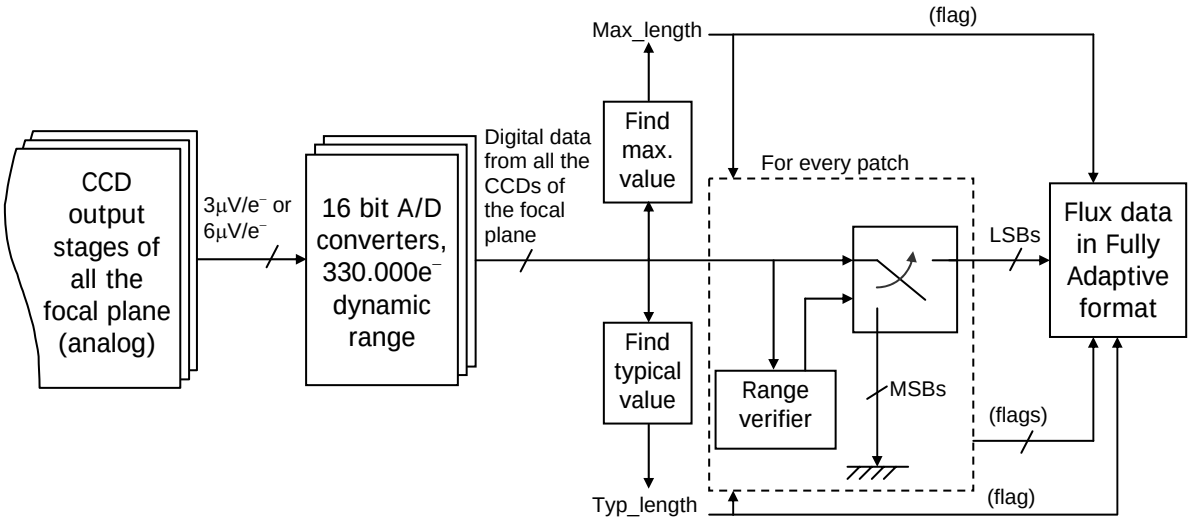


Figure 11.7: Fully Adaptive encoding scheme

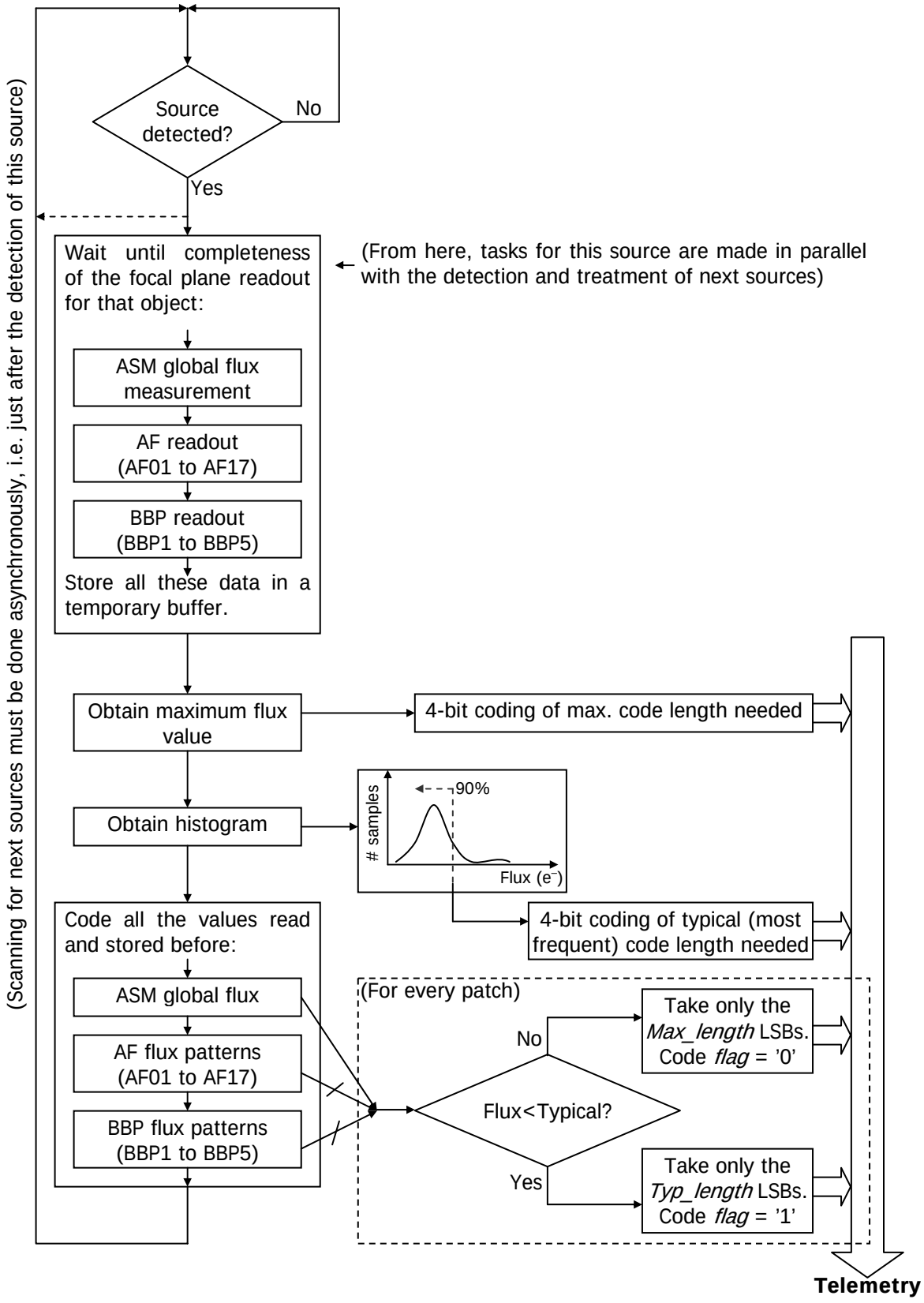
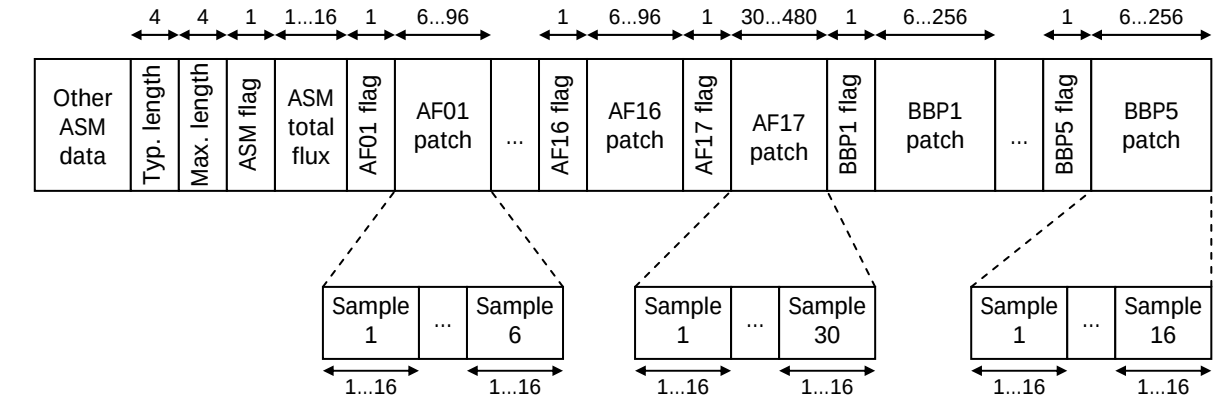


Figure 11.8: Operation diagram of the Fully Adaptive Encoder

11.3.5.2. Data frames

Figure 11.9 describes the data frames generated using this encoding scheme. Field lengths, here shown in bits, will be variable within the ranges specified. Again, this data frame scheme is valid for Astro-1. For Astro-2 the number of samples for every patch of the BBP will be only 10, so the maximum size of a BBP patch will be 160 bits.



Minimum frame size: **188 bits** (**5.7%** of F16E size).
Maximum frame size: **3343 bits** (**100.9%** of F16E size).

Figure 11.9: Data frames generated with a Fully Adaptive encoding

11.3.5.3. Performance estimations

The performance of this coding scheme is really difficult to estimate, and some simulations (including cosmic rays) should be done. However, taking into account table 11.1, an approximate calculation can be done. First of all, the sizes (in bits) needed to code a given flux value have to be found, this is, S_{max} obtained from table 3.12 of ESA (2000):

G [mag]	S_{max} [e^-]	$Counts_{max}$ (16-bit ADC, 330.000e^- dynamic range)	Bits needed	Fraction of all stars
<15	330000	65535	16	4%
15 – 16	5000	993	10	4%
16 – 17	2000	398	9	8%
17 – 18	750	149	8	14%
18 – 19	300	60	6	25%
19 – 20	150	30	5	45%

Table 11.2: Bits needed to code fluxes with 5e^- resolution

The lengths in bits shown in table 11.2 have been calculated assuming a resolution of 5,035e^- (330.000$e^-/2^{16-1}$). Hence, with the *typical* lengths shown in this table, the average size of a flux measurement will be 0.04×16 + 0.04×10 + 0.08×9 + 0.14×8 + 0.25×6 + 0.45×5 = 6.63 bits. The global budget (average sizes) is the following:

- Typ/Max lengths flags: 4+4=8 bits
- Typ/Max coding flags: 1 (ASM) + 17 (AF) + 5 (BBP) = 23 bits
- ASM total flux: 6.63 bits
- AF01 to AF16: 16 patterns of 6 sample each, 16×6×6.63 = 636.48 bits
- AF17: 30 samples, 30×6.63 = 198.9 bits
- BBP (Astro-1): 5 patterns of 16 samples each, 5×16×6.63 = 530.4 bits
- BBP (Astro-2): 5 patterns of 10 samples each, 5×10×6.63 = 331.5 bits

TOTAL: 1403.41 bits (Astro-1) or 1204.51 bits (Astro-2), for measured flux data of 1 source. Compression ratio (wrt full 16-bit encoding, no compensation): **2.36**.

We must remember that this compression ratio corresponds to the "best case". However, cosmic rays effects will be lower with this coding scheme because of the "maximum length" analysis. The problem of this coding scheme is the resources needed for its implementation: a sequential operation is not allowed. When a source is detected, all the CCDs have to be read first, and after this all the calculations and operations have to be done. Also, many operations have to be executed in order to code all the flux data for a given source.

11.4. MODEL-BASED ENCODING

11.4.1. General description

Point-like sources in Gaia do not produce point-like images over the focal plane, due to the PSFs or Point Spread Functions. That is, a point-like source appears with a given profile when projected over the focal plane, and this profile is the measurement to do (the *patterns* or *patches* read all over the astrometric field and the BBP). In this profile, a very high percentile of the flux of the source is concentrated in a small zone, the center of the PSF. Therefore, the *wings* of the PSF will not have as much energy (flux) as the center, so the measurements on the borders of a patch will lead to much smaller fluxes. In fact, some realistic PSF models lead to a factor 3.5 between PSF center and its wings (just at the border of a standard AF patch).

In order to code flux data, some adaptation can be done taking into account the global flux of a source, but also taking into account its profile. That is, one can take into account the 3.5 factor between PSF center and its wings, and therefore to use less bits to code the wings. This is the basic principle of model-based encoding. However, this codification has some problems: point-like sources will follow the model, but it is not clear what will happen with binary (*visual* binaries) and slightly extended sources. Not many of the sources scanned by Gaia will be binary or extended, but they should also be correctly coded. If the model is strictly followed during coding operation, these sources (as well as cosmic rays) would imply data loss, so some kind of "security system" (flags) have to be included.

Finally, for this encoding technique, and in order to obtain a higher compression ratio (and a feasible implementation within the baseline), no resolution compensation will be done. That is, the encoding task will be just to take the appropriate Least Significant Bits (LSBs) of the data. Also, better adaptation ranges have been selected, although this could lead to a higher penalty due to cosmic rays. Other adaptation ranges leading to better results should be found via simulation.

11.4.2. Assumptions

This is the coding scheme with more strict assumptions:

- The PSF has to be roughly constant all over the focal plane.
- Sources have to be point-like.
- TDI rates and attitude should have very small variations: if the PSF is not well centered in some of the patterns the compression ratio will be much lower.

11.4.3. Operation

1. Obtain the ASM flux data.
2. Depending on ASM flux data, the number of bits to be considered from the A/D conversions (i.e. the number of LSBs transmitted for every sample) will be selected:
   a) Magnitude<15 (Charge>10.000e^-): 16 bits.
   b) 15≤Magnitude<17 (10.000e^->Charge>750e^-): 11 bits.

c) 17≤Magnitude<18 (750e^->Charge>300e^-): 8 bits.

d) 18≤Magnitude<19 (300e^->Charge>150e^-): 6 bits.

e) Magnitude≥19 (Charge<150e^-): 5 bits.

3. Furthermore, the following most significant bits will be removed from every patch:

   a) In AF01 to AF16: 1 MSB from the samples at the borders (i.e., samples 1 and 6 of every patch). For example, if the charge is lower than 150 e^-, only 4 bits will be taken for these samples. This is due to the model: if the wings of the PSF are 3.5 times fainter than the center, then one bit can be saved.

   b) In AF17: 2 MSB from the 5 samples at every border (i.e., samples 1 to 5 and 26 to 30), and 1 MSB from the following 5 samples (i.e., samples 6 to 10 and 21 to 25). This is due to the wider area covered: the PSF wings will be much fainter than the center.

   c) In BBP1 to BBP5: 2 MSB from the 2 samples at every border, and 1 MSB for the next 2 samples at the border.

4. CCD analog data is verified:

   a) If the charge measured for that CCD (for the whole pattern) remains within the predicted margin plus a 10%, then the "adaptive" coding is used and it is marked assigning "1" to the flag for that pattern. For example, if ASM flux was 9.999e^- (or lower) and the flux read for that CCD is 10.999e^- (or lower). Also, samples using fewer bits (i.e., border samples) must have a flux value lower than this limit.

   b) If some of the samples of that patch cannot be coded with the selected bit pattern, full 16-bit coding is used for the whole pattern and it is marked assigning "0" to the flag. For example, if ASM flux was 670e^- and the flux read for that CCD is 900e^-.

Figures 11.10 to 11.13 show the bit pattern to be used for every patch of the focal plane (using as an example a source with 17≤M_V<18, so center samples will use 8 bits). Left panels (or top panel, in fig. 11.11) show the number of bits to use in every sample, while right panels (or bottom panel, in fig. 11.11) show the profile of maximum values allowed –so these last figures should be the ones fitting every PSF. The range verifier will just have to compare the flux values of every patch with these patterns. Each and every one of these patterns is based only on estimations of the PSF. They should be verified and improved comparing them to the latest PSF models, in order to obtain a close fit of every bit pattern to the PSF pattern, and thus obtaining the lowest telemetry consumption. This improvement shall be done in further studies, when all the results of the Gaia simulator will be available.

This coding scheme is similar to the Modified "Adaptive" Encoding (MAE), but Model-based Encoding leads to a higher compression ratio. However, the resolution is lower and constant, while in MAE resolution compensation was obtained for the faintest stars (although this feature is out of the baseline). Furthermore, cosmic rays, bad tracking and other perturbations will lower the compression ratio. Their effects should be analyzed via simulation.

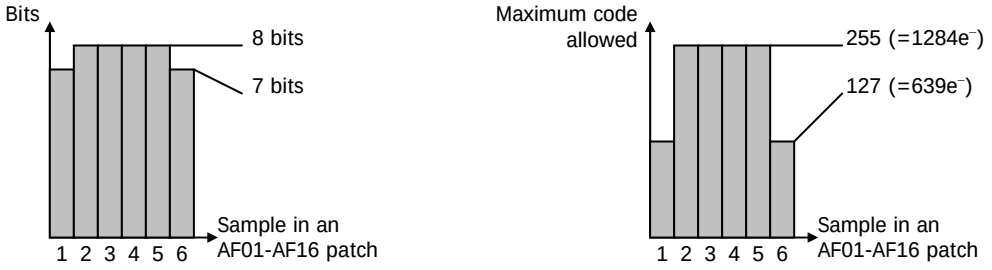


Figure 11.10: Bit pattern for AF01–AF16 patches

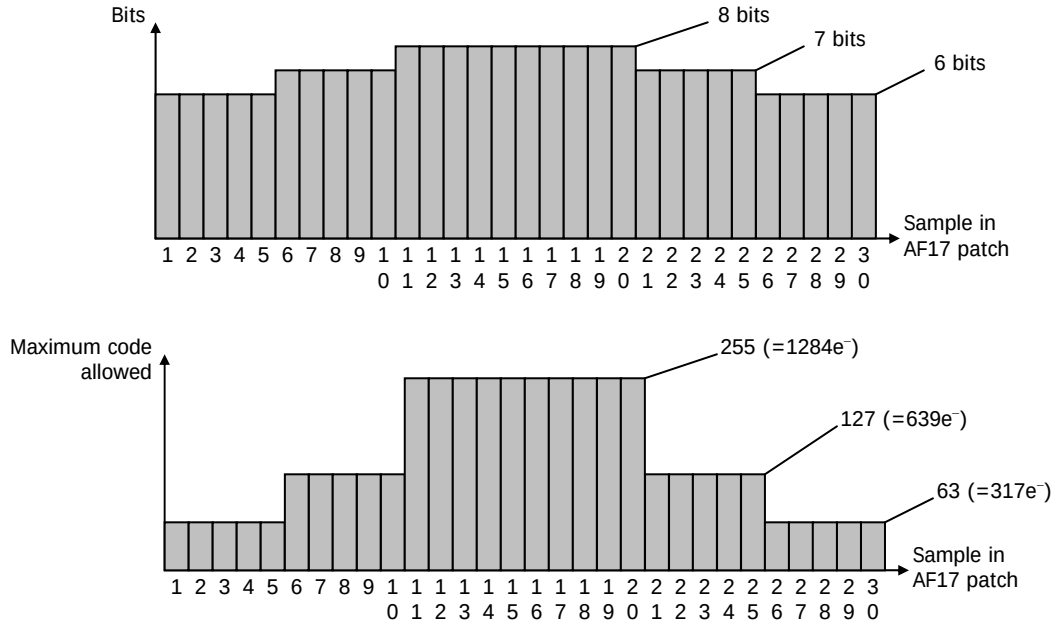


Figure 11.11: Bit pattern for AF17 patches

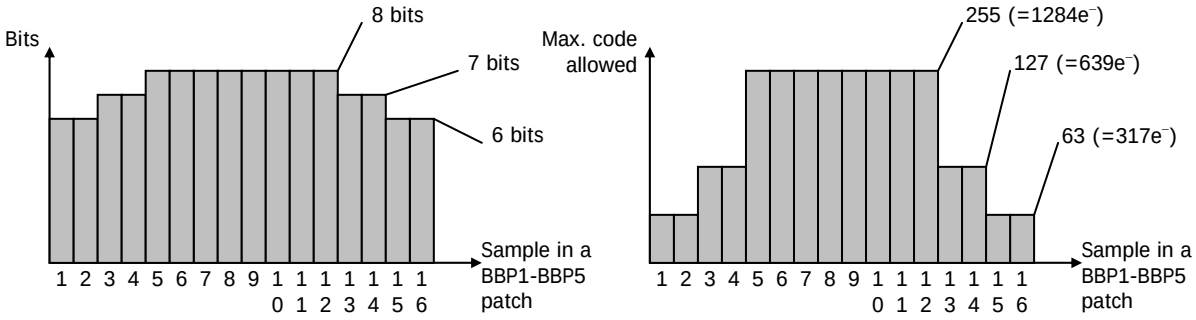
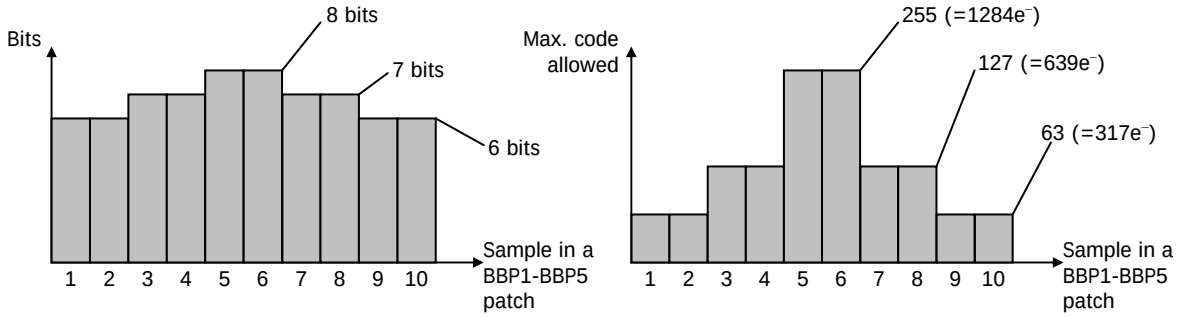


Figure 11.12: Bit pattern for BBP1-BBP5 patches in Astro-1



(Samples in the Astro-2 BBP are 6 times wider than in the Astro-1 BBP)

Figure 11.13: Bit pattern for BBP1-BBP5 patches in Astro-2

This encoding scheme is valid all over the focal plane except for the ASM, where flux data has to be coded in full 16-bit format. However, some kind of model-fit for the ASM3 patch could also be considered. Figure 11.14 illustrates a possible implementation.

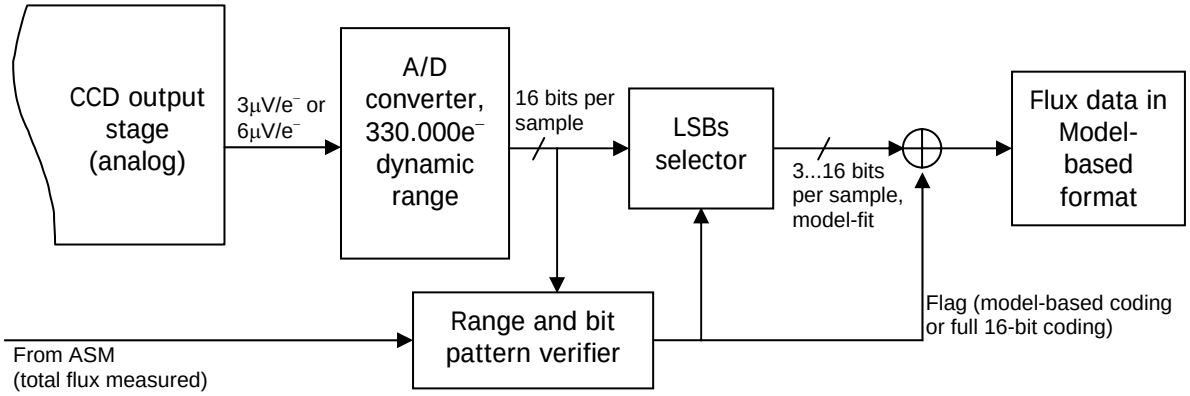
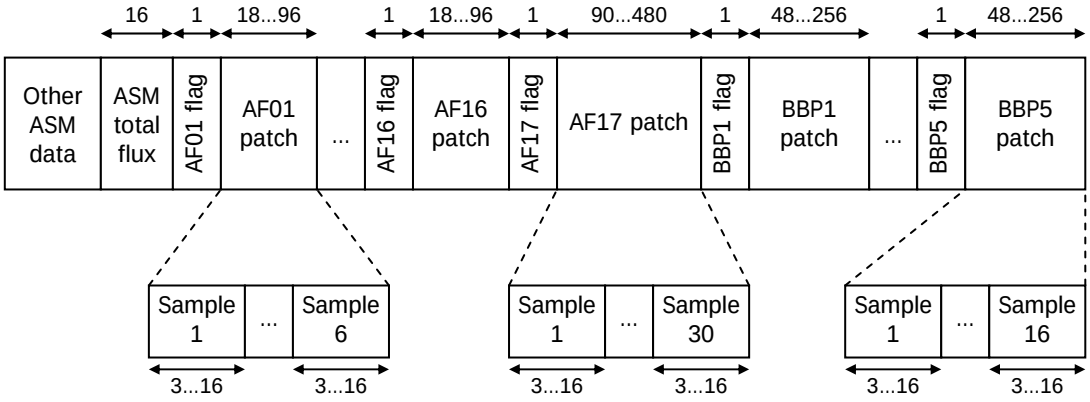


Figure 11.14: Model-based encoding scheme

11.4.4. Data frames

The following figure describes the data frames generated using this encoding scheme. Field lengths, here shown in bits, will be variable within the ranges specified:



Minimum frame size: **656 bits** (**19.8%** of F16E size).
Maximum frame size: **3334 bits** (**100.7%** of F16E size).

Figure 11.15: Data frames generated with a Fully Adaptive encoding

Again, this data frame scheme is valid for Astro-1. For Astro-2 the number of samples for every patch of the BBP will be only 10, so the maximum size of a BBP patch will be 160 bits. The overhead due to flags is 0.7%, the same obtained with MAE.

11.4.5. Performance estimations

The Galaxy model shown in table 11.2 will be used again to calculate an estimate ("best case") of the reliability of this encoding scheme. MSBs subtracted from the borders of every patch will be taken into account here. First, average sample sizes will be found:

- ASM total flux: 16 bits.
- AFxx or BBPx center sample: 0.04×16 bits + (0.04+0.08)×11 bits + 0.14×8 bits + 0.25×6 bits + 0.45×5 bits = 6.83 bits
- AFxx or BBPx border sample (–1 MSB): 0.04×15 bits + (0.04+0.08)×10 bits + 0.14×7 bits + 0.25×5 bits + 0.45×4 bits = 5.83 bits
- AF17 or BBPx border sample (–2 MSB): 0.04×14 bits + (0.04+0.08)×9 bits + 0.14×6 bits + 0.25×4 bits + 0.45×3 bits = 4.83 bits

Now we can find the average patch sizes:

- AF01 to AF16 pattern: 4×6.83 bits (pattern center) + 2×5.83 bits (pattern borders) + 1 bit (flag) = 39.98 bits
- AF17 pattern: 10×6.83 bits (pattern center) + 10×5.83 bits (pattern borders) + 10×4.83 bits (pattern borders) + 1 bit = 175.9 bits
- BBP pattern (Astro-1): 8×6.83 bits (pattern center) + 4×5.83 bits (pattern borders) + 4×4.83 bits (pattern borders) + 1 bit = 98.28 bits
- BBP pattern (Astro-2): 2×6.83 bits (pattern center) + 4×5.83 bits (pattern borders) + 4×4.83 bits (pattern borders) + 1 bit = 57.3 bits

Finally, total size (average) can be found easily:

- ASM total flux: 16 bits
- AF01-AF16: 16 columns of 39.98-bit patterns. Hence, 16×39.98=639.68 bits
- AF17: 175.9 bits
- BBP (Astro-1): 5 columns of 98.28-bit patterns. Hence, 5×98.28=491.4 bits
- BBP (Astro-2): 5 columns of 57.3-bit patterns. Hence, 5×57.3=286.5 bits

TOTAL: 1322.98 bits (Astro-1) or 1118.08 bits (Astro-2), for measured flux data of 1 source. Compression ratio (wrt full 16-bit encoding, no compensation): **2.52**.

This compression ratio is the "best case", but it could be lowered due to cosmic rays, attitude errors or non-point-like sources. These effects could be lowered (and therefore the compression ratio would be much close to this "optimal" one) if MbE (Model-based Encoding) and FAE would be mixed. That is, a Fully Adaptive Encoding with model-compensation in the borders of the patches. This option will not be studied in detail in this chapter due to the similar implementation with FAE. Resulting compression ratio with a *Fully Adaptive Model-based Encoder* (FAMbE) would be, more or less, the same than the one obtained with MbE (at least as the "best case"). The improvement is that *real* compression ratio with FAMbE would be much closer to the ideal, that is, cosmic rays and other perturbations would be treated in a better way.

The resources needed to implement MbE (Model-based Encoding) are very low because no resolution compensators are needed at the output of the CCDs. Furthermore, sequential operation is allowed. Therefore, this is the most suitable coding strategy if on-board available resources are very poor. Optimal parameters for this codification (flux adaptation levels and PSF adaptation pattern) should be found via simulation.

11.5. DIFFERENTIAL ENCODING

11.5.1. General description

The idea of a differential encoding is focused specially on the Astrometric Field: a stellar source being scanned by Gaia, after being detected by the ASM, will be measured 16 times along AF01 to AF16 in order to get a very high centroid precision. If the attitude and the TDI rates are well controlled (and this is not a very strict assumption), these 16 consecutive measurements should be very similar, specially consecutive measurements (AF01-AF02, AF09-AF10...). Therefore, the differences between one AF pattern and the following one should be very small. If one takes the flux differences between patterns as the data to be coded, instead of absolute flux data, then the required data lengths should be much smaller. This is the main idea of differential encoding, although it can be implemented in several ways.

The only problem of this coding scheme is its applicability: it can be applied only in AF01 to AF16 (and, in fact, AF01 cannot take advantage of this coding scheme), because continuity and pattern uniformity is needed in order to have small differences between consecutive patches. AF17 has a different geometry, so it is impossible to use a differential coding wrt AF16. Also, BBP columns have a different filter each, so patterns will probably be different between consecutive patches. However, some tests (simulations) should be done on BBP patterns to confirm this.

11.5.2. Assumptions

The only assumption is the continuity between consecutive AF patches. To get this, TDI rates and satellite attitude variations have to be small enough during 1 second –the time between the readouts of two consecutive patterns, and this is not a very strict assumption. Also, some problems could be found with cosmic rays: they can make one pattern very different from its neighbors, so differences could become large. Some simulations have to be done in order to determine the effect of cosmic rays in the compression ratio obtained.

11.5.3. Reference Encoding

In a differential encoding system the first sample is coded in an absolute manner, and after this all the remaining samples can be coded in a differential way. Here *reference encoding* stands for AF01 coding, that is, codification of the reference to all the other patterns. This reference could be coded as full 16-bit. However, some bits can be saved if a *fully adaptive encoding* or a *model-based encoding* is used, not only for the AF01 but also for all the rest of data that cannot be encoded in a differential manner. The reason of this recommendation is obvious: FAE and MbE offer, for the moment, the best lossless compression ratios. Therefore, the recommendation is to encode ASM, AF17 and BBP flux data with a Fully Adaptive Encoder (even with FAMbE, if possible) if non-sequential operation is allowed, or with a Model-based Encoder if operation must be sequential and resources available are lower. In the following subsections, FAE will be considered for non-sequential differential encoders, and MbE for sequential ones.

11.5.4. Basic Differential Encoding

This is the first (and easiest) scheme of differential encoding: for every pattern measured (i.e., for AF02 to AF16), the values obtained (i.e., the 6 samples obtained) have to be coded in a differential way. That is, the real value (not the differential one) of the previous pattern has to be subtracted to the real value measured in the pattern being codified. This subtraction or difference has to be done in a per-sample basis, subtracting to sample *i* the value of sample *i* of the last pattern. Details are given in the following subsections.

This is the basic operation, getting a feasible codification with much less bits per sample (i.e., per *differential* sample). However, sometimes the difference may be higher than the maximum available with this reduced number of bits (e.g., if some cosmic ray is detected). Then, some mechanism has to be developed in order to guarantee data integrity. This mechanism is the absolute encoding described in the previous section (11.5.3). Therefore, if some sample of a pattern is too *different* of the previous one, the whole pattern will be coded with MbE (including its flag), and the *differentiators* will be reset. A flag will indicate the way in which a pattern is encoded: BDE or MbE.

It is also important to note that 6 independent *differentiators* are needed in order to implement this encoding scheme. However, this is not a problem in terms of resources needed: a *differentiator* is, in fact, a simple subtraction. Finally, these differentiators will be reset for every source being analyzed (i.e., after AF01 has scanned it), and also after a MbE encoding (i.e., after some cosmic ray, etc.).

11.5.4.1. Operation

Figure 11.16 shows the flux diagram with the operation of the Basic Differential Encoder. The notation used in this diagram is described below.

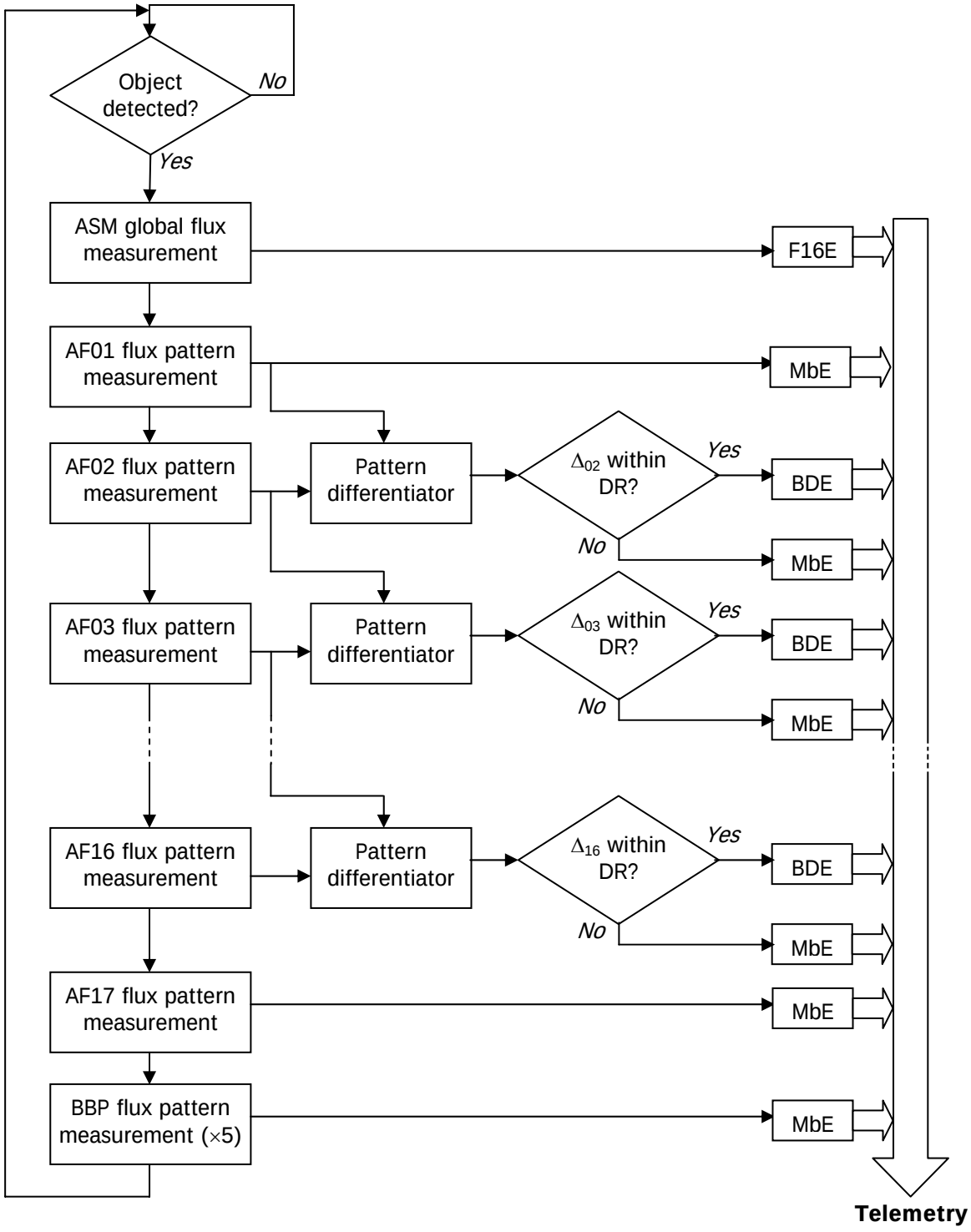


Figure 11.16: Operation diagram of the Basic Differential Encoder

AF_{XX}: pattern absolute data (6 samples for AF01-AF16, 30 samples for AF17).

Δ_{xx}: AF_{XX} pattern differential data ($\Delta_{xx} = AF_{XX} - AF_{XX-1}$), for XX = 02 to 16. An operation schematic (example with AF07 and AF08) is shown in figure 11.17.

BBPx: pattern data (16 samples in Astro-1, 10 samples in Astro-2).

F16E: Full 16-bit Encoding.

MbE: Model-based Encoding.

BDE: Basic Differential Encoding. That is, to encode Δ_{xx}.

DR: Dynamic Range = -2^b to 2^b-1, where *b* is the number of bits used for differential encoding (excluding sign bit).

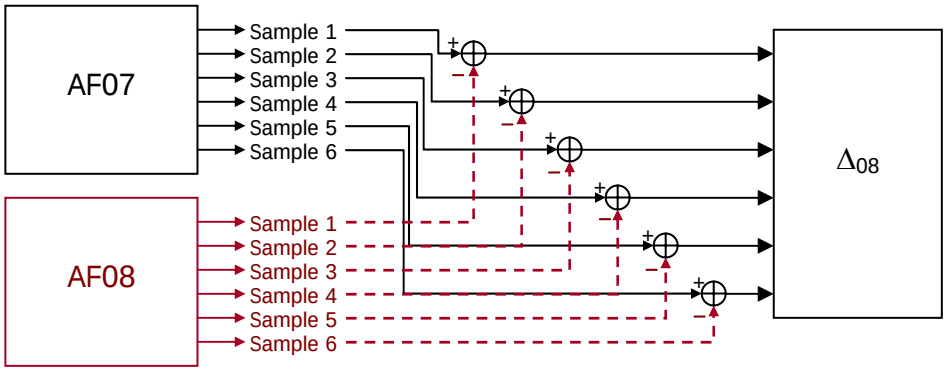
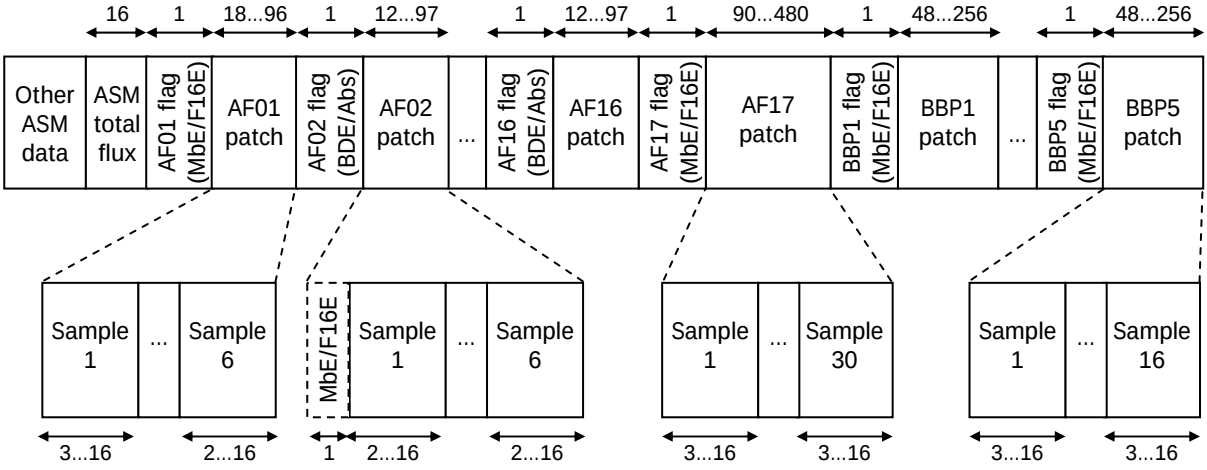


Figure 11.17: Operation schematic for a pattern differentiator

11.5.4.2. Data frames

The coding scheme used in every AFxx (AF02 to AF16) will be indicated with a 1-bit flag: coded with BDE or coded in absolute value (MbE or F16E). The rest of patches (AF17 and BBPx) will be coded with MbE, including its flag. When an AF patch (except AF17) cannot be coded in a differential way, a flag will be included in order to note whether a MbE or F16E is used. The number of bits to be used for the differential encoder is still TBD. Some simulations should be done before a decision is taken, in order to obtain an optimal value. However, the minimum number of bits to be used is 2: 1 for the sign and 1 for the absolute value of the difference (i.e., a dynamic range of –2...+1). A maximum recommended length is 10 bits (1 for the sign and 9 for the absolute value of the difference). With this maximum recommended length, the dynamic range is –512...+511, so variations up to ±2578e^- could be codified (with a 5e^- resolution).



Minimum frame size: **566 bits** (**17.1%** of F16E size).

Maximum frame size: **3349 bits** (**101.1%** of F16E size).

Figure 11.18: Data frames generated with a Basic Differential encoding

11.5.4.3. Performance estimations

Differential encoders are probably the best solution for the AF, but their performance is the most difficult to estimate: no galaxy model is valid here, and first estimations are really difficult to obtain. Their performance depends on attitude noise, TDI errors, CCD response, cosmic rays... so good estimations can only be obtained via simulation, as well as an optimum length for differential codes. However, some approximate maximum and minimum performances can be obtained here: taking into account the need of, at least, 2 bits to code a differential value (the best case), the recommended maximum length of 10 bits, and the average size of MbE data fields (obtained in section 11.4.5), these results can be obtained:

- MbE/F16E coding flags: 17 (AF) + 5 (BBP) 1-bit flags = 22 bits. 15 of these 22 bits will be included only if absolute coding is required (see figure 11.18), so only 7 bits will be considered for the best case.
- BDE/Absolute coding flags: 15 (AF02 to AF16) 1-bit flags = 15 bits
- ASM total flux: 16 bits
- AF01: 6 samples (MbE coding), 39.98 bits
- AF02 to AF16: 15 patterns of 6 samples each, 2 to 10 bits per sample (best case: BDE coding), 15×6×(2 to 10) = 180 to 900 bits
- AF17: 30 samples (MbE coding), 175.9 bits
- BBP (Astro-1): 5 patterns of 16 samples each (MbE coding), 491.4 bits
- BBP (Astro-2): 5 patterns of 10 samples each (MbE coding), 286.5 bits

TOTAL: 925.28 to 1645.28 bits (Astro-1), or 720.38 to 1440.38 bits (Astro-2), for flux data of 1 source. Compression ratio (wrt full 16-bit encoding, no compensation): **1.99 to 3.75**.

The lower limit of this compression ratio is lower than the ratio obtained only with FAE or MbE, although this lower limit is expected not to be reached. However, the problem is the coding of out-of-range values: they can suppose a high number of bits per data field, and therefore a much lower compression ratio. Cosmic rays, therefore, can be a serious problem in this scheme (although a minimum compression ratio of 2 is expected on average). The maximum reachable compression ratio (3.75) could be reached if a perfect scan is maintained (no attitude noise, perfect TDI scanning, and homogeneous CCD responses along the AF). In a real implementation this limit will be lower, leading to expected compression margins of 2 to 3 (to be confirmed with simulations). Moreover, this compression ratio will be much more uniform than the one obtained with FAE or MbE, where the compression is obtained through a galaxy model where faint stars represent an important group. BDE offers data compression not only with faint stars, but also with very bright stars. The resources needed to implement this encoding scheme are just some more than the ones needed for MbE: basic differential encoding is capable of a fully sequential operation, and the calculations needed do not use many resources.

11.5.5. Adaptive Differential Encoding

This coding scheme is, more or less, the same BDE (Basic Differential Encoding). The only difference is the length of the differential codes: in BDE, this length is fixed and determined by simulations and statistical studies. In Adaptive Differential Encoding, this length is continuously and automatically adapted by the system and no studies are needed. This is an important advantage because flux variations from a CCD to the next one also depend on absolute flux of the source: brighter sources will lead to important flux variations between CCDs, although the percentile of these variations is low.

The way this coding length is obtained is the following: after the ASM and the AF01 have been scanned for a given source, flux values are analyzed and an optimal differential code length is found (for example, as a fixed percentile of maximum flux). Then, this length will be

used for AF02 to AF16. This calculation method leads to a sequential processing all over the AF (i.e. pattern by pattern) if combined with MbE, although optimal compression results (even with cosmic rays) would be obtained using FAE.

11.5.5.1. Operation

The operation of ADE (Adaptive Differential Encoding) is more or less the same that BDE, just including an *optimal length calculation* just after AF01 readout, as shown in figure 11.19. The notation used is the same as in BDE (see figure 11.16).

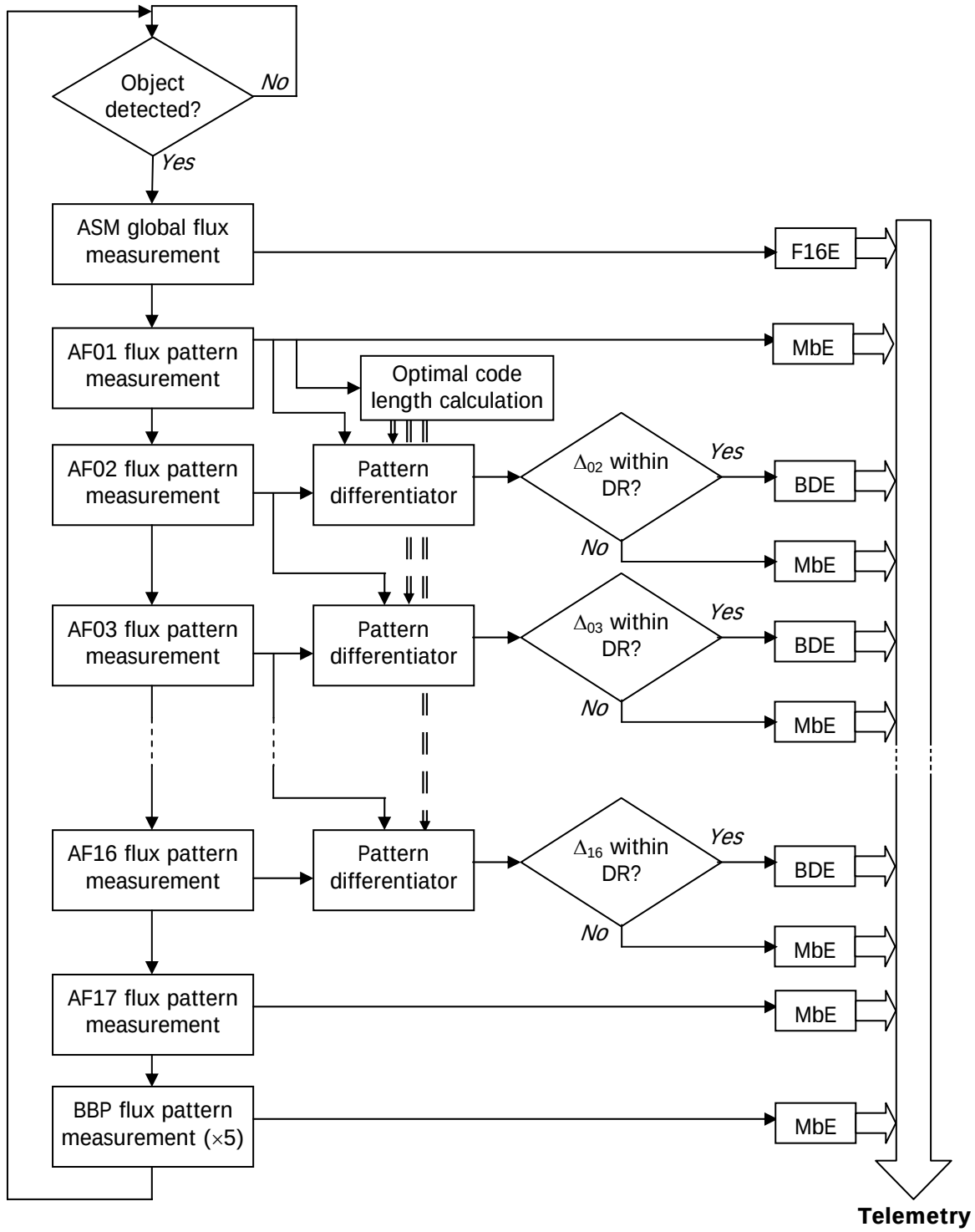


Figure 11.19: Operation diagram of the Adaptive Differential Encoder

The method of calculation of the optimal code length should be obtained via simulation. However, a possible starting point is the following: we could consider variations between CCDs of 5% of the maximum flux for a given source (to be confirmed with simulations). Then, the differential codes should be able to code this. Therefore, the following method could be applied for this calculation:

a) Measure the six samples of AF01 and take the maximum value (e.g., 167.058 e^-, or a digital value of 33.176 in 16-bit).

b) Take the 5% of this value (in the example, 1.659 in 16-bit).

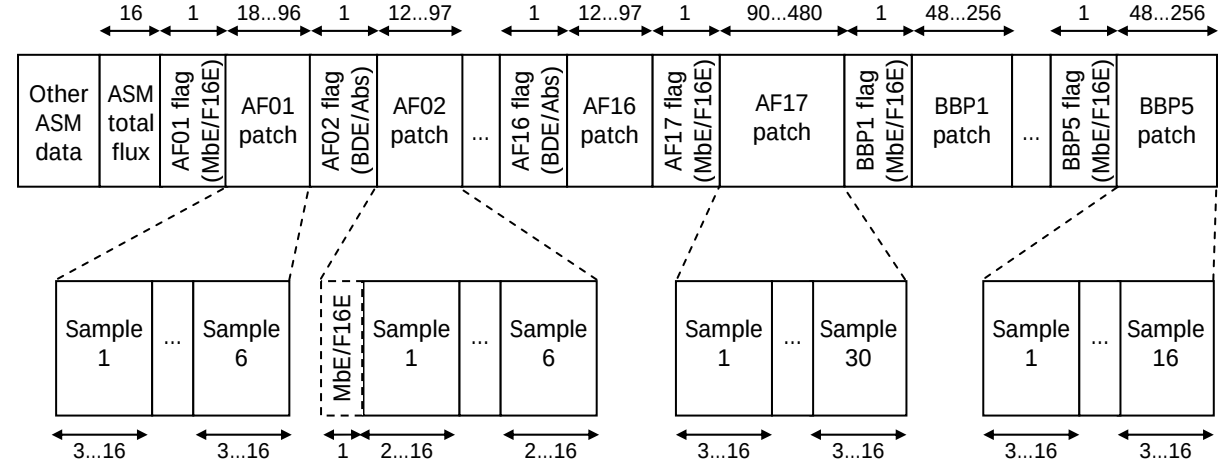
c) Obtain the number of bits needed to code this value (in the example, 11 bits, able to code up to 2.047).

d) Add 1 bit for the sign, and this will be the number of bits to use for a code. In the example 12 bits per sample will be used (+1 flag bit per patch), in front of the 16 bits needed for an absolute coding. This means a compression ratio of 1.31 (for AF02-AF16) even for this very bright source. In faint sources (e.g., 100 e^-) the compression ratio can be easily 3.8 for AF02-AF16.

This method should be improved with PSF models, this is, using still less bits to code the border samples of the patch. This will be studied in other documents and in simulations.

11.5.5.2. Data frames

The data frames for ADE will be exactly the same as for BDE. No extra flag will be needed to indicate the code length used if a fixed criterion is used (e.g., the method shown in the previous section).



Minimum frame size: **566 bits** (**17.1%** of F16E size).
Maximum frame size: **3349 bits** (**101.1%** of F16E size).

Figure 11.20: Data frames generated with an Adaptive Differential encoding

11.5.5.3. Performance estimations

The *theoretical* performance of ADE is the same as BDE (i.e., 1.99 to 3.75 compression ratio). The difference is in the *real* performance, which will be obtained with simulations and will show a better performance for ADE than for BDE (ADE adapts better to every source and will give a more uniform compression ratio). The problem of BDE (solved with ADE) is the use of a fixed code length, which gives poor compression ratio with faint sources, and still worse with brighter ones (due to possible out-of-range differential values). A more accurate estimation of the performance than the range 1.99-3.75 can be found using the galaxy model, and a first estimation of the *calculation criterion* (i.e., to consider variations of 5% of the maximum flux

of the source). The following table, obtained from table 11.2, shows the number of bits to be used (including sign bit) to code the differences along AF02-AF16:

G [mag]	S_{max} [e^-]	$Counts_{max}$ (16-bit ADC, 330.000e^- dynamic range)	5% of $Counts_{max}$	Bits needed		Fraction of all stars
				Absolute	Differential	
<15	330000	65535	3277	16	13	4%
15 – 16	5000	993	50	10	7	4%
16 – 17	2000	398	20	9	6	8%
17 – 18	750	149	8	8	5	14%
18 – 19	300	60	3	6	3	25%
19 – 20	150	30	2	5	3	45%

Table 11.3: Bits needed to code absolute and differential fluxes with 5e^- resolution

Looking at this table we can see an important improvement in the compression: in most of the cases, 3 bits can be saved wrt absolute codification. Using this in combination of the average size of MbE data fields (obtained in section 11.4.5), an approximate *real* value of the ADE compression ratio can be found:

- MbE/F16E coding flags: 17 (AF) + 5 (BBP) 1-bit flags = 22 bits. 15 of these 22 bits will be included only if absolute coding is required (see figure 11.20), so only 7 bits will be considered for the best case.
- BDE/Absolute coding flags: 15 (AF02 to AF16) 1-bit flags = 15 bits
- ASM total flux: 16 bits
- AF01: 6 samples (MbE coding), 39.98 bits
- AF02 to AF16: 15 patterns of 6 samples each. Average size per sample: 13×0.04 + 7×0.04 + 6×0.08 + 5×0.14 + 3×0.25 + 3×0.45 = 4.08 bits. Therefore, the total size for AF02-AF16 will be 15×6×4.08 = 367.2 bits.
- AF17: 30 samples (MbE coding), 175.9 bits
- BBP (Astro-1): 5 patterns of 16 samples each (MbE coding), 491.4 bits
- BBP (Astro-2): 5 patterns of 10 samples each (MbE coding), 286.5 bits

TOTAL: 1112.48 bits (Astro-1), or 907.58 bits (Astro-2), for measured flux data of 1 source. Compression ratio (wrt full 16-bit encoding, no compensation): **3.05**.

The average effect of cosmic rays and other perturbations will be slightly lower with this coding scheme, although their effect in faint sources will be very important. The resources needed to implement this encoding scheme are the same needed for BDE plus a module for the code length calculation. This last one will not need many resources, and a sequential operation will also be allowed if combined with MbE.

11.5.6. Fully Adaptive Differential Encoding

As an improvement to ADE (Adaptive Differential Encoding), FADE (Fully Adaptive Differential Encoding) obtains the optimal differential code length from every source, taking into account not only AF01 data but also AF02 to AF16 data. The problem of ADE is that there is the possibility of obtaining a wrong code length if a cosmic ray falls in AF01, or even if the criterion for the code length selection is not optimal. This is solved with FADE, which makes a statistical study of flux values in order to obtain a *really* optimal differential code length for every scan of every source. Moreover, out-of-range patterns are coded in a different way than with ADE: they are all analyzed and coded also in a differential way, but with a *maximum* differential code length –just like happened with FAE, but working always with

differential values. Therefore, AF02-AF16 patterns are always coded differentially, but using a *typical* (optimal) or *maximum* length. The other patterns (AF01 or *reference*, AF17, BBP1-BBP5 and ASM global flux value) will be coded with FAE (or with FAMbE, if possible).

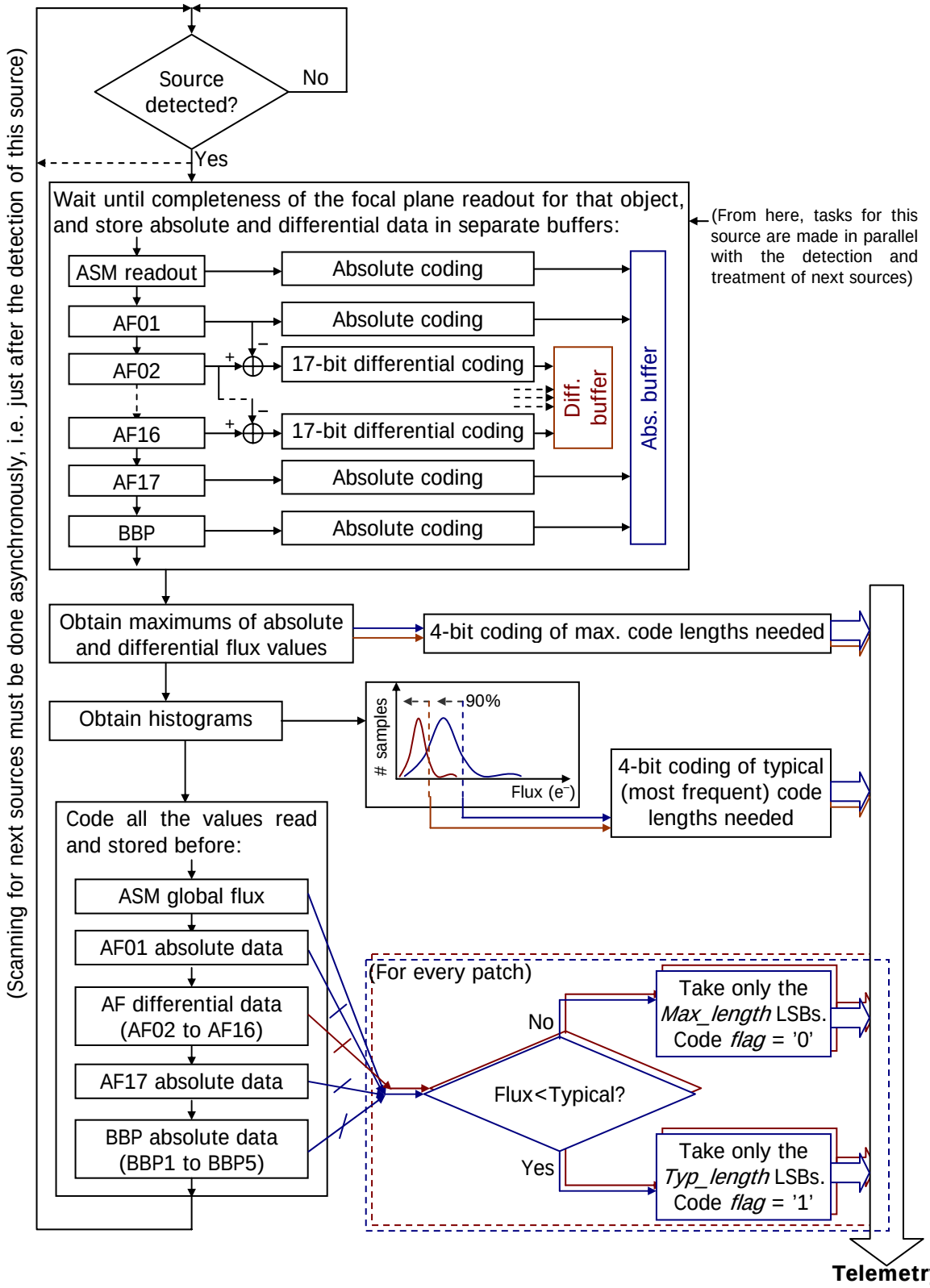


Figure 11.21: Operation diagram of the Fully Adaptive Differential Encoder

The only problem of this coding scheme is the resources required: as in the case of FAE, every source has to be scanned in the whole focal plane before being analyzed and coded, and this could lead to a higher processing consumption. Moreover, sequential operation is not allowed

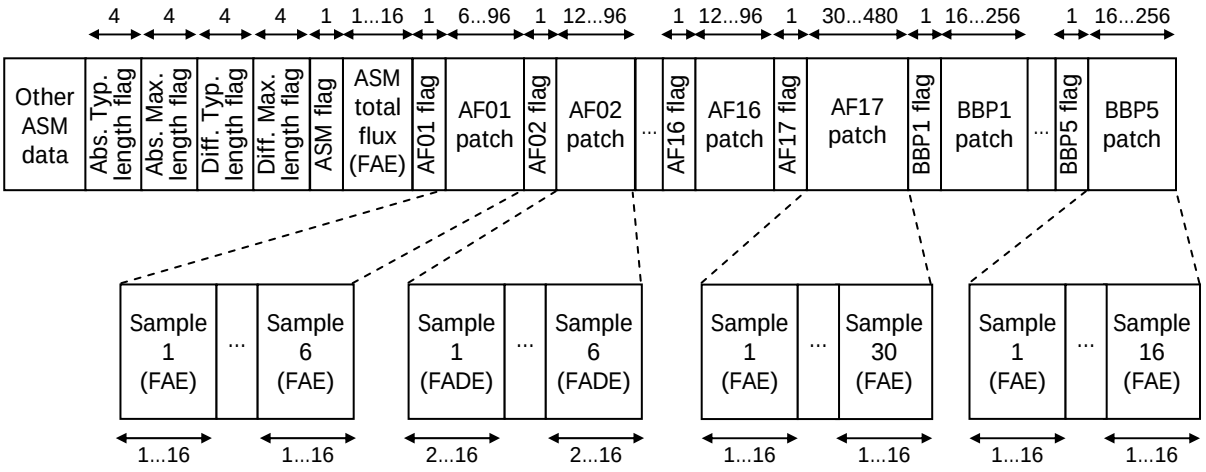
here. On-board processing capability, however, is expected to be enough to implement this coding scheme.

11.5.6.2. Operation

The operation diagram of FADE is shown in figure 11.21, which is basically the same of FAE but just including the differential blocks. In this diagram, 17-bit (1 for the sign and 16 for the value) differential calculations will be used in order to keep all the information, even in very bright sources and large variations between CCDs. Then, the *typical* and *maximum* values will be found for both groups, absolute and differential values.

There is an important detail to take into account: thanks to the *maximum differential length* analysis, there is the possibility to measure large differences that could lead to a needed code length of 17 bits (16 for the value and 1 for the sign). If this happens (i.e., if differences larger than ±32767 are measured), then an absolute F16E coding will be used for these patches. This will be marked assigning a value of 15 to the *maximum differential length* (this is, its maximum possible value), and this will mean that all the differential patches that cannot to be coded with a *typical* differential length will be coded in absolute F16E. In order to implement this, also absolute values measured in AF02-AF16 will have to be stored in the temporary buffers.

11.5.6.3. Data frames



Minimum frame size: **336 bits** (**10.1%** of F16E size).

Maximum frame size: **3351 bits** (**101.2%** of F16E size).

Figure 11.22: Data frames generated with a Fully Adaptive Differential encoding

The 1-bit flags shown in figure 11.22 indicate whether a maximum or a typical code length is used for every patch. The type of codification depends only on the patch: AF01, AF17 and BBP will be coded always with FAE (i.e., absolute value), while AF02 to AF16 patches will be coded with FADE (i.e., differential value, except for large variations which will be coded in absolute F16E value).

11.5.6.4. Performance estimations

As seen in ADE, the maximum differential code length is expected to be lower than 13 bits. With this assumption (TBC with detailed simulations), and using the results obtained with FAE estimations, the expected performance will be within the following margins:

- Typ/Max differential (FADE) lengths flags: 4+4=8 bits

- Typ/Max absolute (FAE) lengths flags: 4+4=8 bits
- Typ/Max FADE coding flags: 15 bits (AF02-AF16)
- Typ/Max FAE coding flags: 1 (ASM) + 2 (AF01/AF17) + 5 (BBP) = 8 bits
- ASM total flux: 6.63 bits
- AF01: 6.63×6=39.78 bits
- AF02 to AF16: 2-13 bits each (1 sign + 1-12 diff.), ×6×15 = 180-1170 bits
- AF17: 30×6.63 = 198.9 bits
- BBP (Astro-1): 5 patterns of 16 samples each, 5×16×6.63 = 530.4 bits
- BBP (Astro-2): 5 patterns of 10 samples each, 5×10×6.63 = 331.5 bits

TOTAL: 994.71-1984.71 bits (Astro-1) or 795.81-1785.81 bits (Astro-2), for measured flux data of 1 source. Compression ratio (wrt full 16-bit encoding, no compensation): **1.63-3.44**.

Although this range seems worse than the one obtained with ADE, cosmic rays effects will be lower with this coding scheme because of the "maximum length" analysis, and real results will be better. Also, no *calculation criterion* has to be obtained for this scheme: optimal code length will automatically be obtained, so its results will be always better than ADE because FADE will operate (and auto-calibrate) always with *real* data. Simulations have to be done in order to obtain its average compression ratio, although it is expected to be slightly higher than 3. Finally, the performance of FADE could be improved using a FAMbE algorithm instead of FAE for the coding of absolute values.

11.6. SUMMARY OF CODIFICATION SCHEMES

	Resolution	Data integrity	Assumptions	Operation	Compression ratio	Overhead	On-board require-ments	Others
Full 16-bit Encoding:								
Full 16-bit, baseline.	5e^-	Yes	None	Direct coding from ADC output	1.0	0.0%	Minimum	
Full 16-bit, compen-sation.	Up to 1 e^-	Yes (DR flags)	None	Compensation previous to ADC, direct coding from ADC output.	<1.0 (DR flags)	0.7%	Resolution compensators at the output of every CCD.	
Adaptive Encoding:								
Original "Adaptive"	1.5 to 5 e^-	No	Galaxy model. Low flux variations along focal plane	Fainter stars coded with a higher resolution and less bits.	1.6	0.0%	Resolution compensators at the output of every CCD.	Compression obtained with Galaxy model (depends on star).
Modified "Adaptive"	2.9 to 5 e^-	Yes	Same	Fainter stars coded with a higher resolution and less bits. DR is verified.	1.7	0.7%	Resolution compensators, range verifiers.	Same
Fully Adaptive	5e^-	Yes	Same	Measurements analyzed, optimal encoding lengths used.	2.4	0.9%	Sequential operation not allowed (DR analysis *a posteriori*).	Same

Table 11.4: Summary of codification schemes (I)

	Resolution	Data integrity	Assumptions	Operation	Compression ratio	Overhead	On-board require-ments	Others
Model Based Encoding:								
Model Based	$5e^{-}$	Yes	Galaxy model. PSF model. Point-like and well-centered sources. Low flux variations.	Fainter stars coded with higher res. and less bits. PSF wings coded with less bits.	2.5	0.7%	Range verifiers.	Compression obtained with Galaxy model (depends on star). Problems with double or extended sources. Very sensitive to cosmic rays and PSF miscenterings
Differential Encoding:								
Basic Differential	$5e^{-}$	Yes	Very low flux variations between consecutive patterns: Good attitude and TDI control.	Coding of the difference between consecutive patterns: less bits needed (code length fixed).	2.0 – 3.8 (TBC)	≤1.1%	6 differentiators per CCD (AF02-AF16). Sequential operation allowed if MbE is used for absolute coding.	Compression obtained almost in a continuous way (no Galaxy model needed). Problems with cosmic rays.
Adaptive Differential	$5e^{-}$	Yes	Same	Pattern differences coding. Code length estimated from the maximum flux of every star.	3.0	≤1.1%	6 differentiators per CCD (AF02-AF16). Optimal code length calculation module. Sequential operation allowed if combined with MbE.	Continuous compression. Cosmic rays effect is slightly reduced.
Fully Adaptive Differential	$5e^{-}$	Yes	Same	Pattern differences coding. Optimal code length obtained for every star.	1.6 – 3.4 (TBC)	1.2%	6 differentiators per CCD (AF02-AF16), histogram calculation. Sequential operation not allowed.	Continuous compression. Cosmic rays effect is reduced. Optimal code length used always.

Table 11.5: Summary of codification schemes (and II)

With the results summarized in table 11.5, but always taking into account that these are just estimations (simulations must be done), the following can be considered the *best* simple codification schemes for Gaia:

- ADE (Adaptive Differential Encoding): offering the highest compression ratio and the best adaptation to every source, if sequential operation is needed and on-board resources are very limited. Compression ratio about 3 is expected for Astro flux data.
- FADE (Fully Adaptive Differential Encoding): this solution, only available if non-sequential operation per star is permitted and on-board resources are enough, offers the highest compression ratios without the need of any compression algorithm like Rice, LZW or Huffman (and therefore with a simple implementation), although they could be applied at a later stage on-board. However, its improvement using FAMbE (Fully Adaptive Model-based Encoding) for absolute values would be very interesting, and most probably better compression results would be obtained. Also, the possibility of BBP differential coding should be studied with simulations. Compression ratios higher than 3 are expected with standard FADE, and improved versions could reach a ratio of 3.5 or higher.

11.7. SIMULATION OF PRELIMINARY COMPRESSION SYSTEMS

In the previous sections of this chapter we have described several codification systems (focused on the flux data of Astro) that can "manually" compress the data up to acceptable ratios, even above 3 (theoretically). Recalling that the work described in this chapter could be considered a "feasibility study" for the thesis, some simulations had to be done in order to verify the correct operation of these systems. For this, we prepared a set of software simulation tools which were applied to GASS data (although in that case it was GASS version 1, for "Gaia-1" design), in order to check their reliability with the most realistic data available at that moment. The software for preliminary data compression tests, developed in C, can be considered as the predecessor of the TM CODEC since it already performed some of the elementary operations of this software.

Not only the preliminary codification schemes described in this chapter were implemented, but also some variations of them in order to slightly improve the ratios achieved –yet without modifying the basic concepts of each scheme. Maybe the most relevant of these variations was FAMbDE, which stands for *Fully Adaptive Model-based Differential Encoder* and was the practical implementation of the option already recommended at the end of section 11.5.6.4. FAMbDE, at the same time, evolved to other modifications looking for the highest ratios, which led to FAMbDE3. This last system used 3 differentiators with independent references, so that the differentiator with the lowest bit sizes was always used –and hence it better adapted to the variations from the ideal measurements. The final results (Portell et al. 2002b) are summarized in table 11.6, where we can see that FAMbDE3 offers the best compression ratios.

Compression method	Typical comp. ratio	Max. comp. ratio
WinZip	1.9	2.4
FAMbE	2.3	2.7
FADE	2.5	3.1
FAMbDE3 (FADE modified)	2.7	3.2
FAMbDE3 + WinZip	3.0	3.8

Table 11.6: Simulation results of the preliminary data compression systems

As also noted in Portell et al. (2002b), the compression requirement for the old design of Gaia was of 7.2, which was a really impressive requirement taking into account that the data compression had to be lossless. Nevertheless, as we can see in table 11.6, we achieved very interesting results despite of using only preliminary and simple designs. Furthermore, these results confirm the estimates shown in table 11.5. On the other hand, the typical and maximum ratios achieved with a standard compression system such as WinZip is also included in table 11.6, where we can see that standard systems have nothing to do with these tailored coding schemes. Nevertheless, the combination of both solutions (WinZip after FAMbDE3) leads to even higher compression ratios. All these results showed us that we were on the good direction, so we could start all of the detailed and correctly scheduled studies before reaching the comprehensive compression analysis. In the next chapter we will describe the final results of our work.

Chapter 12

An Improved Data Compression System

In the last three chapters we have described a new simulation tool named Telemetry CODEC, a set of telemetry tests using GASS simulated data, and the preliminary data compression systems that we developed during the first stage of our work. Now we will combine this software tool with the knowledge acquired during the first data compression studies. As a result, in this chapter we describe a data compression system specifically designed for the science data of Gaia (and for its latest design). This system has been implemented and successfully tested, and the resulting compression ratios are discussed. Only the Astro instrument has been taken into account, although the main ideas could also be applied to the MBP instrument. The RVS would eventually require a completely different design to compress its data.

Our goal with this chapter (and with the work behind it) is to design and develop a simple, yet powerful data compression system, capable of offering the highest compression ratio with the minimal hardware requirements –so that it could be easily implemented onboard. This system must be tested using the most realistic telemetry data currently available (i.e., GASS data as converted by *txt2bin*). Actually, this data compression system shall be considered as a part of the full Telemetry CODEC as already noted in chapter 9. With these tests we offer a realistic estimate of the data compression ratio that can be actually achieved. Furthermore, with the statistical analyses also performed here we provide a hint on future improvements of this system (or even on developments of new systems).

We have organized this chapter as follows: section 1 of this last chapter gives an overview of our data compression system and its operation principles. The implementation and testing environment is described in section 3, while section 4 lists the several tests and the statistical results obtained. The final compression ratios achieved are listed in section 5, while in section 6 we draw our conclusions.

12.1. DESCRIPTION OF THE DATA COMPRESSION SYSTEM

12.1.1. Background

The design of the data compression system in Gaia has always been a very important issue and a true challenge, because of the large amount of data that Gaia will generate and the need of receiving it without any losses. Preliminary studies have revealed a requirement on the compression ratio of 3 to 4, which cannot be reached with the off-the-shelf lossless compression systems currently available. This requirement pushes the Information Theory to its limit, implying very detailed studies of the nature of the Gaia science data, leading to new compression systems specifically designed for this mission.

As it will be shown in the next sections, several tests with standard data compression systems such as *gzip*, *rar*, *lha*, *bzip2*, etc. reveal that it is absolutely impossible to fulfill the requirements. Actually, it is even impossible to simply obtain a compression ratio higher than 2 *only* with these systems. Moreover, it is necessary to achieve high ratios without prohibitive hardware requirements. Also, the design of completely new data compression algorithms has been discarded, not only for its complexity but also for the dubious merit of this approach –i.e., the improvement in the results would surely be not worth the effort in developing them. Finally, there already exist excellent implementations of standard compression systems (in ASICs or DSPs), including space-qualified components. For all these reasons, the best solution

that we envisaged for Gaia was to develop a *pre*-compression system rather than a full data compression system. This is, to modify the data, "adapting" them to some "standard formats" (including uniform symbol sizes), trying to make the inherent redundancies of the data evident to those standard compression systems. In other words, such a pre-compression system should emphasize the statistical behavior of the data. In this way, the compression systems could clearly take advantage of these redundancies and, therefore, offer a much higher overall compression ratio.

Such a data pre-compression system has an acceptable complexity, both in design efforts and in implementation requirements. The preliminary studies on the data compression of Gaia science data (described in the previous chapter) already used this pre-compression technique with successful results –although in those studies we intended to achieve a high compression ratio using only the "pre-compressor". The design that offered the best results, FAMbDE (Fully Adaptive Model-based Differential Encoder), was designed on the basis of the nature of Gaia data, taking advantage of the measurement repetition (i.e., the several AF measurements), the PSF model, and the Galaxy model. As a result, a compression ratio of up to 3.8 was obtained when combining FAMbDE with WinZip. Unfortunately, this result was obtained only with the old Gaia design (with 17 AF columns), where more redundancies were present and, therefore, the data could be much more compressed.

Those preliminary studies were partly based on a data compression system developed for a Spanish mini-satellite (Portell 2000), where excellent results were achieved (even beyond the requirements of the mission) by using this pre-compression approach. The differential coding was specially useful in that case, as well as for Gaia where several measurement repetitions are expected to be performed. Now, taking into account the new design of the data pre-compressor system for Gaia, these kind of solutions must be thoroughly explored. More specifically, the stream partitioning appeared as the most powerful solution, and in the next sections we will verify how this idea will offer the best results in the new design of Gaia. Finally, data predictors have been introduced in this new system, which are based on reliable models of both the instruments and the on-board TM model.

12.1.2. Operation overview

First of all, it is very important to bear in mind that our data compression system is, basically, a pre-compressor that prepares the data for being fed afterwards into a standard data compressor. This is, we have not designed a new data compression system end-to-end, but only defined how to prepare the data in order to be better compressed with standard systems.

On the other hand, we have always had in mind the usual operation of the standard compressors when designing this pre-compression system. For example, we have tried to output each of the data sub-streams as uniformly as possible, in the sense of symbol sizes – since standard compressors operate better with fixed symbol sizes and, usually, with 8-bit or 16-bit symbol sizes. Also, we took into account that most of the data compressors take advantage of extreme probabilities in the data, i.e., data fields with very steep histograms. Compressors based on (or including) Adaptive Huffman coding specially take advantage of this, assigning less bits to more probable values and vice versa. Therefore, we have included differential coding with predictors, trying to minimize the differences (which are the output codes) and, therefore, steeping the resulting histograms. All in all, it results in a pre-compressed data with high redundancies that are much more evident to the compressor. In other words, the pre-compressed data can be much more compressed than the raw input data because the compressor can finally take advantage of the intrinsic redundancies.

12.1.2.1. Stream partitioning

This is the first operation to be performed on the data in order to pre-compress them in the optimal way. While acquiring GASS simulated data, the system must redirect each of the different data fields to separate processing systems. The issue was to decide which partitions

we should do, in order to obtain a balance between the number of sub-streams, the processing requirements, and the resulting ratios after applying data compressors.

By looking at the telemetry model used by GASS (Masana et al. 2004), which can be considered as a reference for the flight-realistic case, one can see a few sets of data that imply most of the TM volume. These sets of data correspond mainly to flux data, whether of ASM, AF or BBP patches. Therefore, we should focus on compressing these data and, therefore, partition them as necessary. Taking into account the expected behavior of each of these fields, the partition was obvious:

- ASM patches,
- AF01-10 patches,
- AF11 patches, and
- BBP patches.

By managing each of these data fields separately we should obtain optimal results when applying standard data compressors afterwards. But in order to achieve even better results, we should also optimize the rest of significant TM data. By analyzing the TM model, it seems clear that the readout coordinates for the AF and BBP windows also occupy a significant part of the telemetry size, so they should be partitioned as well. Also, the TDI Offsets matrix also supposes a large set of values. And finally, the rest of TM data (ASM detection data and so on) should not be mixed with these partitions, so an additional "reference" (or "synchronism") sub-stream must be generated. Therefore, these are the rest of sub-streams that should be output, besides of the main (flux data) sub-streams listed before:

- Reference, detection and auxiliary data
- TDI Offsets
- Readout times for AF and BBP windows
- Readout positions for AF and BBP windows

12.1.2.2. Differential coding with predictors

Almost all of the science data generated by Astro is composed of series of values with usually small differences between them, or with differences that can be easily predicted. It implies that these data can be coded differentially, this is, transmitting the differences rather that the absolute values, or transmitting the differences wrt some nominal or predicted value. Figure 12.1 illustrates the general operation of a differential coder including data predictors. This general case includes predictors of several orders (i.e., depending on several samples previously transmitted), using as a reference absolute samples or even differential samples (already processed). This is, however, a generalization of this kind of data processing, which can lead to complex and richer studies on improved data compression systems. In our case, and as a first solution to the problem, we will focus on the particular case shown in figure 12.2. This case operates only with the previously transmitted sample, thus implying less processing requirements.

The only difficulty in obtaining a histogram as steepest as possible after this processing (and, therefore, obtain the highest compression ratio afterwards) is the definition of the data predictor. Every data field must be treated separately for this, leading to a different predictor for every field. The following are the several data fields handled separately by our data pre-compressor, including a brief description of the prediction process:

- **TDI Offset Matrix**: the prediction for this field should be based in the behavior of the focal plane, combining the 1-second length data sets with the nominal TDI period. Nevertheless, since the current GASS version does not generate realistic TDI matrices yet, this data field will be kept unmodified by our pre-compressor.

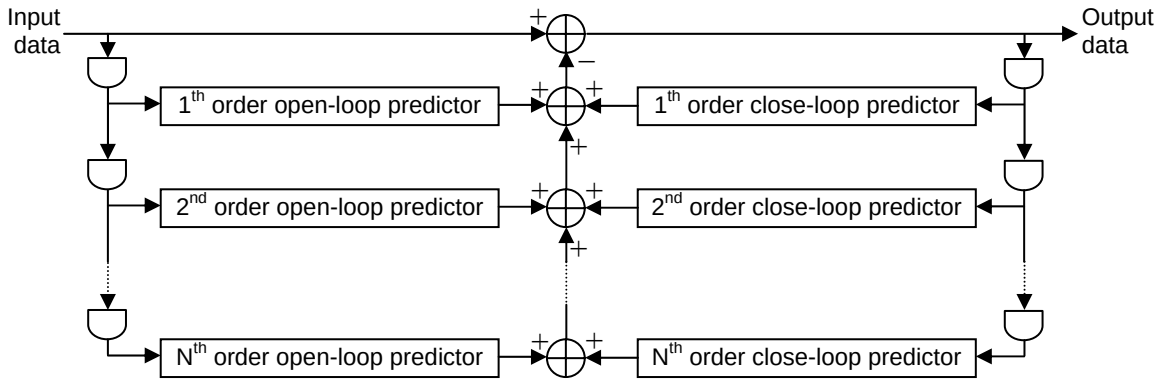


Figure 12.1: Generic structure of a differential coder including data predictors

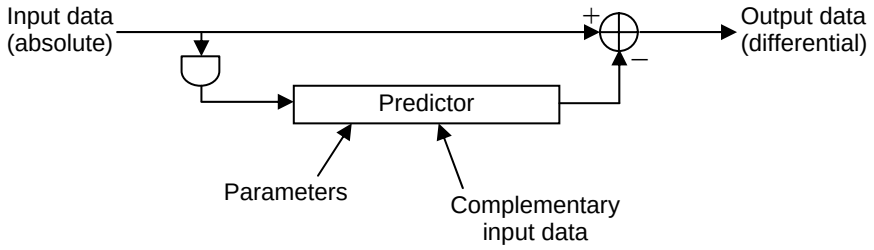


Figure 12.2: Data pre-compression structure selected for Gaia

- **ASM samples**: an excellent prediction can be done in this data field, since it will contain a 2-dimensional set of samples and, thus, will reflect the PSF shape. Therefore, the data can be transformed in a way that the differences between the samples of a star get minimal. This leads us to a different pre-compression of the data, using the simplest predictor (i.e., with no modification) but subtracting the samples in a very different order.

  The idea is to start transmitting the faintest samples and finish with the brightest ones, trying to obtain smooth flux changes between samples. Figure 12.3 illustrates the order in which we code a 5×5 ASM patch, together with some colors indicating three "groups" of samples (faintest, faint, bright, brightest). Brighter grays indicate fainter samples. This operation can be considered as a transformation of coordinates. In the right panel of this figure we also include a typical set of values that GASS outputs for ASM. It must be taken into account that the samples in the 4 borders are obtained from the numerical co-addition of 4 readout samples, which implies higher flux values in those samples.

21	1	9	2	22
3	10	17	11	4
12	18	25	19	13
5	14	20	15	6
23	7	16	8	24

401	102	124	105	404
102	112	240	126	108
124	240	1804	412	193
105	126	412	157	117
404	108	193	117	412

Figure 12.3: Coding order for the ASM samples, and an example of GASS ASM patch

- **Readout coordinates**: another excellent prediction can be done with this type of data. As indicated in Masana et al. (2004), the readout times for each of the AF and BBP windows are coded as absolute TDIs since the beginning of the data set, while the positions are coded as across-scan pixels in the CCD. This is an expensive coding, and can be significantly improved. First of all, the readout times for each window could be

predicted using the detected transit time of the ASM and the focal plane geometry. If this predicted value is subtracted to each absolute value, the resulting error should be really small. By transmitting these values we will obtain a large compression ratio afterwards. On the other hand, readout positions can also be predicted by simply using the detected across-scan coordinate in the ASM. Subtracting this to the absolute values we should obtain small values again.

- **AF01, AF11 and BBP1 patches**: the several patches that Astro will acquire along the focal plane are repeated measurements of the same object. Therefore, as previously indicated, we can differentially code the samples in order to obtain very small values. As in any other differential coding system, this one needs an initial reference, which should be AF01 for AF02-10 patches and BBP1 for BBP2-5 patches. AF11 cannot be coded wrt any other patch due to its different sampling scheme.

  A simple coding for these references could be an absolute coding, transmitting the samples unchanged. This is a valid solution when the references imply small relative sizes (such as the first sample in the ASM). But when the references get larger, we should find an alternative, optimized coding. As a first valid approach, we have selected another differential coding, but now subtracting sample to sample rather than patch to patch. This is, the 2nd sample is coded as its difference wrt the 1st sample; the 3rd sample is coded as its difference wrt the 2nd sample; and so on. This solution should also imply smaller codes than the absolute case, due to the smooth shape of the PSF.

- **AF02-10 and BBP2-5 patches**: in this case we already have the references available (i.e., AF01 and BBP1). Therefore, we can code these patches differentially wrt the previous patches. This is, each of the AF02 sample will be coded as its difference wrt the corresponding sample of AF01, and so on.

To summarize the different coding techniques used for the Astro instrument, the TDI Offset Matrix is currently unchanged (waiting for a realistic GASS output). For the ASM we apply a transformation of coordinates, thus using a complex order in the differentiator but with no predictor. Readout coordinates use a predictor based on their nominal values (based on the ASM detection and the focal plane geometry). The AF01, AF11 and BBP samples use a sample-to-sample differentiator with no predictor, and finally the AF02-10 and BBP2-5 samples use a patch-to-patch differentiator with no predictor. Figure 12.4 hereafter illustrates this process.

We have also included some additional improvements in this data pre-compression system, trying to offer a yet more uniform output for every data sub-stream. For example, it is known that a differential coding requires an additional bit for every value, to indicate the resulting sign of the operation (in order to guarantee the lossless coding). When applied to noisy data as in Gaia, these signs may repeatedly and randomly change, leading to an almost uniform probability (and, therefore, uncompressible data). If these signs were included in the same differential data, the resulting compression ratio would, for sure, decrease. This is why we decided to store (or "transmit") all of the resulting signs in a separate stream. Finally, another improvement in this pre-compression system is the output of smaller codes whenever possible. For example, if the coder detects that all of the samples in a given patch are small enough, it will code each of them in 8 bits rather than in 16 bits. It offers a pre-compressed output with a smaller size than the input, but this was not our main intention: it is done because after several tests we verified that it still improves the overall compression ratio. That is, despite we are outputting different symbol sizes it is by far compensated by the improvement in the final compression ratio.

12.1.3. flux diagram

Figure 12.4 illustrates with a flux diagram the operation of our pre-compression system, where we can see the eight final sub-streams generated. We combine the AF01 and AF02-10 sub-streams into a single "standard AF patches" sub-stream, and BBP1 with BBP2-5 into a single BBP sub-stream. The reason is that the differences in their data are not enough to make a real

difference when applying standard compressors on them, while any reduction in the number of sub-streams to compress will decrease the hardware requirements.

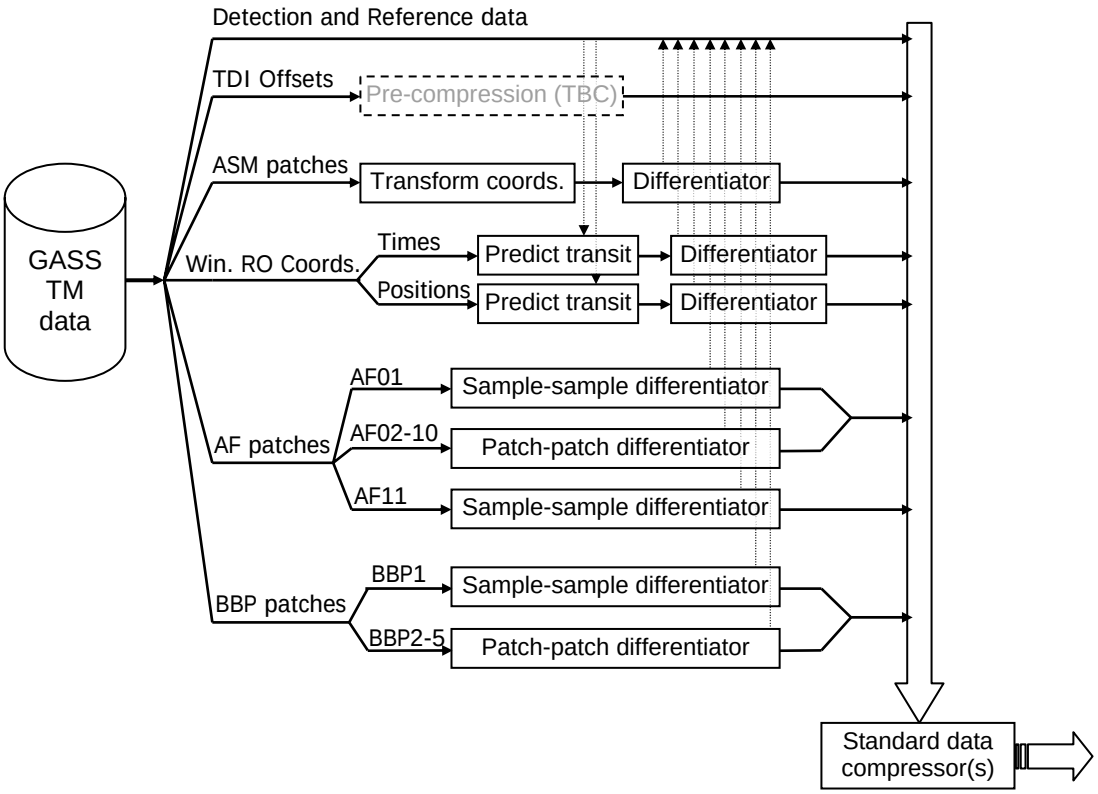


Figure 12.4: Operational flux diagram of the Stream Precompressor and Partitioner (SPaP)

The application of standard data compressors afterwards should be done in parallel, which implies 8 different (and independent) data compressors operating simultaneously. Nevertheless, in order to save resources, in the next sections we also analyze the effect of combining the several sub-streams into a single one before being compressed. If the difference is small, then only a single data compressor (with fast enough execution speed) will be needed.

Finally, the restoration process is not included in this flux diagram for the sake of clarity. This decoding process is a simple inversion of the prediction and differentiation systems, this is, a set of integrators and data processors. Its implementation would require similar resources to those needed in the pre-compression process on-board, although in this case the resource consumption would not be a problem (since the decoding will be done on ground). As it will be shown in the next sections, the decoding process has also been implemented and tested successfully.

12.2. SIMULATION ENVIRONMENT

12.2.1. Implementation of GaiaSPaP and GaiaSRaD

The data pre-compression software has been developed within the frame of the Static TM CODEC already implemented (see chapters 9 and 10). More specifically, it uses some of its internal libraries, receives the file format generated by *txt2bin*, and decodes the data in this format again. The software has been developed in C++. The implementation is fairly straightforward, simply going into the following loop while there is data to pre-compress:

- Read the Data Set (DS) header, determining the number of sources in this DS and other parameters. Then, for each source in this DS:

- Read the ASM data, transforming and coding the ASM patch.
- Read the AF and BBP data, including the window readout coordinates.
- Predict and code the readout coordinates.
- Differentiate and code the AF and BBP data.

Besides the data processing itself, the GaiaSPaP (Stream Pre-compression and Partitioning) software also analyses the data and offers a set of statistical information. More specifically, the following products are computed and given (whether as console text output or file output):

- Percentage of TM occupation of each sub-stream data, both in the input sizes and output sizes. It indicates the relevance of every data field, i.e., which data fields occupy more TM volume and, therefore, on which data fields we must focus more efforts.
- Throughput of the pre-compression process.
- Entropies (in bits) and Shannon limits (as maximum theoretical compression ratios) of every data field, before and after the pre-compression process.
- Overall entropies and Shannon limits, before and after pre-compression.
- Histograms of each data field, before and after pre-compression. For the faint stars (using a 6-sample AF patch), histograms of each sample are also offered.

All of these statistical analyses are not performed in GaiaSRaD (Stream Restoration and Decoding) software, which purely decodes and restores the single, GASS-coded TM stream.

12.2.2. Environment and standard compressors used

The TM data used for the data compression tests have been obtained from GASS version 2.2 (October 2004), after being quantized and converted to binary by *txt2bin* version 1.0.2. This *txt2bin* version is slightly different to that one used in the TM tests described in chapter 10. The major changes include the correction of a bug in the *gaia_file* library, which was causing an extra bit to be written in many of the data fields; its effect was an increment of about 1% in the output file sizes. Also a few operations have been optimised in order to offer a faster execution.

The CPU where these tests have been run is an Intel Pentium-M (with 1MB of L2 cache), at 1.4GHz and 768MB of RAM. The OS is Gentoo Linux, with a 2.6.8 kernel, except for one of the standard compressors which had to run under DOS. In that case, the OS was MS Windows XP Professional.

In order to offer a richer study, a large set of standard data compressors has been tested. Most of them are well-known archiving solutions. Here we list these data compression systems and the versions used, also including brief descriptions of each one:

- Zip 2.3 (1999/11/29): maybe the best known and most used archiving utility. PKZip and WinZip are some variants of this.
- GZip 1.3.5 (2002/09/30): the standard in UNIX and Linux systems. Based on the Lempel-Ziv (LZ77) coding used by Zip. This is basically the same system as Zip, but since this is a different implementation we also included it in the tests –mainly to compare its throughput with Zip.
- ARJ 3.10g (2004/10/14): another standard archiving tool, which had a golden age in the DOS-based computers during the last decade.
- AIN 2.32 (1996): a Transas Marine (UK) Ltd. product that competed with ARJ, usually offering better compression ratios with faster execution times. We could only find the DOS version for these tests, so it has been the only compression system tested under MS Windows.
- LHa 1.14i (2000): based on the old LHarc archiver for UNIX.

- RAR 3.40 (2004/09/08): a powerful (yet slow) archiving system often used in MS Windows environments (but with an equivalent Linux implementation).
- BZip2 1.0.2 (2001/12/30): it combines the Burrows-Wheeler block sorting compression algorithm with Huffman coding, offering better compression than LZ77/LZ78-based systems. Since it is focused on giving an excellent statistical compression it appeared as a good candidate for our purposes.
- SZip 1.12 (2000): it is not the Rice-based compressor (which also uses this name usually), but a utility created by Michael Schindler (www.compressconsult.com). This system seems to offer excellent compression ratios, although it usually needs large amounts of data for this –which excludes a flight-realistic implementation (Pace 1998).

Whenever possible, all of these compression systems have been executed with the "--best" switches on, this is, requesting the best compression (which usually implies lower throughputs). Many of these archiving utilities also offer additional configuration switches, which may be inspected in the future in order to improve the compression, or to simulate flight-realistic conditions (e.g., restarting the dictionaries and trees more often).

Regarding the GASS simulations used in these tests, we have selected basically the same set used in chapter 10, thus testing the systems under low-density, high-density and extreme-density areas of the sky. Obviously, all of these simulations have a limiting magnitude of 20, in order to offer the highest realism in the results. The TM file sizes range from 53MB to 1.45GB, while the quantized *.bin* files are about 2.37 times smaller.

12.2.3. Execution and reliability tests

Once the implementations of GaiaSPaP and GaiaSRaD were ready, the Static Telemetry CODEC Release 1.1 was complete so we proceeded to execute some elementary tests in order to verify their correct operation. The most important test was related to the verification of a correct pre-compression and restoration, thus verifying that this system is, actually, *lossless*. This test was successful: a few TM files from GASS were converted with *txt2bin*, coded with GaiaSPaP, decoded with GaiaSRaD and restored by *bin2txt*. By visual inspection everything was absolutely fine, so the lossless operation of all the Static TM CODEC from end to end was verified. An automatic verifier, including a calculation of the quantization errors (RMS and maximum values), shall be developed in the future.

Another interesting test was related to the throughput of the pre-compression system. Under the testing environment described in the previous section, GaiaSPaP pre-compresses at an average rate of more than 3Mbps (as binary data *read*). Regarding *txt2bin*, owing to the improvements and modifications already noted in the previous section, its throughput has increased from about 1Mbps to more than 2Mbps (as binary data *written*). The typical throughputs of each of the standard compressors will be shown in the next section. It is important to note that these low processing speeds are caused by the *gaia_file* library, in charge of the bit-level accesses to disk; an improved implementation is currently being developed, which shall significantly increase these throughputs (preliminary tests reveal a processing speed which is more than 10 times faster).

12.3. DATA PRE-COMPRESSION AND STATISTICAL TESTS

12.3.1. Simulator results

Figure 12.5 shows the snapshot of the console output after a GaiaSPaP run. Here we already find much statistical information about the TM file analyzed and pre-compressed, including the percentage of TM occupation for every sub-stream. Figure 12.6 summarizes it, illustrating the approximate contributions of every data field to the input and output TM streams.

```
********** GAIA: TM CODEC and Data Compression simulations **********
SPaP: Stream Precompressor and Partitioner (testbed) for GASS TM binary data
Version 1.2.2 (released 19-Jan-2005). (c) 2005 J. Portell (portell@ieec.fcr.es)
Accepted input data: GASS v2.2 processed by txt2bin v1.0.2

[BI] Allocating 7168 KBytes for the histograms...  [ OK ]
[BI] Sub-stream histograms will be saved to data/precomp/hists/
[BI] Output sub-streams will be saved to directory data/precomp/spp/
Partitioning data/bin/d120.2-3m7s_m20.tm.bin...
100%  data processed  [ OK ]

[BI] Input size: 598 MBytes, distributed as following:
[BI] 4.9417% sync, detection and generic data,
[BI] 0.0068% TDI Offsets data,
[BI] 12.6701% ASM Patches data,
[BI] 8.6157% Readout Times data,
[BI] 5.5748% Readout Positions data,
[BI] 31.1528% AF Patches data,
[BI] 13.3506% AF11 Patches data, and
[BI] 23.6876% BBP Patches data.

[BI] Total output size: 356 MBytes (ratio 1.6810), with:
[BI] 19.5183% Reference Stream (0.4256),
[BI] 0.0114% TDI Offsets Stream (1.0000),
[BI] 10.9167% ASM Patches Stream (1.9510),
[BI] 5.1117% Readout Times Stream (2.8333),
[BI] 4.2598% Readout Positions Stream (2.2000),
[BI] 27.6973% AF Patches Stream (1.8908),
[BI] 12.4228% AF11 Patches Stream (1.8066), and
[BI] 20.0621% BBP Patches Stream (1.9848).

1591075 sources in 189 Data Sets.
[BI] Observation time:  3.1500 minutes.
[BI] Processing time:   26.6337 minutes.
        Processing speed ratio: 0.1183 seconds/second.
2.9977 Mbps processed (read) in average.

Saving histograms...\  [ OK ]
Determining the entropy of the main data fields...\  [ OK ]
Entropies in bits of the main data fields (original/coded, and Shannon limits):
   TDI Offset:        9.9557 bits / 9.9557 bits,        1.0045 / 1.0045
   ASM fluxes:        3.9466 bits / 3.3948 bits,        4.0541 / 4.7130
   Readout Times:     14.3333 bits / 2.9011 bits,       1.1861 / 5.8599
   Readout Positions: 10.9229 bits / 1.9575 bits,       1.0071 / 5.6194
   AF01-10 fluxes:    7.9522 bits / 6.3154 bits,        2.0120 / 2.5335
   AF11 fluxes:       5.5750 bits / 5.2600 bits,        2.8700 / 3.0418
   BBP fluxes:        5.3847 bits / 4.1359 bits,        2.9714 / 3.8686
Overall Shannon limit (assuming a ratio of 1.5 for the Reference Stream):
   Without pre-compression -> 2.1127
   After applying SPaP     -> 3.1989
```

Figure 12.5: Snapshot of GaiaSPaP output after pre-compressing and analyzing a TM file

Although this pre-compression system was not designed to compress the data but simply to prepare them for being compressed afterwards, we implemented it in such a way that smaller code sizes are used whenever possible. For example, the software verifies the value of the 16-bit flux values; if they are small enough, they are written as 8-bit –thus reducing the output size but keeping the byte-based symbol sizes. As it gets evident in the simulator results, we can already see an overall compression ratio of about 1.68, and even more interesting partial results as shown in Table 12.1. This table lists the several sub-streams with science TM data (ordered from most to least contribution to the overall TM size), and the approximate averages of the partial pre-compression ratios.

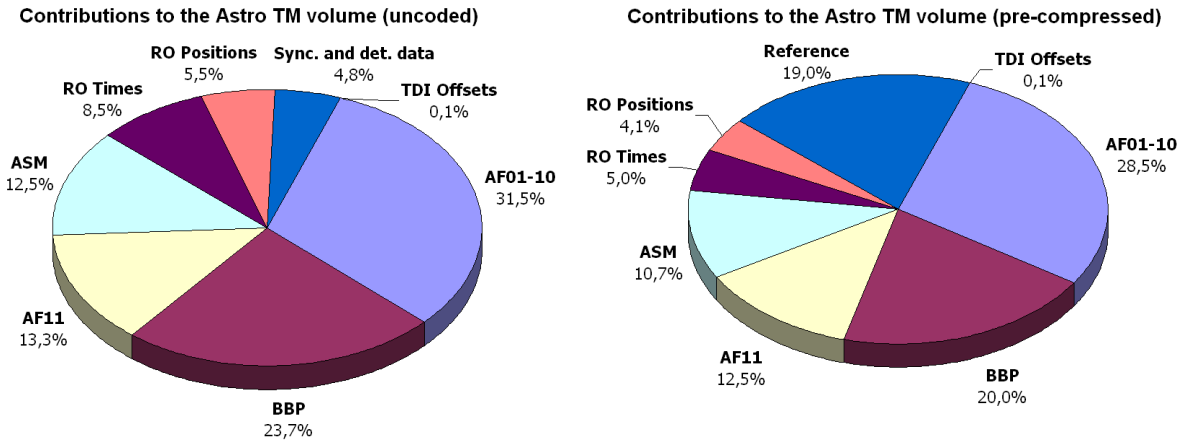


Figure 12.6: Approximate contributions of each data field to the original (left panel) and pre-compressed (right panel) TM budgets

TM sub-stream	Partial pre-compression ratio
AF01-10	1.8
BBP	1.9
AF11	1.7
ASM	1.9
Readout Times	2.8
Readout Positions	2.2
Synchronism and detection data	0.4 (inclusion of flags)
TDI Offset matrices	1.0 (no coding performed)

Table 12.1: Partial pre-compression ratios obtained with GaiaSPaP

12.3.2. Histograms and entropy: the Shannon theoretical limit

Another interesting result offered by this software is the calculation of the entropy for every data field and for the whole TM data file, from which the theoretical compression limit (the Shannon limit) is obtained. As seen in Figure 12.5, the amount of bits required to code each data field are noticeably smaller than the actual number of bits used in the TM model (Masana et al. 2004). This is due to the distribution of observed stellar magnitudes in the Gaia data, since the faint stars (leading to small fluxes, and therefore small code values) are much more abundant than the bright stars. But the interesting issue is that after applying our pre-compression system the amount of required bits per field decreases again. At the end, it implies that the overall Shannon limit (i.e., the theoretical limit for the compression ratio) is much better (i.e., higher) after pre-compressing than before. Tables 12.2 and 12.3 summarize these partial and total improvements. This is a good example of the following fact: the Shannon limit is not a universal and unique measurement of how much a data field (or a set of data) can be compressed, but this limit depends on the way the data is pre-processed before compressing it. In other words, we can see that the expression of the Shannon limit,

$$H = \sum_i P(i) \log_2 \frac{1}{P(i)} \qquad \text{Shannon limit} = \frac{\text{Original field size}}{H}$$

cannot recognize all of the redundancies, as it happens with the standard compression systems. However, if we prepare the data before being compressed, both the Shannon formula and the data compression systems will “recognize” and take advantage of all the redundancies. Summarizing, under the present pre-compression strategy the maximum achievable compression ratio is 3.29, but maybe in the future we can develop an improved pre-compression system which may increase this limit.

Field	Original field size (bits)	*Original*		*Pre-compressed*	
		Entropy (bits)	Shannon limit	Entropy (bits)	Shannon limit
TDI Offsets	10	9.9	1.01	9.9	1.01
ASM samples	16	4.2	3.8	3.4	4.7
RO Times	17	14.3	1.2	2.9	5.8
RO Positions	11	10.9	1.0	2.0	5.5
AF01-10 samples	16	8.0	2.0	6.4	2.5
AF11 samples	16	5.8	2.7	5.6	2.9
BBP samples	16	5.5	2.9	4.2	3.8

Table 12.2: Partial pre-compression ratios obtained with GaiaSPaP

Shannon limit	Minimum	Average	Maximum
Without pre-compression	1.95	2.07	2.14
With SPaP pre-compression	2.88	3.13	3.29

Table 12.3: Theoretical maximums of the compression ratios with and without pre-compression

Figures 12.7 to 12.13 are illustrative examples of this improvement in the partial (and total) compression ratios. As previously introduced, a set of data fields (or a data file) can only be compressed if the several values occur with different probabilities, i.e., if the histogram of the field values (or Probability Density Function, PDF) is not uniform. Moreover, the more extreme probabilities we find, the higher the compression ratio can be. This is, in order to compress a data field, its histogram (or PDF) must be as steep as possible, accumulating most of the probability of occurrence in a few values. In this way, a standard compression system will assign many less bits to the most probable values, whereas the least probable values (or not probable at all) will be coded with more bits. This is the elementary concept of data compression. As shown in the figures, we are steeping all the histograms and, therefore, very much improving the final compression ratio achieved. The figures have been obtained analyzing the histogram files generated by *GaiaSPaP* with *entrops.m*, a MatLab function also included in the *Static TM CODEC* Release 1.1.

We start by examining figure 12.7, which shows the original and the pre-compressed histograms of the TDI Offsets. This case is only included for completeness (both plots coincide), since we did not include any pre-compression for this data field. Nevertheless, it is a good illustration of how "uncompressible" is the TDI Offset matrix, since this histogram is almost uniform (note that the plot is semi-logarithmic). When realistic TDI outputs become eventually available from GASS, this field should be processed in some way. Also, we must note that this field represents a negligible contribution to the total TM volume, but we have included it in the pre-compression system in order to prepare the future inclusion of the Optimized TDC (see chapter 7).

Figure 12.8, on the other hand, is the first clear example of *histogram steeping*. We have plotted the original histogram of the ASM sample values in blue, and the histogram of the pre-compressed values in black. We must recall that all of these plots are semi-logarithmic, in order to better appreciate how most of the probability concentrates in a narrow interval. Furthermore, the data to be compressed is composed of much smaller values, thus making the sub-stream more uniform –avoiding the code differences due to different star magnitudes. These differences will obviously appear as well, but they will get much smaller.

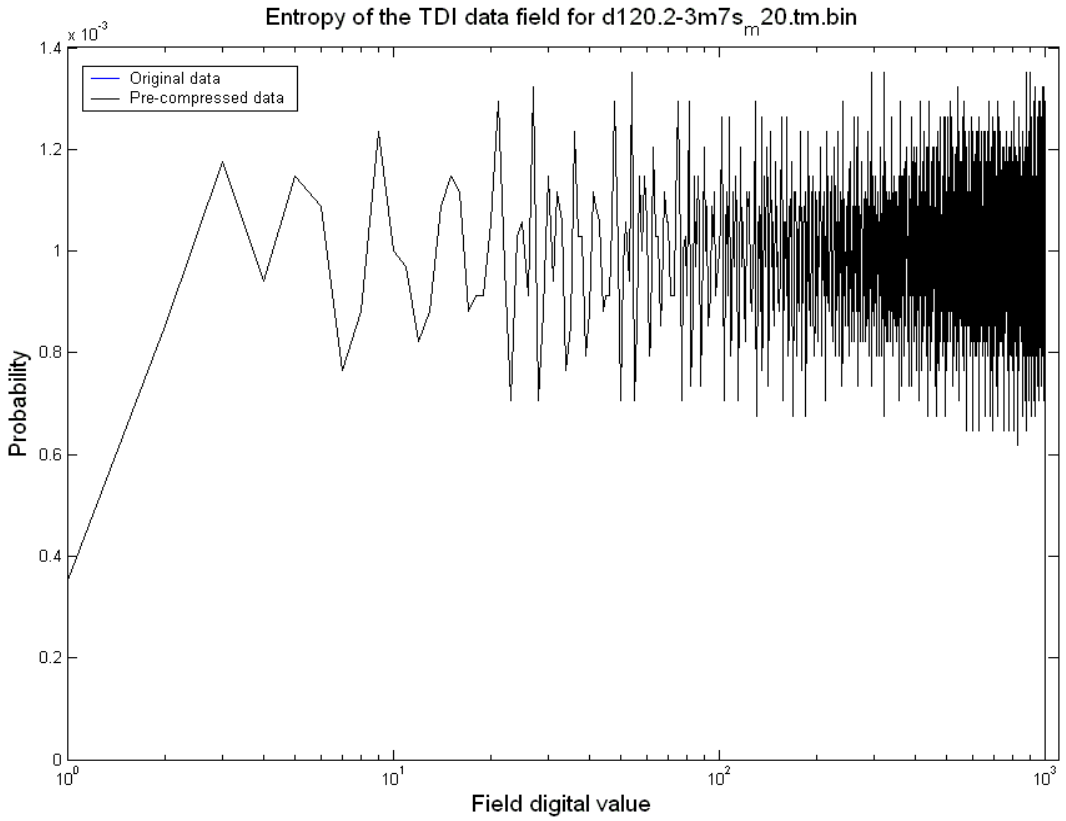


Figure 12.7: Histogram of the TDI Offset field values

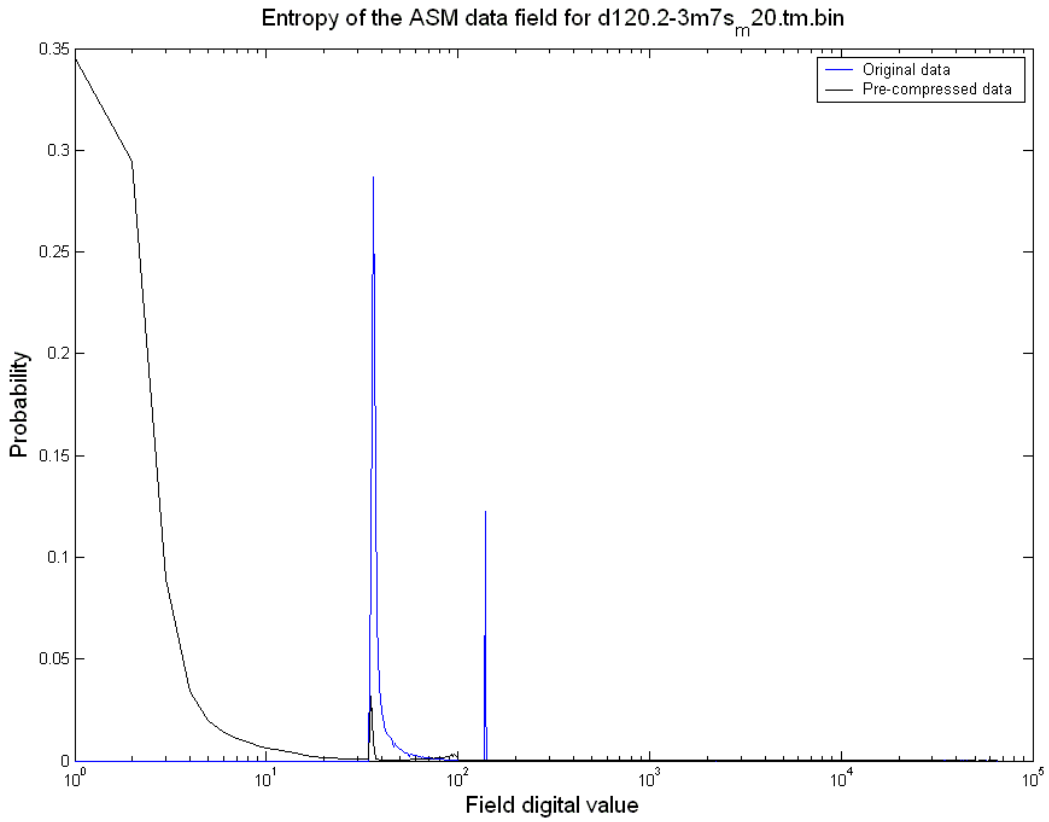


Figure 12.8: Histograms of the ASM sample field values

Figure 12.9 is a stunning demonstration of histogram steeping. While the blue (original) histograms are almost invisible –due to the almost uniform distribution of the values, the black

(pre-compressed) histograms get really steep. This is the reason of the large increase in the Shannon limit for these fields after being pre-compressed, as shown in Table 12.2 before.

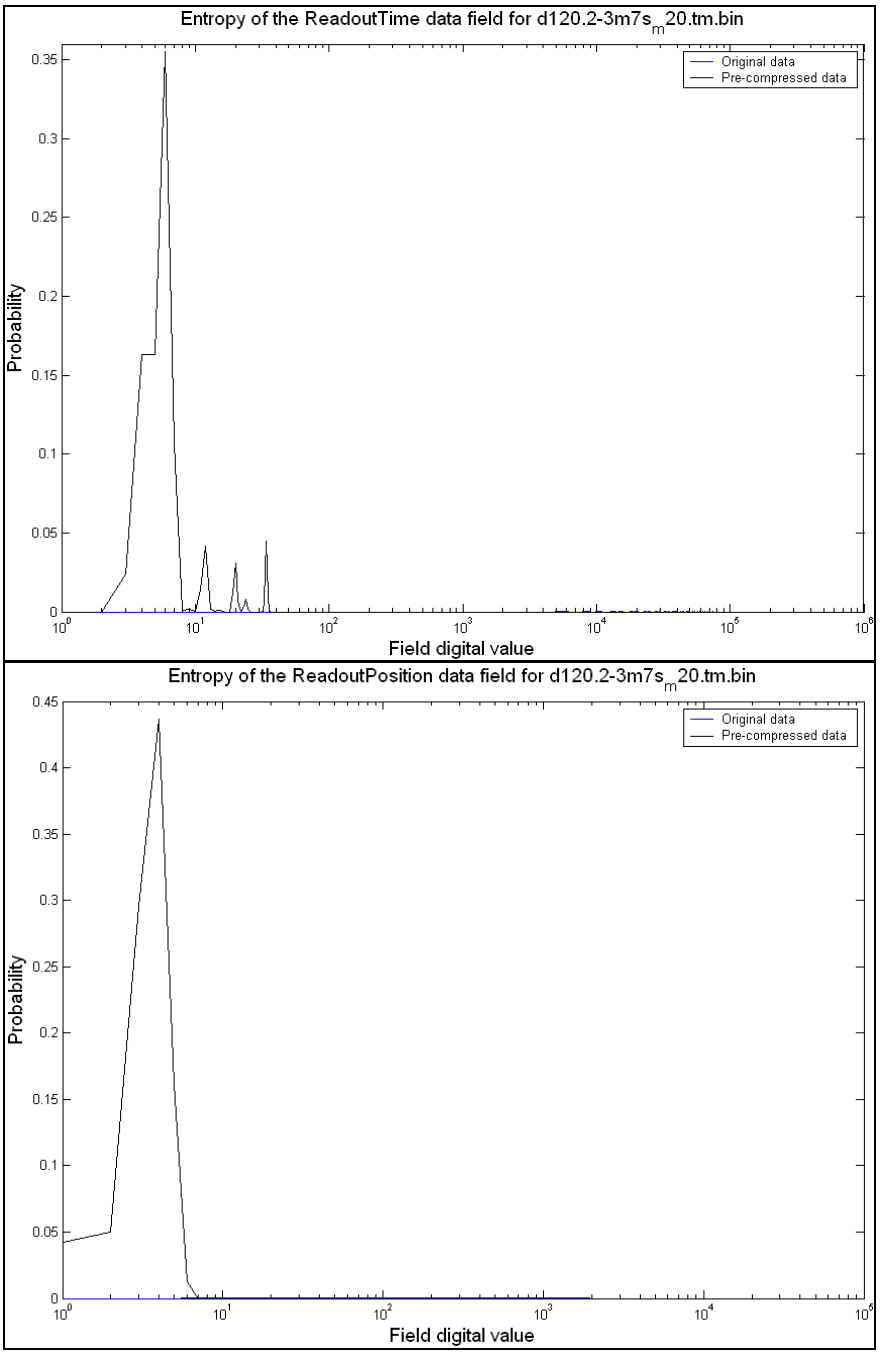


Figure 12.9: Histograms of the readout time (top panel) and position (bottom panel) field values

Now we turn our attention to the most important data fields: AF and BBP samples. First, figure 12.10 shows the histogram of the most contributing data field, corresponding to the AF01-10 samples. The blue histogram recalls us in someway the histogram that we found when testing *txt2bin* in chapter 10, when we analyzed the AF peak values. That histogram depended on the limiting magnitude used in GASS, and if we compare this case with the histogram obtained with *txt2bin* with 20th magnitude simulations they look really similar –as we could expect. This original histogram is relatively steep and with a peak on fairly low values, which leads to the relatively high Shannon limit shown in Table 12.2. But when we pre-compress these data differentially, the probabilities get even steeper and the peak moves to even lower values. All in all leads to better compression ratios achievable (at least theoretically), as Table 12.2 shows.

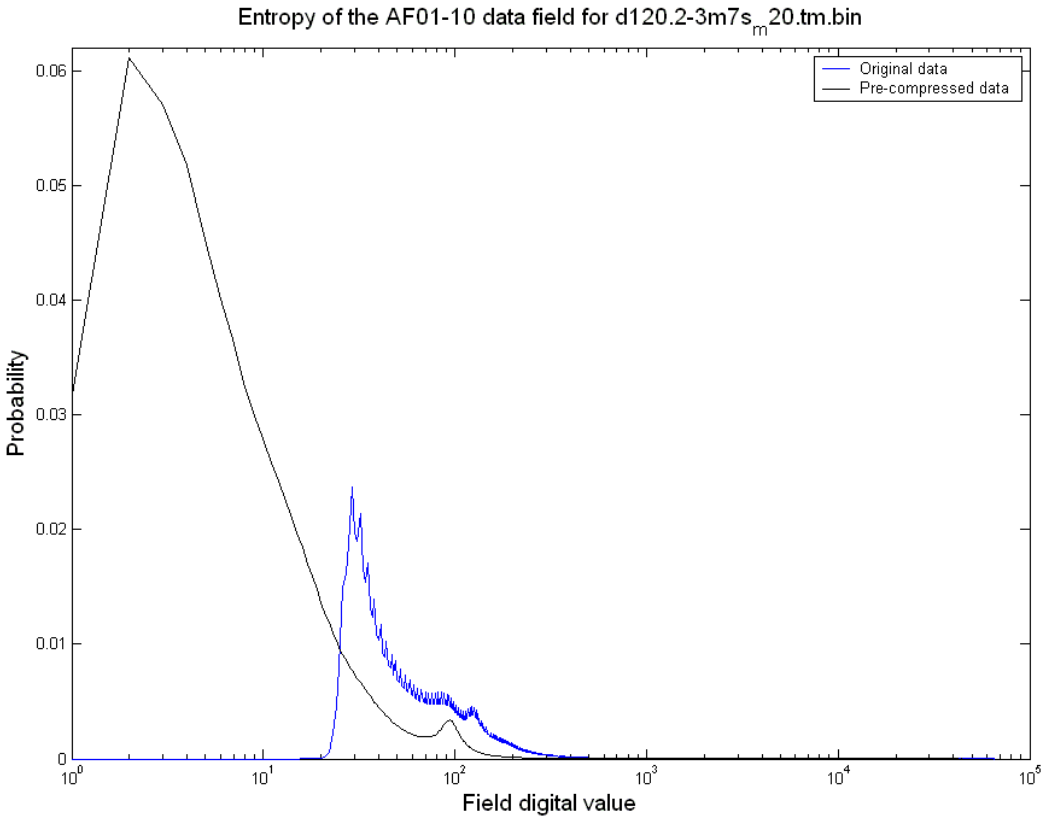


Figure 12.10: Histogram of the AF samples field values

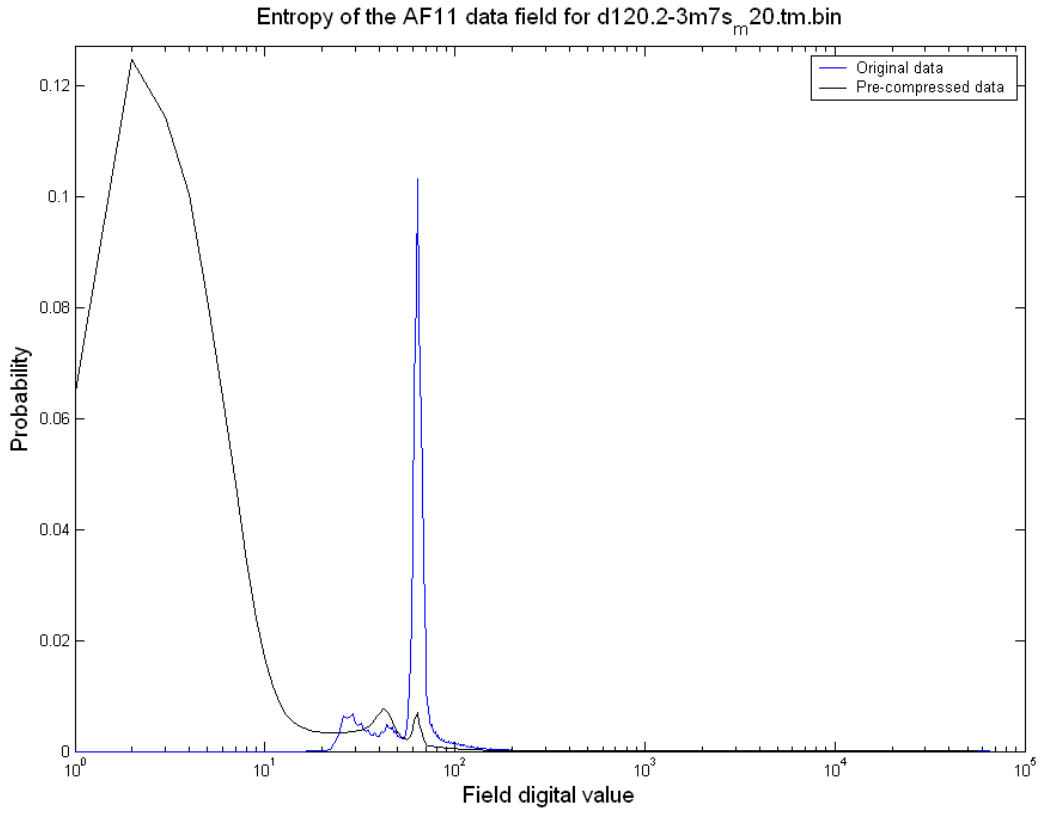


Figure 12.11: Histograms of the AF11 sample values

The analysis of the AF11 samples (figure 12.11) is similar to the "standard" AFs (AF01-10): while the original histogram is already fairly steep, the coded histogram reduces the average code values and narrows the typical range that they fit in. We must note that this result has

been obtained with sample-sample differentiators, while AF01-10 are coded with patch-patch differentiators (except for AF01).

Finally, figure 12.12 shows a similar result for the BBP samples. In this case we see different peaks in the original histogram, surely as a result of the different filters used. These separate peaks noticeably reduce the maximum compression ratio achievable, while a differential pre-compression of the samples leads to almost a single peak in lower values.

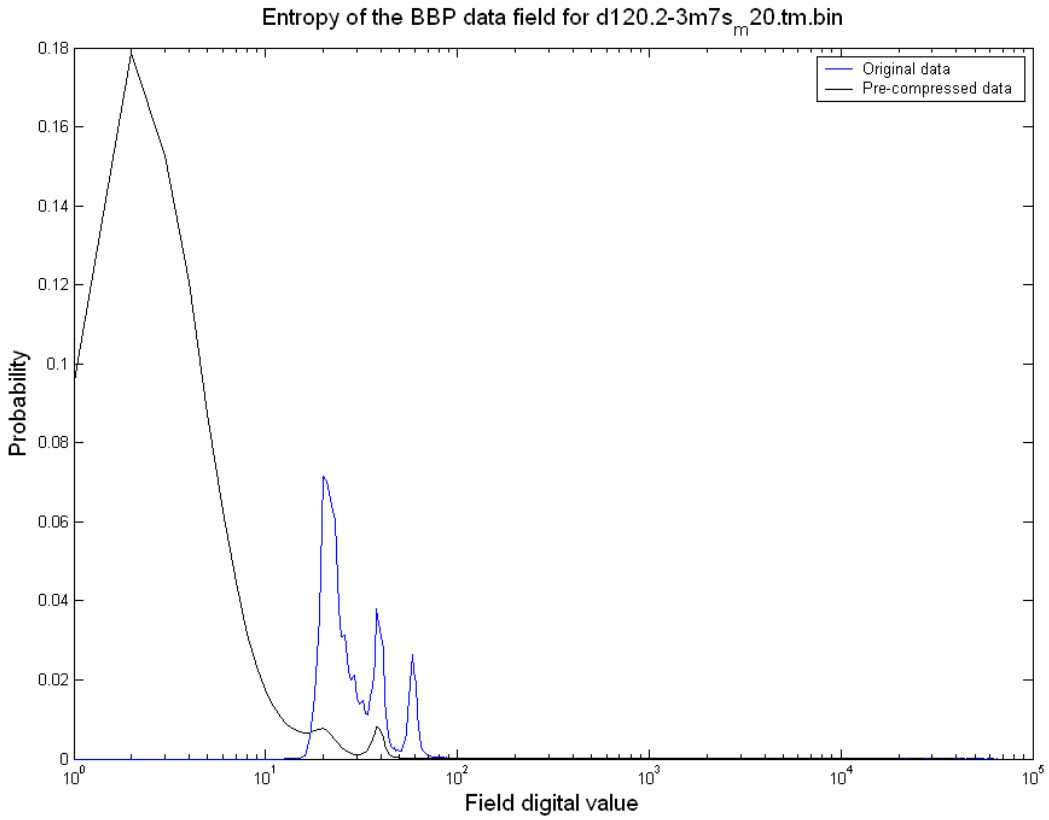


Figure 12.12: Histograms of the BBP sample values

As a last analysis, we wanted to perform a more thorough test on the case where most data are generated: the faint stars. In this case, assuming magnitudes fainter than 16, the AF windows are always 6 samples wide. This particular case (faint stars and 6-samples AF patches) represents the most important contribution to the total TM volume to be transmitted, and therefore we must focus our efforts on this. Figure 12.13 shows an extended histogram set, where we can see a separate histogram for each of the AF samples, before and after the pre-compression process. Here we can see a systematic error in the GASS output (which has been corrected in the last version): the centroid is always on the 3rd sample, instead of being shared between the 3rd and the 4th ones. It implies that the 6th sample is always the faintest one, which also reflects on the pre-compressed values. The reason is simple: faint stars will lead to smaller differences among different patches, while brighter stars will feature larger successive differences. Therefore, fainter or brighter samples of the same star will equally lead to smaller or larger differences. From this, we can obviously conclude that the samples with the peak fluxes will be the most difficult to pre-compress and compress. On the other hand, we can also see in the histograms that those corresponding to the 2nd and 4th pre-compressed samples cannot get as steep as we would like. The reason is another systematic error in GASS: instead of outputting patches with a centroid that smoothly shifts from one sample to the next (due to tracking errors, NEOs, etc.), the patches feature rapid changes between successive measurements. This should be investigated in the future, since its effect is an important decrease in the overall compression ratio.

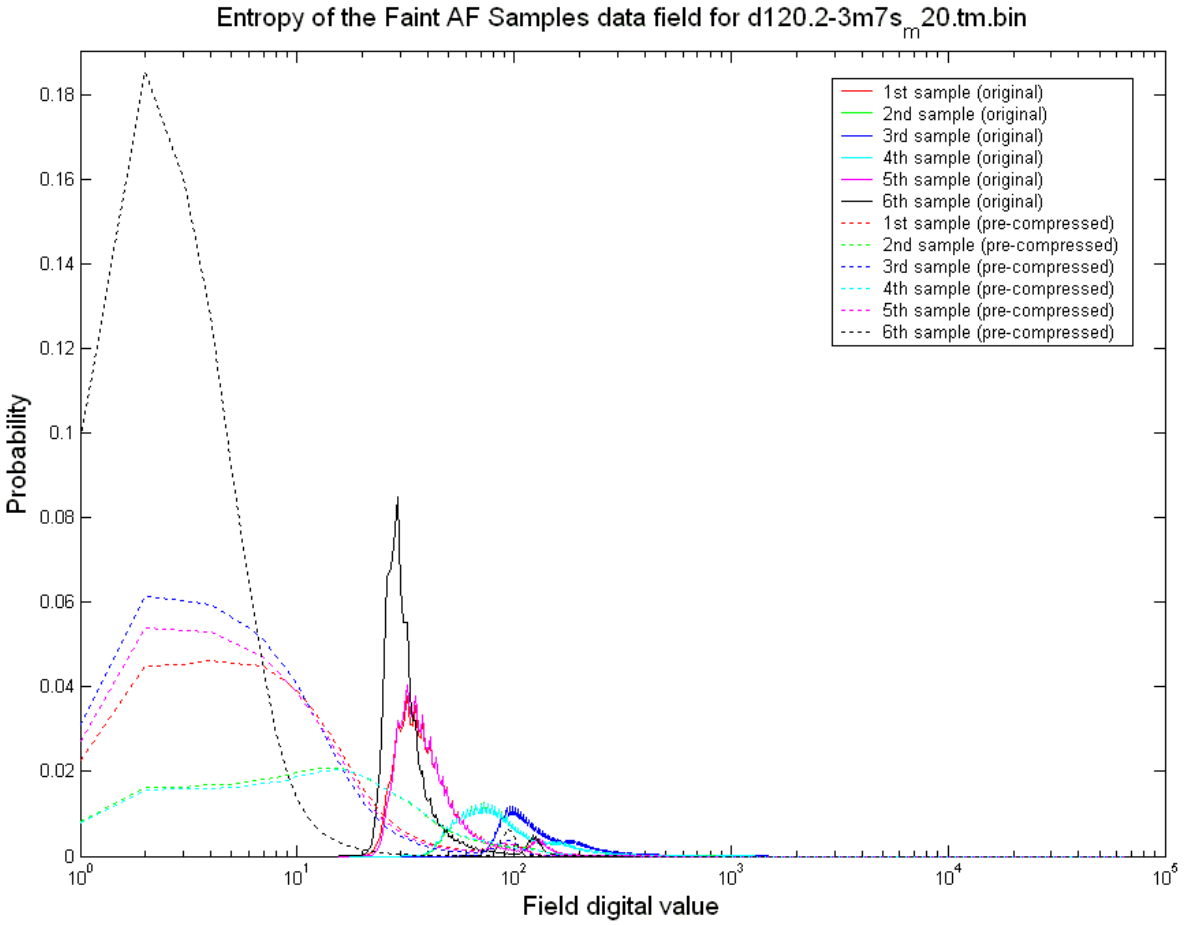


Figure 12.13: Extended histograms of the AF samples from faint stars

12.3.3. The real limits

In the previous section we have explained the theoretical compression ratios that can be achieved, and illustrated with histograms the improvement wrt standard compression systems. Except for the histograms (which only offer a qualitative result of this full data compression system), these previous results are theoretical. When actually implementing such a system we must take into account several considerations, in order to really obtain a *lossless* system. For example, the pre-compression software must add some flags to the data. These flags indicate, e.g., the coding of the data fields (8 or 16 bits), or the resulting sign of the differential coding. The inclusion of these flags obviously decrease the overall compression ratio –as indicated in Table 12.1, where we can see that the reference sub-stream decreases its size a factor of 0.4 (i.e., its size increases a factor of 2.5). On the other hand, we can think of powerful pre-processes to the data leading to large compression ratios, but these will surely increase the amount of required flags. Therefore, we must find a compromise between a powerful enough pre-compression system, and the amount of flags that it must introduce (in order to keep the full system as lossless).

Additionally to these design limits, we must also take into account the implementation limits. This is, we should develop a system with enough simplicity to be implemented in the available hardware resources on-board, which will usually be small. The throughputs found in these tests are encouraging, since we have determined that the bottleneck is the *gaia_file* library, in charge of the bit-level file I/O (and this data access will surely have a much better implementation on-board). Despite of this, the average throughput is about 3Mbps, which is not far from the requirements that we will have on-board. The final throughput should be at least 10 to 20Mbps in order to cope with the sky areas with high densities. The improved *gaia_file* currently being implemented is offering even higher throughputs in the first tests.

12.4. DATA COMPRESSION: RESULTS

After testing the pre-compression simulator and analyzing its results, in this section we show the final results obtained when applying standard compression systems, both to the original and to the pre-compressed data.

12.4.1. Ratios achieved using only standard systems

First of all, we have applied standard data compression systems to TM data (after being quantized and converted to binary by *txt2bin*). Since preliminary tests revealed that the data compression ratio can depend on the star density, we have tested these systems on several simulations. Table 12.4 summarizes the results, where the simulation start (indicated as days since 2010.0) and the simulation length are shown as well. The best results are highlighted in grey. The smaller binary file is about 22.6MB, while the larger one is about 623MB. Table 12.5, on the other hand, shows the average throughput offered by each of these standard systems, indicated as megabits read per second. This table highlights the fastest systems for every simulation.

Stars/s	***Day***	***Length***	*Arj*	*Zip*	*Gzip*	*Ain*	*Lha*	*Rar*	*Bzip2*	*Szip*
207	89.875	2.12 h	1.399	1.403	1.403	1.406	1.421	1.529	1.618	1.678
318	89.75	53.2 m	1.431	1.436	1.436	1.439	1.459	1.570	1.671	1.730
1208	89.0	51.0 s	1.452	1.459	1.459	1.464	1.488	1.596	1.699	1.767
3705	113.05	2.61 m	1.515	1.521	1.521	1.526	1.556	1.680	1.854	1.865
8418	120.2	3.15 m	1.490	1.499	1.499	1.504	1.535	1.640	1.852	1.892
14784	113.056	8.0 s	1.499	1.507	1.507	1.512	1.544	1.659	1.876	1.923
19991	113.0565	3.0 s	1.495	1.503	1.504	1.509	1.540	1.660	1.868	1.922

Table 12.4: Compression ratios obtained using only standard systems

Stars/s	*Arj*	*Zip*	*Gzip*	*Ain*	*Lha*	*Rar*	*Bzip2*	*Szip*
207	21.524	18.158	17.379	20.919	12.477	1.953	14.590	10.174
318	20.360	15.955	15.681	20.462	11.708	1.922	14.626	10.235
1208	21.032	16.334	16.526	20.489	12.445	2.199	15.114	10.677
3705	19.044	14.858	14.723	19.584	10.943	1.865	14.449	10.360
8418	20.382	15.991	15.759	19.663	11.992	2.049	15.102	10.698
14784	19.669	15.048	15.067	18.434	11.339	1.981	14.224	10.083
19991	19.844	15.089	15.175	18.273	11.637	2.034	14.092	10.020

Table 12.5: Throughputs (in Mbps) offered by standard data compressors

These results are illustrated in figures 12.14 and 12.15 for a better understanding. As can be seen there, in the best of the cases, the compression ratio achievable using *only* a standard system is smaller than 2. The best system (in terms of compression ratio) seems to be *Szip*, which can offer a ratio of up to 1.92 (and a worst case of about 1.6). These results are completely unsatisfactory, since the requirement for Gaia is a ratio of 3 or more. Therefore, with these tests we verify that the use of a standard compression system is not enough.

Regarding the throughput in the data compression process, the fastest systems are *Arj* and *Ain*, which can compress at a rate of about 20Mbps. *Rar* is the slowest system, compressing at a rate of only 2Mbps, although its compression ratio is noticeably better. When combining the compression ratio and the processing rate, *BZip2* appears as the best solution, since it offers the second best ratio (and not far from *SZip*) while the rate is much better (more than 14Mbps). *Rar* should be absolutely discarded, since its compression ratio is smaller than *BZip2* while its rate is stunningly slow. *Ain*, finally, appears as another interesting solution if the hardware resources are really limited, since it compresses better than Arj, Zip or GZip, while its execution is faster.

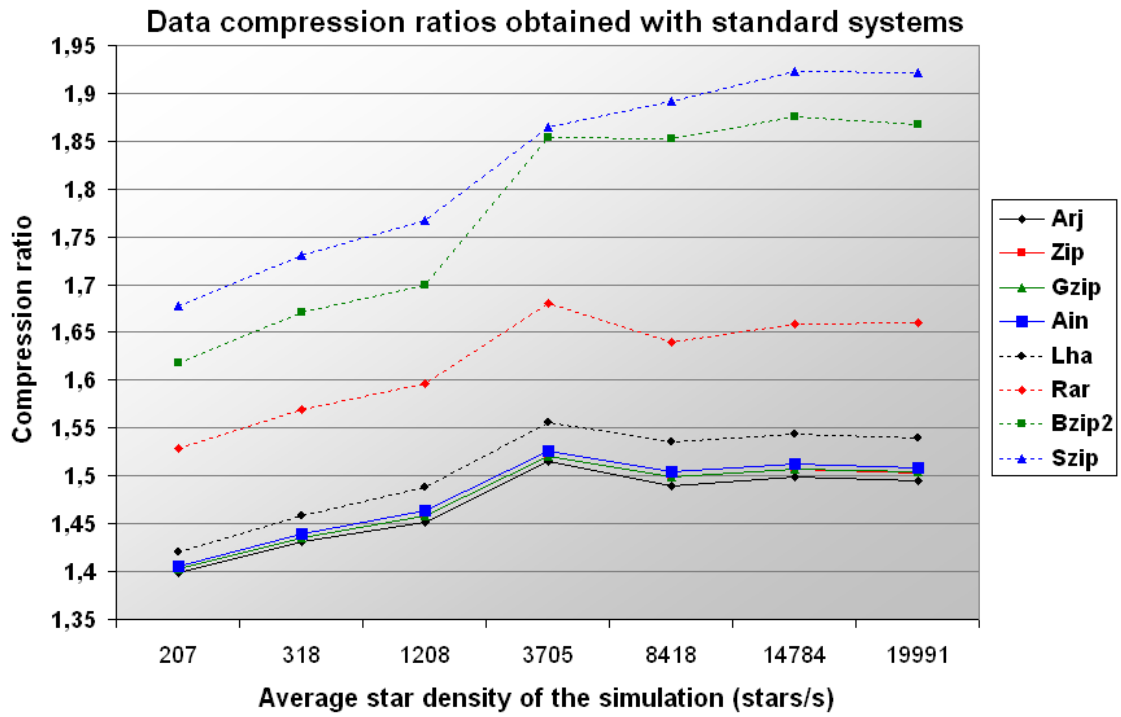


Figure 12.14: Data compression ratios obtained using only standard systems

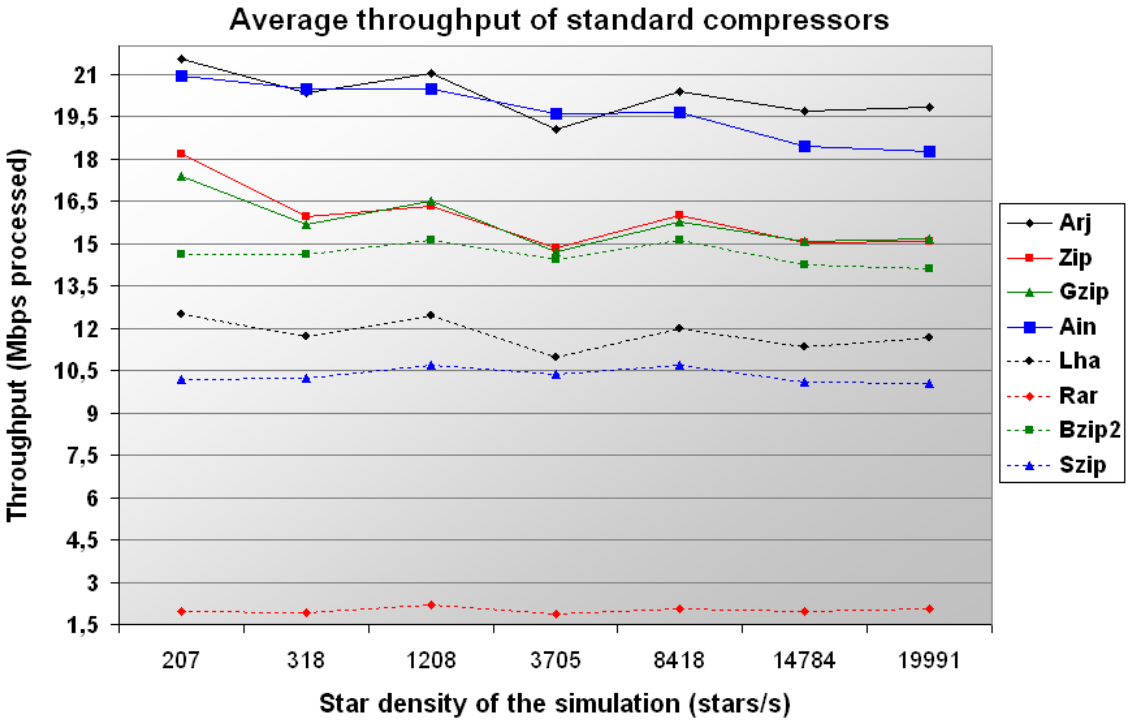


Figure 12.15: Average throughputs of standard data compression systems

Table 12.6 summarizes the most relevant cases of these data compression simulations using only standard systems. We can see how the Shannon limit shown in section 12.3.2 is not reached at all (as expected), although we have obtained up to 90% of the best limit (which was 2.14). This is, indeed, a good result, demonstrating that SZip is an excellent solution for data compression. From table 12.6 we can also see that the best compromise solution (highlighted in grey) is BZip2, taking into account both the compression ratio and the processing speed. The latter is satisfactory, but the compression ratio ranges between 1.62 and 1.87, which is absolutely insufficient for our needs. These ratios would lead to final TM rates of about 400kbps in the best of the cases (i.e., with low star densities), taking into account

Astro1+Astro2. This rate is acceptable, but when entering high star densities it increases to about 6Mbps, and to 32Mbps in peak star densities. Therefore, it is obviously imperative to increase the compression ratio.

Star density	Best ratio			Ratio / speed compromise			Best throughput		
	Ratio	System	Speed (Mbps)	Ratio	System	Speed (Mbps)	Ratio	System	Speed (Mbps)
Low (~200 stars/s)	**1.68**	SZip	10.2	**1.62**	BZip2	14.6	**1.40**	Arj	21.5
High (~3700 stars/s)	**1.86**		10.3	**1.85**		14.4	**1.53**	Ain	19.5
Peak (~20000 stars/s)	**1.92**		10.0	**1.87**		14.1	**1.49**	Arj	19.8

Table 12.6: Summary of compression ratios and throughputs using only standard systems

12.4.2. Ratios achieved applying standard systems after SPaP

Now we enter the final phase of our study: to execute the standard compressors after having applied our pre-compression system, *GaiaSPaP*, on the GASS simulated TM data.

12.4.2.1. Minimum hardware requirements: one single compressor

Having in mind the limited hardware capabilities available on-board, we first tested the simplest approach to our system, implying minimum hardware requirements. This solution is to compress all of the sub-streams with a single compressor, sequentially. A valid simulation for this is to join the 8 sub-streams with *tar* (the standard UNIX/Linux command), and to test its compression with each of the standard systems that we have selected. The counterpart of this is that the compressor will receive different statistics in a single file (or stream), and therefore will surely offer a lower compression ratio, since it will have to continuously adapt its rules to each segment of data. Table 12.7 shows the results, where we can verify that SZip is still the best compression system (in terms of ratio achieved), while BZip2 keeps as the 2nd best solution with a similar compression ratio but faster processing speed (about 36% faster). From this, we conclude that SZip and BZip2 are excellent compressors for general purpose, offering the best results either with non-emphasized data (not pre-compressed) or with pre-compressed data (but mixed each other).

Nevertheless, the most important result shown in this table is the final compression ratio, which has impressively increased wrt that obtained only with standard systems. It is important to note that this ratio is relative to the original binary (raw) size *before* the pre-compression process; this is, the final ratio is the combination of the pre-compression ratio and the compression ratio itself. The latter, when calculated wrt the pre-compressed size, keeps around 1.77 to 1.86 (more or less as before), but when combined with a pre-compression ratio of 1.56 to 1.69 the results are much better: the requirement of 3 gets fulfilled in most of the cases.

Stars/s	***Day***	***Length***	*Best ratio*			*2nd best ratio*		
			Ratio	*System*	*Throughput*	*Ratio*	*System*	*Throughput*
207	89.875	2.12 h	**2.775**	SZip	10.32 Mbps	2.690	BZip2	14.23 Mbps
318	89.75	53.2 m	**2.896**	SZip	10.29 Mbps	2.809	BZip2	14.20 Mbps
1208	89.0	51.0 s	**2.971**	SZip	11.29 Mbps	2.899	BZip2	15.45 Mbps
3705	113.05	2.61 m	**3.137**	SZip	10.45 Mbps	3.052	BZip2	14.13 Mbps
8418	120.2	3.15 m	**3.113**	SZip	10.26 Mbps	3.019	BZip2	14.02 Mbps
14784	113.056	8.0 s	**3.104**	SZip	10.75 Mbps	3.022	BZip2	14.44 Mbps
19991	113.0565	3.0 s	**3.080**	SZip	10.86 Mbps	3.008	BZip2	14.89 Mbps

Table 12.7: Best compression ratios and throughputs obtained with standard systems applied after SPaP

We must clarify at this point the reason why we are able to "compress two times". It is well known that the successive application of a standard compressor two times to the same data does not compress twice: the first compression process offers the best results, while the second one is usually negligible. This is true because a compression system outputs data with an extremely uniform statistic and, therefore, is almost uncompressible. Therefore, when applying a compressor to already compressed data, the resulting ratios are extremely small. On the other hand, our pre-compression system offers an output with extreme probabilities, and therefore any compression system will recognize them and compress the data with high ratios.

12.4.2.2. Best compression ratio: one compressor per sub-stream

The results shown in the previous section are satisfactory, mostly fulfilling the compression ratio requirement of 3. Despite of this, we have tested an even better compression algorithm, processing each of the sub-streams with all the standard compression systems. The most relevant results are shown in Table 12.8, where the tests with the 7 simulations available have been summarized. The compression system offering the best (and 2nd best) results for each sub-stream is also indicated, as well as the percentage of cases where it becomes the best system. We can see that this compression algorithm offers really impressive partial ratios –such as for the readout coordinates or the ASM patches. We must remind that these compression ratios have been calculated wrt the original (raw binary) TM data (before SPaP), and that the lossless operation of the full system has been verified. As we can see, the overall compression ratios are slightly better, as one could expect. The requirement of 3 is now fulfilled in all of the cases except for very low star densities, where the TM sizes are actually not a problem.

The case of the TDI Offsets may look a little strange, since it ranges from no compression to a ratio of almost 4. The reason is that this is the data field with the smallest contribution, and in some simulations the resulting sub-stream file was really small –and hence, the implicit overheads of the data compression systems even expanded a little the file. The reference sub-stream is always expanded a ratio of about 1.6, due to the flags and signs added by the pre-compression system. Finally, SZip and BZip2 systems are confirmed here to be the best solution not only for mixed data (the reference sub-stream), but also for the flux data (except for the BBP sub-stream). Regarding the rest of sub-streams, some of them get best compressed with different systems depending on the conditions. Nevertheless, the differences between the best systems are not large enough to recommend an adaptive implementation, this is, the use of one or another system depending on the conditions. Finally, we can see that *Rar* is the best compressor for sub-streams such as the BBP or the TDI Offsets, which implies the use of an extremely slow system. Table 12.9 shows the weighted throughputs of the standard compression systems that appear in Table 12.8, when taking into account the contribution of their associated sub-streams to the overall TM volume (and assuming independent compressors running in parallel). From this, we determine that the bottleneck in such a compression algorithm is *Rar* yet, but now being able to process 10 Mbps in average –taking into account that these data is already pre-compressed and, therefore, implies lower data rates than the original TM data. In other words, this compression method would be able to process about 19 Mbps in average (taking into account the average pre-compression ratio of BBP). This should be enough for our purposes, and therefore we can consider this scheme as realistic.

The use of the best compressor for each sub-stream guarantees the maximum ratios, and hence it will always be the best solution (at least using this SPaP method). The counterpart of this compression algorithm is the need of 8 compression processes (or ASICs) running simultaneously and in parallel, with the adequate synchronization between them, plus the SPaP process itself. The improvement wrt the simpler compression algorithm (SPaP plus 1 compressor processing the sub-streams sequentially) is 1% to 2%. Therefore, at least with these GASS simulated data, it is not really worth it to use different compression systems running in parallel. Improved GASS versions may output TM data with even more realism, and when compressed with these systems the difference could increase. It is therefore imperative to keep testing these data compression schemes with more realistic data obtained from GASS or even GIBIS.

	Best case			2nd best case		
Sub-stream:	Ratios	Best system	Cases	Ratios	2nd best sys.	Cases
Reference	0.62 – 0.65	SZip	100%	0.60 – 0.64	BZip2	100%
TDI Offsets	0.92 – 3.79	Rar/Gzip	43% each	0.92 – 2.20	LHa	57%
ASM samples	9.00 – 15.38	BZip2	100%	8.38 – 12.76	SZip	100%
RO Times	22.59 – 39.41	SZip	57%	20.06 – 36.50	Rar	57%
RO Positions	28.87 – 37.95	SZip	71%	27.77 – 36.61	Rar	57%
AF01-10 samples	2.21 – 2.53	SZip	100%	2.15 – 2.48	BZip2	100%
AF11 samples	2.89 – 3.14	SZip	100%	2.75 – 3.01	BZip2	100%
BBP samples	3.44 – 3.99	Rar	100%	3.40 – 3.90	SZip	100%
OVERALL	**2.801 – 3.173**	-	-	**2.714 – 3.072**	-	-

Table 12.8: Summary of compression ratios obtained with several standard systems applied to the sub-streams generated by GaiaSPaP

System:	**Associated sub-stream**	**Real throughput (worst case, approx.)**	**Weighted throughput**
SZip	Reference	10 Mbps	52 Mbps
	ASM samples		93 Mbps
	RO Times		200 Mbps
	RO Positions		244 Mbps
	AF01-10 samples		35 Mbps
	AF11 samples		80 Mbps
	BBP samples		50 Mbps
BZip2	Reference	14 Mbps	74 Mbps
	ASM samples		131 Mbps
	AF01-10 samples		49 Mbps
	AF11 samples		112 Mbps
Rar	TDI Offsets	2 Mbps	2000 Mbps
	RO Times		40 Mbps
	RO Positions		49 Mbps
	BBP samples		10 Mbps
GZip	TDI Offsets	15 Mbps	15000 Mbps
LHa	TDI Offsets	11 Mbps	11000 Mbps

Table 12.9: Weighted throughputs of the best compression systems for the SPaP scheme

12.4.2.3. Summary of results

Figure 12.16 shows the compression ratios achieved when using different methods:

- Optimal case with more resource consumption, using the best compressor for each sub-stream.
- Conservative case with minimal resource consumption, using a fast compressor for the combined sequence of sub-streams.
- Standard case, using the best standard compressor for the raw binary data.

From this figure we can see that even the conservative case (combined sub-streams compressed with BZip2) fulfills the requirements of a compression ratio of 3 in most of the cases. The standard solution (without SPaP) must be absolutely discarded, while the optimal solution (SPaP plus independent compressors) could be considered if its implementation on-board is permissible within the hardware processing limits.

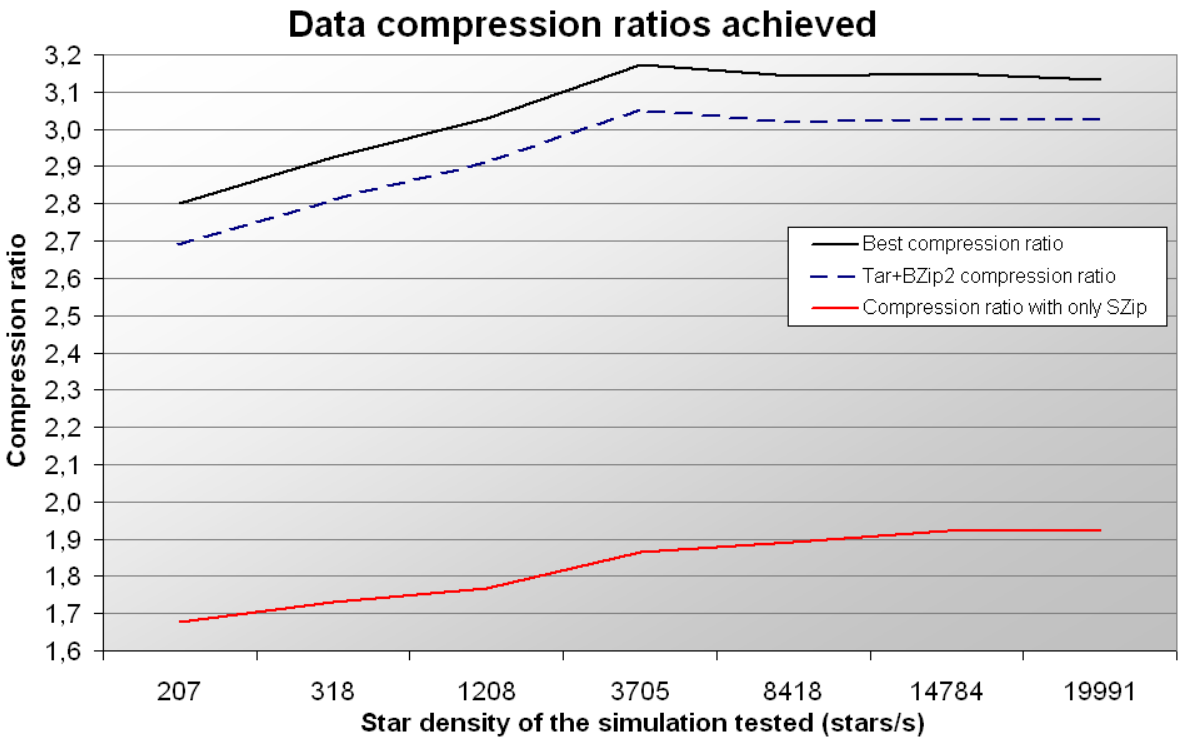


Figure 12.16: Data compression ratios achieved with standard and optimized compression algorithms

12.5. CONCLUSIONS

12.5.1. Final results

After several tests performed with the new *GaiaSPaP* software and 8 standard compression systems over GASS simulated TM data, we have determined two main solutions to the Gaia data compression problem:

1. Conservative solution:
   - Pre-compress and partition the telemetry data with *GaiaSPaP*
   - Combine the resulting sub-streams into a single stream. This could be done, for example, for every Data Set (DS) or for every minute of data, thus containing a stream of data containing 1 DS or 1 minute of measurements.
   - Compress this combined stream with BZip2, the best standard compressor offering excellent ratios with a fast execution speed.
2. Optimal ratio solution:
   - Pre-compress and partition the telemetry data with *GaiaSPaP*
   - Compress each of the resulting sub-streams with the adequate standard system, independently and in parallel:
     - Reference sub-stream with SZip
     - TDI Offsets with Rar
     - ASM data with BZip2
     - AF/BBP window readout times with SZip
     - AF/BBP window readout positions with SZip

- AF01 to AF10 data with SZip
- AF11 data with SZip
- BBP data with Rar

The difference between these two solutions, in terms of compression ratio, is about 4% (1% to 2% if SZip is used instead of BZip2 in the conservative solution). The ratios obtained with the first solution range from 2.69 (in low density areas) to 3.05, compressing more than 3 in areas denser than 3500 stars/s approximately –equivalent to about 285000 stars per square degree. On the other hand, the second (optimal) solution offers ratios from 2.80 to 3.17, compressing more than 3 in areas denser than 1000 stars/s approximately –i.e., about 81000 stars per square degree. The resulting TM data rates obtained with both solutions and with the standard solution are shown in figure 12.17. The use of SPaP before the application of a standard data compressor, therefore, appears as the best solution for the moment; not only for the high compression ratios achieved, but also because it relaxes the throughput requirements of the standard compressor (since this pre-compressor already reduces the TM throughput in a ratio of more than 1.6).

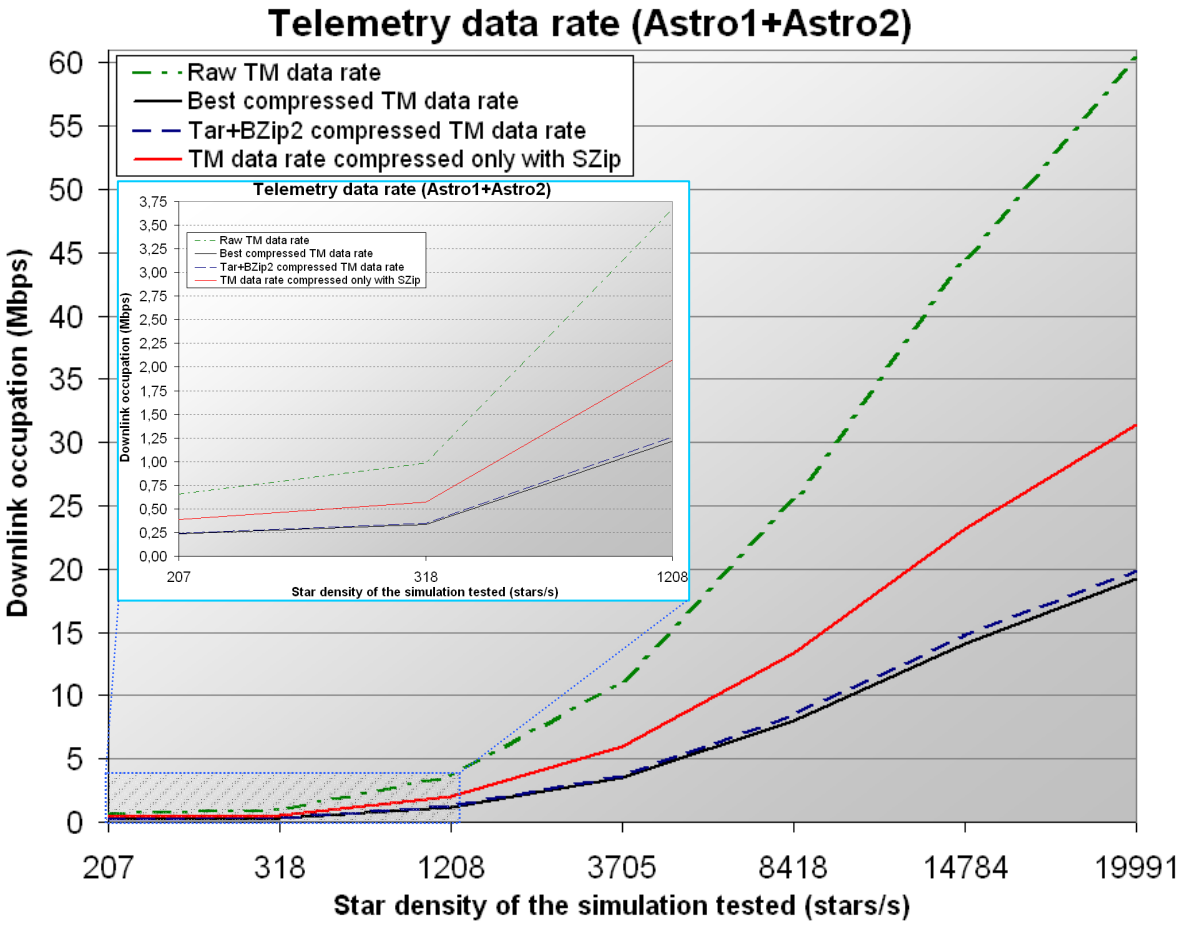


Figure 12.17: Average TM data rates of Astro, obtained with different compression methods

When analyzing the several tables shown throughout this chapter we can see that the most problematic issue is the compression of the data fields that imply the most TM occupation, which are the AF01-11 sub-streams (AF01-10 plus AF11). These data suppose 41% of the overall telemetry volume (after pre-compression), while their typical compression ratios cannot surpass 2.52 (AF01-10) or 3.14 (AF11) in the best case. The AF01-10 patches are specially problematic, so forthcoming studies on data compression should focus specially on these data. The ASM and BBP, also representing an important amount of telemetry, can be well compressed –reaching ratios of almost 4 (BBP) or a spectacular 15 (ASM), so they do not pose a problem.

12.5.2. Flight-realistic considerations

A good data compression method must also take into account several flight-realistic considerations in order to avoid potential problems when receiving and decoding the data on ground. First of all, it must be robust in front of "strange data behaviors", this is, the system must not highly depend on the models of the data. For example, the pre-compression method should not expect an ideal PSF (or LSF) profile, because if it is not fulfilled (e.g., due to cosmic rays, a NEO or a bad star tracking along the focal plane), the compression ratio would dramatically decrease. In this sense, the SPaP method seems robust enough because it simply expects a (more or less) uniform set of measurements along the focal plane. Even more, the GASS simulations used in these tests usually showed strange (and rapid) patch changes along the AF, and despite of this we have obtained such high compression ratios.

On the other hand, a very important issue is the reset of the "compression rules" for each of the systems to be used. Usually, standard data compressors dynamically adapt to the input data, and use some *tree* or *dictionary* to code and compress the data. This same tree or dictionary must be exactly reproduced on reception (i.e., on ground), in order to correctly decode the data, but none of these trees or dictionaries is literally transmitted (in order to avoid TM consumption): they are deduced from the data that is being received. This is, the compression rules are dynamically determined both on-board and on ground with the same algorithm. The problem appears when there is some transmission error (due to the noisy communications channel) and the data received on ground is incomplete. Then, the receiver will use different decoding rules than those used on-board, and therefore all the data received from that moment will be unavoidably lost, at least until the next reset in the coding rules. This issue was first raised in Geijo (2002), although Pace (1998) already indicated the problem of transmitting too large data structures. Therefore, whichever compression system is used, the reset of the compression rules must be done from time to time. It contradicts with the fact that most of the data compression systems improve their ratios when compressing large amounts of data without resetting these rules, since the statistics estimates internally done by these systems get more and more accurate.

A compromise solution must be found for this problem. As a first approximation, we can recommend a reset in the compression rules every 10 seconds in average density areas, or every minute in low density areas, for example. This would lead to sub-stream sizes large enough to be well compressed, but not so much to represent an unacceptable loss if a transmission error occurs. A better solution for this would be the initialization of the coding rules to a predefined state, instead of the zero-state standard initialization. This is, the compression rules of each compressor could be initialized to the predicted statistics (i.e., the histograms shown in section 12.3.2). In this way, we could reset the rules more often, avoiding transmission problems but keeping high compression ratios at the same time. The counterpart is that this would require a high control over the compression systems, the feasibility of which should be studied in the future.

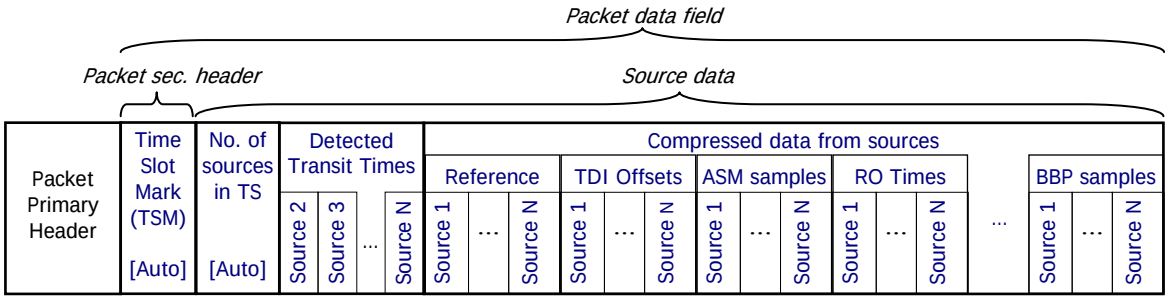


Figure 12.18: Possible inclusion of the compression method within the Packet Telemetry standard

Finally, the data compression method must take into account the TM standards to be used, such as the ESA Packet TM standard. It implies that the recommendation for Gaia data compression must also consider the insertion of the compressed data into the Source Packets of the TM

system. The ideal solution would be to keep the structures defined in chapter 7, for example in the way illustrated in figure 12.18, which at the same time makes possible the sequential inclusion of the sub-streams generated by SPaP and compressed afterwards.

12.5.3. Possible improvements

The data compression algorithm proposed in this chapter, based on SPaP, is a first attempt of a valid solution for the problem of the large TM volumes generated by Astro. It can obviously be improved, not only in terms of the compression ratio achieved but also in terms of simplicity and efficiency. Forthcoming studies will deal with these issues in the near future, but as a first set of recommendations we list here some concepts that should be taken into account:

- As previously noted, the AF01-10 patches are the most consuming data and, at the same time, the most difficult fields to be compressed. It is therefore imperative to focus on this issue.
- Generally, for any of the sub-streams, the histograms should get even steeper. It means that better predictors should be developed –or that predictors should be used where they are not, as in the AF01-10 sub-stream where only a differentiator is used.
- The AF01-10 sub-stream could be partitioned again, e.g., generating one sub-stream per sample. This could be done, for example, only for faint stars (where the patch is composed of 6 samples), in order to avoid problems with the multi-window sampling scheme. As shown in figure 12.13, this could improve even more the histogram steeping in some cases, helping increasing the overall compression ratio.
- The faint stars (e.g., fainter than 16^{th} magnitude) represent the highest percentage of data to compress. Therefore we must also focus on this.
- PSF-based predictors could be used, although it would be risky in the sense that any disturbance in the measurement would sensibly decrease the compression ratio. On the other hand, if such a method would be able to offer a ratio of, e.g., 2.5 to 3 in the worst of the cases, it could be acceptable.
- Tailored data compressors could be developed. In Portell (2000), Adaptive Huffman and LZW with selectable symbol sizes (from 3 to 14 bits per symbol) were used, offering excellent results. In our case, similar compressor could also be developed (substituting the standard compressors studied in this chapter), using the adequate symbol size in every sub-stream. Also, as indicated in section 12.5.2, a predefined reset of the compression rules should be implemented. This would significantly reduce the "learning time" of every compressor, offering very high compression ratios from the very beginning.
- Correlation and auto-correlation analyses of the data fields could help in determining better solutions, besides of the histograms (or Probability Density Functions, PDFs) shown in this chapter.
- In the worst cases (i.e., under extremely high star densities), some controlled losses could be inserted in the system if necessary. For example, the saturation level could be decreased to the 50% or to the 25% of the original value, thus reducing 1 or 2 bits per sample without affecting the faintest stars. Also, a logarithmic-like codification could be used, using larger quantization steps for brighter flux values and vice versa –as in Portell (2000) for the energy fields.
- Finally, *GaiaSPaP* (and the TM CODEC in general) could be improved in order to offer an already flight-realistic output, fulfilling the Packet TM standard and taking into account the resets of the compression rules. Also, the data analyses should be optional, thus making possible more realistic estimates on the system throughput.

Conclusions and Forthcoming Work

Conclusions

Many issues have been addressed in this work, from general definitions of the mission to detailed algorithms for coding and compressing the science data generated by the satellite instruments. Most of these issues are original contributions of this work, while others are simply compilations of existing definitions –or even third-party descriptions included in the document for the sake of completeness and clarity. For example, chapter 3 is simply a compilation of already fixed parameters and characteristics of the mission, although their compilation into a handy guide has resulted very useful to several teams. Chapter 2, on the other hand, describing a set of reference systems and conventions, represents the start of a large work directly requested by the mission scientist; although this work is being superseded by newer definitions and recommendations (Bastian 2004), it constituted a non-trivial and necessary starting point which has been really useful for the mission. Finally, chapter 4 not only has compiled some operation issues of the Astro instrument, but it also has introduced new and extremely useful proposals such as the use of Field Density Indexes (FDI), which are currently being widely used by the scientific community and, furthermore, suppose very useful tools for the internal management of the payload data handling system. Other analyses on the operation of the Astro instrument, such as the problem of the crowded fields and the readout capability of the focal plane, have substantially helped in the refinement of the instrumentation guidelines.

The PDHS has been studied in detail in the second part of our work, which started from very preliminary existing designs described by ESA (2000) and refined them in such a way that the mission project team has appreciated our work. Although the final implementation of the PDHS will be determined by the prime contractor of the project (which will be Astrium or Alcatel/Alenia Spazio), our work has been used as an extremely detailed refinement in the scientific requirements (and as a guideline for the implementation of the PDHS). In this way, thanks to our work, the error margin achieved in the final implementation (due to possible misunderstandings between science and project teams) will be much smaller than usual. It is worth noting at this point that most of the ideas introduced in our work have been included in the designs of the competing industrial consortia.

Regarding the telemetry and communications of the mission, chapter 6 of this thesis shows a compilation of data fields and the explanation of their basic “transmission” scheme in GASS simulations. We must note that we have actively participated in the definition of this telemetry model and, specially, in its timing scheme. The latter was one of our highest interests, since we already found in previous studies (Portell 2000) that a good timing scheme can save a lot of telemetry volume. Therefore, the case of Gaia has been thoroughly studied in chapter 7, not only identifying the most critical timing requirements (which was an open design issue) but also defining new timing, codification and transmission schemes. Not only these schemes are perfectly integrated in the ESA packet telemetry standard, but they take advantage of its structure and offer a completely optimized operation, continuously adapting to the observation conditions. With the simple inclusion of our system, more than 1 kbps can be saved in average, reaching up to 20 kbps or more in crowded fields.

This improvement in the conditions of the science telemetry gets reinforced thanks to the work we have introduced in chapter 8, where longer contact times between Gaia and the ground station can easily be achieved by simply decreasing the minimum elevation angle. This new proposal, currently being studied by the science and project teams of the mission, can make use of additional correcting codes (Geijo 2002) if the margin in the channel quality is not enough. If this option proposed by ourselves is finally included in Gaia, the overall downlink capacity will be increased by almost 10%.

Despite of the availability of powerful mission simulators like GASS or GIBIS, their output data cannot be directly used for telemetry, coding and –specially– data compression simulations. In order to verify not only our designs (such as those of chapter 7) but also any

other proposal from third-parties, adequate binary files with simulated telemetry data were required. One of the most interesting (and promising) proposals included in our work is the design and implementation of a *Telemetry CODEC*, a new, complete and flexible software tool for Gaia simulations. In its current version it analyses and converts simulated telemetry data from GASS format to a more flight-realistic format, ready to be processed and evaluated by data compression simulators. We have not only conceived this tool and implemented its first (static) version, but we also have defined the basis of a *Dynamic Telemetry CODEC*, a completely flexible tool, configured with XML files and capable of performing a large set of complex operations on almost any type of simulated data.

The last part of the thesis has addressed the problem of the data compression in Gaia, which poses a challenge for the information theory and data compression systems. In chapter 11 we have analyzed the current coding scheme of Astro data, studied a third-party proposal, and finally offered new proposals that should significantly increase the compression ratio. These preliminary proposals, although successfully tested, were based on an old design of Gaia and, furthermore, they had to be improved in order to achieve even better results. This much improved data compression system has been described in the last chapter, where detailed statistical analyses of Gaia data have also been introduced. Differential coders, combined with the adequate data predictors, are the basis of our optimized data pre-compression system; after this, and adequately partitioning the data in sub-streams, standard data compressors can offer much higher ratios. Thanks to our work, final ratios of 2.7 can easily be obtained in the worst of the cases, while the best standard compressor (and in the best of the cases) cannot surpass 1.9. Our results get even better in high density areas (where the telemetry problem gets more important), reaching compression ratios of up to 3.2 (and spectacular partial ratios of 15.4, in the ASM field). It is worth noting at this point that these compression ratios obtained with our design are the highest ones ever achieved with realistic Gaia telemetry data. Our system is the only one that fulfills the compression requirement of 3 for the Astro instrument –although recent studies recommend slightly higher ratios of 3.5 to 4 in order to guarantee the transmission of all the science data. Forthcoming studies may improve even more our design and such high compression ratios could be achieved in the future.

Forthcoming Work

With the work presented in this thesis we have offered to ESA the design of a complete payload data handling, compression and transmission system. Its definition, reliability and quality levels have been qualified as excellent by scientists and engineers of the project –which is really satisfactory taking into account that this is a space project and, therefore, its constraints are usually very stringent. Nevertheless, being such a large-scale project (and having suffered so many changes in its design), it is obvious that many aspects of our work may need an update or improvement. First of all, the following items may need an update to the latest design of Gaia:

- Definition of the reference systems and conventions. We have based our work on the proposals described in the second chapter of this document, in order to avoid a continuous update of all our designs –which was not worth the required effort and adaptation time. Actually, as recommended by the project scientists, all of the proposals and designs related to Gaia should take Bastian (2000) as the main reference (which, actually, is based on our proposal). Therefore, all of our designs (and their nomenclature) should be adapted to this latest definition of reference systems and conventions.
- Telemetry model, which should not only be updated to a completely flight-realistic design (which is partly done in chapter 7) but should also include the MBP and RVS instruments. This effort should be done in cooperation with the GDAAS team at the University of Barcelona, the photometric working group and the RVS consortium.
- Timing scheme and time data codification. Some work has already been started by telemetry experts of the mission, in order to use a more flight-realistic scheme. These same experts have already requested our work in order to design the adequate Source Packets for these new proposals (in order to fulfill the ESA packet telemetry standard).
- The overall design of the payload data handling system also needs an update, although this issue is competence of the prime contractor of the mission.

An extremely useful simulation tool, the Telemetry CODEC, has been introduced by our work. Its static implementation, including a data compression module, has been developed with excellent results. In order to improve this tool and to offer more flexibility, the following items should be considered:

- The dynamic implementation of the TM CODEC should be carried on, continuing the work already started (which has led to the creation of some common libraries and software data models).
- XML files for the latest designs of GASS (including MBP), GIBIS and the RVS Simulator, including the adequate data processing definitions for each of them.
- To improve the data compression system in order to reach ratios of 3.5 or more, also including additional options such as controlled losses (for extremely high density areas). All of this should be included in the dynamic TM CODEC, and adequately defined in the XML files.
- To define XML files needed for the creation of packet telemetry Trasfer Frames and, if possible, simulate a realistic communications channel in order to evaluate the effect of transmission errors.

Finally, the option for decreasing the minimum elevation angle for the ground station to contact Gaia should be further studied, as well as the inclusion of additional correcting codes within the source packets. We must note that, obviously, as in any other space mission, Gaia has a huge number of issues to consider. Our intention here is far from trying to list all of its open issues, but only to take into account the fields that we have started with our work and that would be interesting to continue in cooperation with other members of the several Gaia teams.

Acronyms

The following table summarizes the acronyms used in this thesis, linking them to the adequate applicability field.

Acronym	Meaning	Applicability
A/D	Analog-to-Digital Converter	Engineering / PDHE
AC	Across-Scan	Reference Systems
ACK	Acknowledgement	Telemetry / Data communications
ACMS	Attitude Control Module System	Spacecraft
ADC	Analog-to-Digital Converter	Engineering / PDHE
ADCM	Attitude Data Collection Module	PDHS
ADE	Adaptive Differential Encoding	Data compression
ADR	Average Data Rate	PDHS / telemetry
ADU	Analog-to-Digital Unit	PDHE / signal processing
AF	Astrometric Field	Payload / focal planes
AL	Along-scan	Reference Systems
AOCS	Attitude and Orbital Control Subsystem	Spacecraft
AP	Application Process	Telemetry
API	Application Process Identifier	Telemetry
ASM	Astrometric Sky Mapper	Payload / focal planes
ASR	Average Star Rate	PDHS
AU	Astronomical Unit	Science
BAMD	Basic Angle Monitoring Device	Payload
BBP	Broad Band Photometer	Payload / focal planes
BDE	Basic Differential Encoding	Data compression
BG	Background	Miscellaneous / telemetry
BIPM	Bureau International des Poids et Mesures	Organizations
bps	Bits per second	Data communications
Bps	Bytes per second	Data communications
CCD	Charge Coupled Device	Engineering / PDHE
CCDRS	CCD Reference System	Reference systems
CCSDS	Consultative Committee for Space Data Systems	Organizations
CDM	Clock Distribution Module	PDHS
CESCA	Centre de Supercomputació de Catalunya	Organizations
CO	Complete Observation	PDHS / science
CODEC	Coder/Decoder	Data processing
CoM	Center of Masses	Payload / Spacecraft
CoMRS	Center of Masses Reference System	Reference systems
CPU	Central Processing Unit	Data processing
DA	Detection Algorithm	PDHS
DBT	Dynamic Barycentric Time (TDB is used eventually)	Reference systems
DEMUX	Demultiplexer	Engineering / PDHE
DFT	Discrete Fourier Transform	Data processing

DH&C	Data handling & Compression	PDHS
DIV	Integer division	Miscellaneous / data processing
DR	Dynamic Range	Data processing
DS	Data Set	Telemetry
DSP	Digital Signal Processor	Data processing
DTD	Document Type Definition	Simulations / file types
DTT	Detected Transit Time	Data processing / telemetry
ECI	Earth-Centered Inertial	Reference systems
ESA	European Space Agency	Organizations
ESOC	European Space Operations Center	Organizations
FADE	Fully Adaptive Differential Encoding	Data compression
FAE	Fully Adaptive Encoding	Data compression
FAMbE	Fully Adaptive Model-based Encoding	Data compression
FDI	Field Density Index	PDHS
FFT	Fast Fourier Transform	Data processing
FIR	Finite Impulse Response (filter)	Data processing / signal proc.
FK5	Fifth Fundamental Catalogue	Reference systems
FOV	Field Of View	Payload / optics
FOVRS	Field-Of-View Reference System	Reference systems
FP	Focal Plane	Engineering / instrumentation
FPA	Focal Plane Assembly	Engineering / PDHE
FPRS	Focal Plane Reference System	Reference systems
GASS	Gaia System Simulator	Simulations / models
GC	Great Circle	Mission / science
GcRS	Geocentric Reference System	Reference systems
GD	Gaia Detection (source detection in the sky mappers)	PDHS
GDAAS	Gaia Data Access and Analysis Study	Simulations / models
GIBIS	Gaia Imaging and Basic Instrument Simulator	Simulations / models
GIS	Global Iterative Solution	Data reduction / data processing
GMV	GMV S.A. (Grupo de Mecánica del Vuelo)	Organizations
GP	Galactic Plane	Mission / science
GPS	Global Positioning System	Miscellaneous
GS	Ground Station	Data communications
GSR	Gaia Study Report (Red Book)	Mission
GST	Gaia Science Team	Mission
GUI	Graphical User Interface	Simulations / interfaces
HiPr	High Priority	Miscellaneous
HK	Housekeeping	Spacecraft / data handling
HKDCM	Housekeeping Data Collection Module	PDHS
HRTF	High Rate Telemetry Formatter	PDHS / spacecraft
HTM	Hierarchical Triangular Mesh	Simulations
I/O	Input/Output	Miscellaneous / data processing
IAU	International Astronomical Union	Organizations / standards
ICRF	International Celestial Reference Frame	Reference systems
ICRS	International Celestial Reference System	Reference systems
IDFI	Integrated FDI	PDHS

IDFT	Inverse Discrete Fourier Transform	Data processing
IEEC	Institut d'Estudis Espacials de Catalunya	Organizations
IEEE	Institute of Electrical and Electronic Engineers	Organizations
IERS	International Earth Rotation Service	Reference systems
IFDI	Integrated Field Density Index	PDHS
IIR	Infinite Impulse Response (filter)	Data processing / signal proc.
ITMC	Integrated Telemetry Curve	Simulations / telemetry
ITU	International Telecommunications Union	Organizations
JPL	Jet Propulsion Laboratory	Organizations
LoPr	Low Priority	Miscellaneous
LSB	Least Significant Bit	Data processing/communications
LSF	Line Spread Function	Instrumentation / optics
MAE	Modified "Adaptive" Encoding	Data compression
MbE	Model-based Encoding	Data compression
MBP	Medium Band Photometer	Payload / focal planes
Mbps	Megabit per second	Data communications
MBps	Megabyte per second	Data communications
MC	Master Clock	PDHS
MOD	Modulus of integer division	Miscellaneous / data processing
μs	Microsecond	Miscellaneous
Ms	Millisecond	Miscellaneous
MSB	Most Significant Bit	Data processing / Data comm.
MTO	Maximum TSM Offset	PDHS / Telemetry
MTS	Maximum TS Sources	PDHS / Telemetry
MUX	Multiplexer	Engineering / PDHE
N/A	Not Applicable	Miscellaneous
NACK	Negative Acknowledgement	Data communications
NEO	Near Earth Object	Science
ns	Nanosecond	Miscellaneous
NSL	Nominal Scanning Law	Mission
OB	On-Board	Miscellaneous
OBD	On-Board Detection Working Group	Organizations / mission
OBS	On-Board Storage	PDHS
PDF	Probability Density Function	Simulations / Data analysis
PDHE	Payload Data Handling Electronics	Engineering / payload
PDHS	Payload Data Handling System	Engineering / payload
PDHU	Payload Data Handling Unit	Engineering / payload
PFL	Probability of Frame Loss	Telemetry / Data communications
PLL	Phase Locked Loop	Engineering / payload
PLM	Payload Module	Spacecraft
PO	Partial Observation	PDHS / science
PPL	Probability of Packet Loss	Telemetry / data communications
PSD	Power Spectral Density	Simulations / signal processing
PSF	Point Spread Function	Instrumentation / optics
QE	Quantum Efficiency	Instrumentation / CCDs
Rb	Rubidium	PDHS

RF :	Reference Frame	Reference systems
	Radio Frequency	Engineering / Data commun.
ROFT	Read-Out Finish Time	Reference Systems / PDHE
RON	Read-Out Noise	Engineering / CCDs
RS	Reference System	Reference systems
R-S	Reed-Solomon	Telemetry / Data communications
RVS	Radial Velocity Spectrometer	Payload / Focal planes
SA	Selection Algorithm	PDHS
SAA	Sun Aspect Angle	Mission
SCOS	Spacecraft Control and Operations System	Telemetry / Standards
SDCM	Source Data Collection Module	PDHS
SDPM	Source Data Processing Modules	PDHS
SDS	Science Data Selection	PDHS
SGA	Stellar and Galactic Astronomy group	Organizations
SIXE	Spanish-Italian X-ray Experiment	Other missions
SoM	Space of Measurements	Reference systems
SP	Source Packet	Telemetry
SPaP	Stream Pre-compression and Partitioning	Telemetry / data compression
SPV	Supervisor (module)	PDHS
SRaD	Stream Restoration and Decoding	Telemetry / data compression
SRS	Satellite Reference System	Reference systems
SSMM	Solid State Mass Memory	PDHS
ST	Satellite Time	Reference systems
stdin	Standard Input	Simulations / computers
stdout	Standard Output	Simulations / computers
SVM	Service Module	Spacecraft
SWA	Sliding Window Algorithm	PDHS / star detection
SWG	Simulations of data stream Working Group	Mission
TB	Barycentric Time	Reference systems
TBC	To Be Confirmed	Miscellaneous
TBD	To Be Determined	Miscellaneous
TC	Telecommand	Telemetry / data handling
TCB	Barycentric Coordinate Time	Reference systems
TCG	Geocentric Coordinate Time	Reference systems
TCP/IP	Transfer Control Protocol / Internet Protocol	Data communications
TDB	Dynamic Barycentric Time (DBT)	Reference systems
TDC	Time Data Codification	Telemetry/PDHS
TDI	Time Delayed Integration	Engineering / focal plane
TDT	Terrestrial Dynamical Time	Reference systems
TF	Transfer Frame	Telemetry
TM	Telemetry	Telemetry / data handling
TMC	Telemetry Curve	Telemetry / simulations
TMcodec	Telemetry CODEC	Telemetry / simulations
TN	Technical Note	Miscellaneous
TS	Time Slot	PDHS / Telemetry
TSM	Time Slot Mark	PDHS / Telemetry

TT	Terrestrial Time	Reference systems
TT&C	Telemetry, Tracking and Telecommand	Data communications
UB	Universitat de Barcelona	Organizations
UPC	Universitat Politècnica de Catalunya	Organizations
VC	Virtual Channel	Telemetry
VLBI	Very Long Baseline Interferometry	Miscellaneous
VPU	Video Processing Unit	PDHE / PDHS
WAM	Window Acquisition Manager	PDHS
WFE	Wave Front Error	Instrumentation / optics
WFST	Wide Field Star Tracker	Spacecraft
wrt	With Respect To	Miscellaneous
XML	eXtensible Markup Language	Simulations / standards

Index of Figures

Index of Tables

References

BIBLIOGRAPHY

1995A&A...303..604A, The extragalactic reference system of the International Earth Rotation Service, ICRS

Ansari, S. et al., 2004, Applying Grid Technology to Gaia Data Processing, Gaia Symposium

Arenou, F., Lim., J.-C., 2003, Simulation of the on-board detection, OBD-FAJCL-01, draft V 1.6

Bastian, U., 2004, Reference systems, conventions and notations for Gaia, GAIA-ARI-BAS-003, v4.0

CCSDS (Consultative Committee for Space Data Systems), 2000, Packet telemetry, CCSDS 102.0-B-5 blue book

CCSDS (Consultative Committee for Space Data Systems), 2002, Time code formats, CCSDS 301.0-B-3 blue book

Chéreau, F., 2002, Gaia_Detect description, OBD-FC-01, rev. 1.3, http://stellarium.free.fr/cv/OBD-FC-01.ps

De Bruijne, J., 2003a, A proposal for a Gaia parameter database, GAIA-JdB-007

De Bruijne, J., 2003b, Gaia clock accuracy and stability, GAIA-JdB-004

De Felice, F., M.G. Lattanzi, A. Vecchiato, P.L.Bernacca, 1997, Relativistic data reduction for the GAIA satellite (I. A non-perturbative approach in the Schwarzschild framework)

EADS/Astrium, 2002, Gaia System Level Technical Reassessment Study (Final Report), EF5/FR7PC/038.02

ESA, 1988, Packet Telemetry Standard, PSS-04-106, Issue 1

ESA, 1994, Packet Utilisation Standard, PSS-07-101

ESA, 1997, The Hipparcos and Tycho Catalogues, ESA SP-1200

ESA, 2000, Gaia concept and technology study report, ESA-SCI(2000)4

Figueras, F., Masana, E., Luri, X., Torra, J., Jordi, C., 2001, GIS implementation in GDAAS: Preliminary aspects, UB-GDAAS-TN-015, v1.0

Geijo, E.M., 2002, Modelo de canal y sistema de control de errores para SIXE (in Spanish), Master Thesis, UPC

Geijo, E.M., Portell, J., García-Berro, E., Luri, X., Lammers, U., 2004, Improved channel coding for longer contact times with Gaia, GAIA-BCN-008, v1.1

Høg, E., 2000, Gaia Technical Report, SAG-CUO-77

Høg, E., 2002a, Sampling in Astro of bright stars and double stars, GAIA-CUO-100.3, http://obswww.unige.ch/~eyer/VSWG/docs/A100.ps

Høg, E., 2002b, Sampling for all magnitudes: Scheme C, GAIA-CUO-113

Høg, E., Arenou, F., Mignot, S., Babusiaux, C., Katz, D., Jordi, C., 2003, Scientific requirements for the on-board processing, GAIA-CUO-117, http://gaia.am.ub.es/PWG/documents/GAIA_CUO_117.pdf

Johnston, K.J., McCarthy, D.D., Luzum, B.J., Kaplan, G.H., Eds., Towards models and constants for sub-microarcsecond astrometry, IAU Colloquium 180, USNO publication

Jordi, C., Babusiaux, C., Katz, D., Portell, J., Arenou, F., 2003, Gaia: Instrument and satellite parameters, GAIA-SWG-012

Klioner, S.A., 2001, Practical relativistic model of microarcsecond astrometry in space

Kovalevsky, J. et al., 1989, Reference Frames in Astronomy and Geophysics (now obsolete, Johnston et al. is recommended instead), Astrophysics and Space Science Library

Lammers, U., 2003, Proposal for Gaia Science Telemetry Definition and Simulation Work, GAIA-UL-002, v1.0

Lammers, U., 2004, Gaia telemetry rate simulations: A first look at the complete picture, GAIA-UL-008, v0.1

Lindegren, L., 2000, Attitude parameterization for Gaia, SAG-LL-30

Lindegren, L., 2001, Proposed prototype processes for the Gaia Global Iterative Solution, GAIA-LL-34

Masana, E., Luri, X., 2000, Gaia Simulator

Masana, E., Fabricius, C., Luri, X., Torra, J., Figueras, F., Jordi, C., García-Berro, E., Portell, J., 2004, GDAAS Telemetry Model, UB-GDAAS2-TN-003, v2.1

McCarthy, D.D., 1996, IERS Conventions, IERS Technical Note 21, Central Bureau of IERS - Observatoire de Paris

O´Mullane, W., 2000, Gaia Data processing and storage concept

Pace, O., 1998, Clarification on the Probability of Frame Loss (PFL) in data transmission to ground, SAG-OP-001

Perryman, M.A.C., de Boer, K.S., Gilmore, G., Høg, E., Lattanzi, M.G., Lindegren, L., Luri, X., Mignard, F., Pace, O., de Zeeuw, P.T., 2001, GAIA: composition, formation and evolution of the Galaxy, Astronomy & Astrophysics, 369, 339

Portell, J., 2000, Diseño del enlace de datos para SIXE (Spanish Italian X-ray Experiment) (in Spanish), Master Thesis, UPC

Portell, J., 2001a, Thesis proposal for Jordi Portell, JPM-PDT

Portell, J., García-Berro, E., Luri, X., Torra, J., Figueras, F., Jordi, C., Masana, E., Arenou, F., Babusiaux, C., 2001a, Astrometric Instrument Model for Gaia, UB-GDAAS-TN-002, v2.0

Portell, J., García-Berro, E., Luri, X., 2001b, Flux data codification: Proposals of simple codification schemes, GAIA-BCN-001, v2.0, http://gaia.am.ub.es/Portell/GAIA-BCN-001.pdf

Portell, J., 2001c, Some ideas for GIBIS and beyond, GAIA-BCN-003, http://gaia.am.ub.es/Portell/GAIA-BCN-003.pdf

Portell, J., García-Berro, E., Luri, X., Jordi, C., 2002a, Proposal of reference systems for Gaia, GAIA-BCN-002, v2.0, http://gaia.am.ub.es/Portell/GAIA-BCN-002.pdf

Portell, J. et al., 2002b, Payload data handling, compression and telemetry formatting, Gaia Simulation Working Group kick-off meeting (Cambridge, UK), http://gaia.am.ub.es/Portell/PDHU-Compression-Printable.pdf

Portell, J., Masana, E., García-Berro, E., Luri, X., 2002c, Proposal of telemetry formatting for Gaia science data, GAIA-BCN-005, v0.2, http://gaia.am.ub.es/Portell/GAIA-BCN-005.pdf

Portell, J., García-Berro, E., Luri, X., Jordi, C., Figueras, F., Masana, E., Torra, J., Arenou, F., Babusiaux, C., 2003a, Technical characteristics and design issues for the Gaia astrometric focal plane, UB-GDAAS-TN-001, v3.0

Portell, J., García Berro, E., Luri, X., 2003b, Proposal of Payload Data Handling unit and internal Data Flux, GAIA-BCN-004, v2.1

Portell, J., García-Berro, E., Luri, X., 2004a, Timing and transmission schemes for Gaia, technical note GAIA-BCN-006

Portell, J., Luri, X., García-Berro, E.,2004b, Telemetry data formats in simulations, GAIA-BCN-007, v1.5

Portell, J., Luri, X., García-Berro, E., 2004c, Definition of a Telemetry CODEC, GAIA-BCN-011, v1.2

Portell, J., García-Berro, E., Luri, X., 2005a, Telemetry and statistical tests with GASS data and TM CODEC, GAIA-BCN-012, v1.2

Portell, J., García-Berro, E., Luri, X., 2005b, Realistic tests of the data compression system using GASS data, GAIA-BCN-013, v1.2

Vannier, M., 2000, Representation of on-board telemetry, SAG-MV-05

INTERNET REFERENCES

[IR.1.] http://aa.usno.navy.mil/data/docs/ICRS_links.html, The International Celestial Reference System (interesting links)

[IR.2.] http://aa.usno.navy.mil/faq/docs/ICRS_doc.html, ICRS (narrative)

[IR.3.] http://ad.usno.navy.mil/OBSS/

[IR.4.] http://gaia.am.ub.es/SWG/ → Tools → Prediction of Gaia transits

[IR.5.] http://hpiers.obspm.fr/icrs-pc/, The ICRS

[IR.6.] http://hpiers.obspm.fr/iers/icrf/iau/icrf_rsc/icrf.def

[IR.7.] http://maia.usno.navy.mil/conventions.html

[IR.8.] http://planetquest.jpl.nasa.gov/Navigator/sim_library.html

[IR.9.] http://sci.esa.int/gaia/

[IR.10.] http://www.ari.uni-heidelberg.de/diva/

[IR.11.] http://www.jasmine-galaxy.org/

[IR.12.] http://www.rssd.esa.int/index.php?project=GAIA&page=Info_sheets_overview

[IR.13.] http://www.usno.navy.mil/FAME/

Annexes

Annex A:

Summary of Gaia features

A.1. MISSION GLOBAL PARAMETERS

Scan strategy	Continuous sky scanning
Effective Observing time (L)	5 years
Orbit	L2 (~1.5×10^6 km from Earth)
SAA (Sun Aspect Angle)	50° ±0.1°
Scan rate (ω)	60 arcsec/s
	1 revolution/6hr
	4 great circles scanned/day
Absolute rate error (ARE) along-scan	1.95 mas/s
Precession rate (ω_p)	0.2143arcsec/s
	0.2143°/h
	1 revolution/70 days
Absolute Pointing Error (APE)	< 30 arcsec
Focal plane passages per star per telescope	Average: 41 (TBC)
Stellar population (V<20), N_S	Avg: 25000 s/deg^2
	Max: 3·10^6 s/deg^2

Table A.1: Summary of main mission parameters

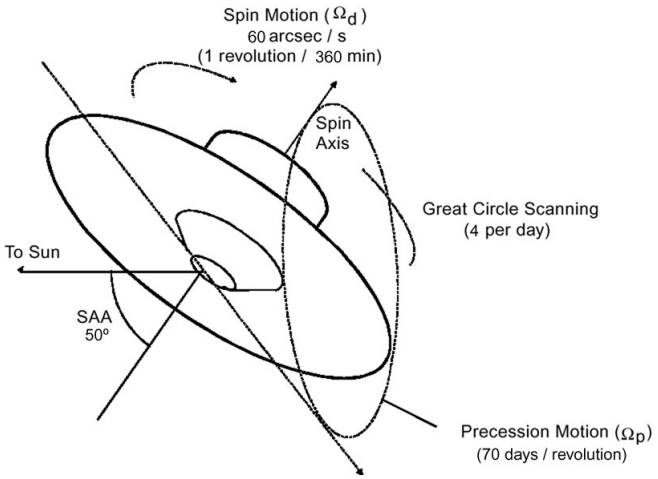


Figure A.1: Nominal Scanning Law for Gaia

A.2. PAYLOAD SUMMARY

One astrometric instrument (Astro) with 2 FOVs (Astro-1 and Astro-2):

- Six-mirror telescopes combining images from both FOVs (Astro-1 or preceding, and Astro-2 or following) into a common focal plane (3 last mirrors common to both FOVs).
- Large optical path length for practically suppressing optical distortion.

One radial velocity spectrometer and medium band photometer (Spectro):

- Three-Mirror telescope without intermediate image
  - Central part directed to the radial velocity spectrometer (RVS)
  - Outer parts directed to the medium-band photometer (MBP)
- Not bisecting Astro-1 and Astro-2 FOVs: the center of Spectro FOV is about 74° after Astro-2, or 180° before Astro-1 (TBC).
- Dispersion in the RVS: along-scan (TBC).

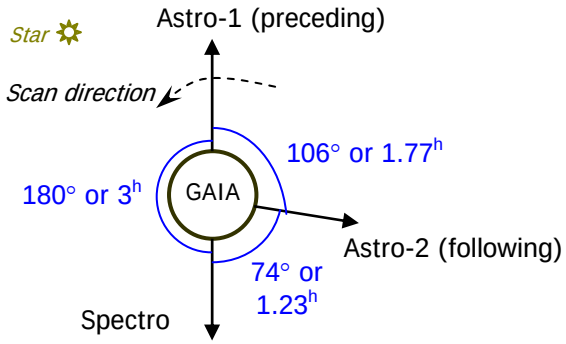


Figure A.2: Gaia scan sequence

A.3. ASTROMETRIC FOCAL PLANE

All the dimensions are noted as width × height (width = along-scan, height = across-scan)

A.3.1. General

Number of telescopes	2
Basic angle	106°
Total size	18×10 CCDs
	768×600 mm^2
	0.944°×0.737°
Along-scan distribution	2 ASM (1 per FOV), 11 AF, 5 BBP
Number of CCDs	180
Size per telescope	17 × ~9 CCDs
Number of CCDs	~153
Angle / linear scale	226.26 μm/arcsec
	4.42 mas/μm
Operation	TDI synchronized with spacecraft rotation (attitude adapts to TDI)
Total integration time (T)	~1587s (~26.5m) per telescope
	~3174s (~52.9m) total
Number of observations (N), average	~41 per telescope
	~82 total
Star flow per telescope	~267 stars/second (average)
	~32000 stars/second (maximum)

Table A.2: General features of Astro

A.3.2. Optics

Primary mirror (AL×AC)	1.4×0.5 m^2
Focal length	46.67 m
Field of view: Effective	~0.43 deg^2
Astrometric Field (11×9 CCDs)	0.66°× 0.66°
Whole Field	0.944°×0.737°
Optical center location: Astro-1	Central pixel of AF06, Row #5
Astro-2	50mm AC from Astro-1 center (AF06, #6)
Optical transmission	>0.86
Number of reflections	6
Type of coating	Ag
Overall aberration WFE	45nm rms
Airy radius @ λ=0.7μm	23.3×65.3 μm^2

Table A.3: Summary of optical features

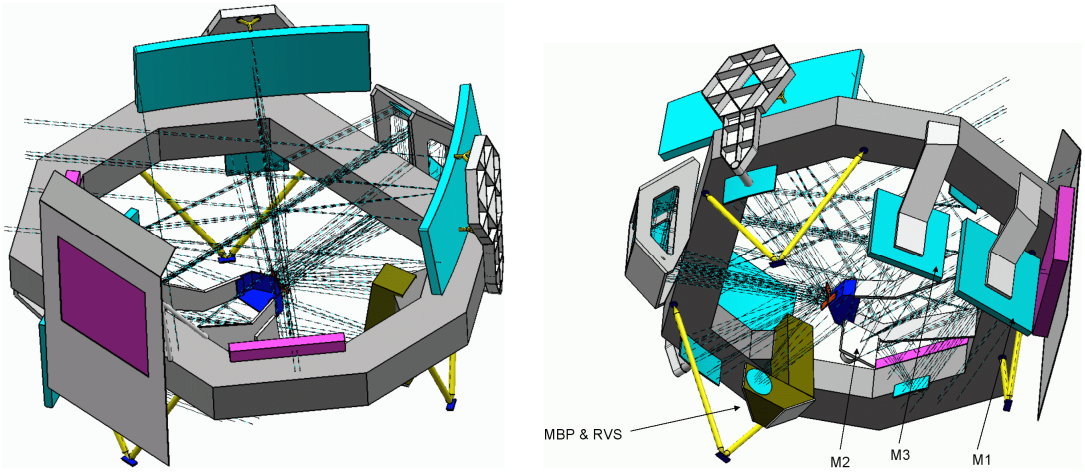


Figure A.3: Payload overview with ray traces

A.3.3. Timing Requirements

Nominal scan rates (sync with 60arcsec/s scan rate):	**Effective (Useful area)**	**Transit time (Incl. Dead zone)**
Pixel TDI period	736.6 μs	–
CCD (ASM & BBP2-5) integration time	1.92 s	2.21 s
CCD (AF & BBP1) integration time	3.31 s	3.61 s
AF01-AF11 integration time	–	39.71 s
Full focal plane integration time	–	56.58 s

Table A.4: Main timing features in the payload

Apparent object across-scan motion in AF due to precession (nominal maximum):	1.285 pixels/s
	4.253 pixels/active surface of a CCD
	51.027 pixels/AF
	0.778 s for a displacement of 1 pixel

Table A.5: Across-scan motion equivalences

A.3.4. CCD Specifications

	Pixels	Linear (mm^2)	Angular ($arcsec^2$)	Angular ($degrees^2$)
Pixel size	–	0.010×0.030	0.0442×0.1326 (44.2×132.6 mas^2)	$1.23 \cdot 10^{-5} \times 3.68 \cdot 10^{-5}$
CCD Size (ASM & BBP2-5):				
Pixel area	2600×1966	26×59	115×261	0.0319×0.0725
Incl. Dead zone	–	30×60	133×265	0.0369×0.0736
CCD Size (AF & BBP1):				
Pixel area	4500×1966	45×59	199×261	0.0553×0.0725
Incl. Dead zone	–	49×60	217×265	0.0603×0.0736

Table A.6: CCD sizes in Astro

Dead zones:	Linear (mm)	Angular (arcsec)	Pixels	Fill factor (%)
Along-scan	≤ 4	≤ 17.67	≤ 400	≥ 86.6 (ASM&BBP) ≥ 91.8 (AF)
Across-scan	≤ 1	≤ 4.42	≤ 33.3	≥ 98.3
Global	–	–	–	≥ 85.2 (ASM&BBP) ≥ 90.3 (AF)

Table A.7: Dead zones in the Astro CCDs

Measure limits:	Charge (e^-)
Pixel full well capacity	190 000
Sample full well capacity	350 000
Output FET saturation	300 000
G magnitude at zero e^-	25.56

Table A.8: Flux measurement limits in Astro CCDs

A.3.5. Parts of the Focal Plane: ASM, AF and BBP

	ASM	AF	BBP
Size (CCDs)	2×10	11×10	5×10
Size (CCDs) per telescope (TBC)	1×9.17	11×9.17	5×9.17
Linear size (mm^2)	60×600	539×600	169×600
Angular size ($arcsec^2$)	266×2650	2387×2650	749×2650
Angular size (as^2) per telescope	133×2430	2387×2430	749×2430
Angular size ($degrees^2$)	0.074×0.736	0.663×0.736	0.208×0.736
Angular size (deg^2) per telescope	0.037×0.675	0.663×0.675	0.208×0.675
Integration time (s) per telescope	1.9	39.7	10.9
ASM magnitude zero	25.71mag (TBC)	–	–

Table A.9: Organization of the Astro focal plane

		ASM		AF			BBP
		ASM1	ASM2	AF01	AF02-10	AF11	
Readout samples (pixels)	**G = 2-8**	2×2		1×4	1×4	1×4	1×4
	G = 8-12			1×2	1×2	1×2	1×2
	G = 12-16				1×12	1×14	1×12
	G = 16-20						
Transmitted samples (pixels)	**G = 2-8**	2×2		1×4		1×4	1×4
	G = 8-12			1×2		1×2	1×2
	G = 12-16			1×12		1×14, 2×14[27]	1×12
	G = 16-20	2×2, 4×4 [1]				1×14, 3×14[1]	1×12, 2×12, 3×12[1]
Acquired window (samples)	**G = 2-8**	N/A [28]		16×6	16×6	16×6 (TBC)	16×6
	G = 8-12						
	G = 12-16			12×6	12×1	68×1	16×1
	G = 16-20			6×6	6×1		
Transmitted window (samples)[29]	**G = 2-12**	8×6		16×6		16×6	16×6
	G = 12-16	48[30]		12×1		40×1[31]	16×1
	G = 16-20	25[32]		6×1		26×1[33]	10×1
Total rms RON (e^-/sample) (TBC)		8.7		9.7	3.7	5.1	4.5

Table A.10: Summary of the sampling scheme in Astro

[27] Extended binning only in the borders (see Høg 2002b, fig. 1), obtained with numerical binning
[28] All the pixels of the CCD are read and binned into 2×2 samples
[29] There are some extra options (to reduce the amount of data transmitted in crowded areas), which are not included in this table: Short/No AF11 and Short BBP (see Høg et al. 2003)
[30] 6×6 plus 3 additional samples around each corner (see Høg 2002b, fig. 1)
[31] 12 samples in the center plus 14 extended samples (2×14 pixels each) at the borders
[32] 5×5, with the samples from the corners containing data from 4 samples (see Høg 2002b, fig. 1)
[33] 6 samples in the center plus 10 extended samples (3×14 pixels each) at the borders

Annex B:

Sampling Scheme

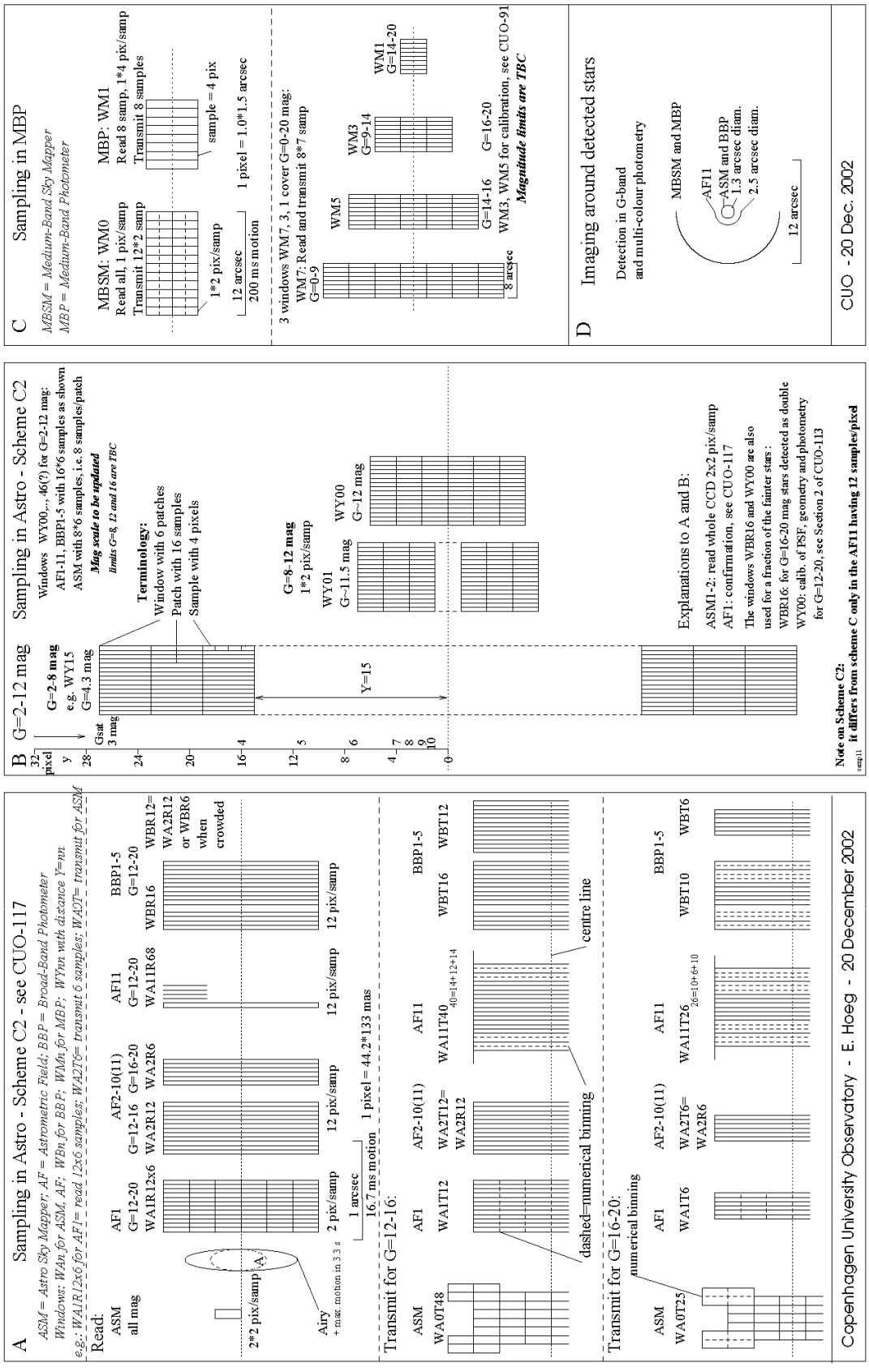


Figure B.1: Sampling scheme used in our work (extracted from GAIA-CUO-117).

Annex C:

Software reference

In this annex we briefly review the several software applications that we have developed in order to test the correct operation of the designs described in this work. A user manual is included, as well as an overview of the main functions of each application. The code listings have obviously been omitted in order to avoid a huge document size, since the code developed during this work amounts to almost 14000 lines. The following are the approximate number of code lines developed for each application:

- Optimization of the Time Data Codification (MatLab): 3500 lines.
- Preliminary data compression tests (C): 1500 lines.
- Static TM CODEC (C++): 5500 lines.
- Visualization tools for TM CODEC statistics (MatLab): 220 lines.
- Dynamic TM CODEC preliminaries (C++): 2900 lines, plus some of the libraries included in the Static TM CODEC.

C.1. Optimization of the Time Data Codification (TDC)

C.1.1. User manuals of the simulators

C.1.1.1. Averaged simulations: gaia_otdc_main.m

This software calculates the data rate output by the data set headers and Astro instrument using *averaged* values, this is, using a single value of *average star rate*. Several codification parameters, as well as the basic mission parameters, can be changed while verifying their effect on the average data rate. Figure C.1 shows a snapshot of the main window of this program with the several areas indicated by the arrows. The codification and mission parameters are explained in 7.3.2. While the latter are (more or less) fixed by engineering teams and by the mission design itself, the codification parameters are the variables that we can explore for obtaining a more optimized transmission scheme. When modifying any of the parameters, both the graphical and numerical results are automatically updated. The types of graphical results that we can choose include the data rate saved (compared to GDAAS2-type codification), bits per star saved and total data rate. By dragging the mouse over the graphical plot we can zoom an area of interest, e.g., the low star densities. The numerical results include several statistics of channel occupation and data rate savings for different star densities, as well as the minimum star density from which the codification scheme is optimal.

Although we could obtain a good codification scheme by simply testing these parameters, some optimization routines have also been developed in order to ease this process. They can be executed by clicking on the buttons of every codification parameter. Figure C.2 shows this routine for the TDI offset resolutions (`gaia_otdc_otor.m`). In this dialog box we can modify both parameters by entering their values in the *edit boxes*, by moving the *sliders*, or by simply clicking on the main plot (where the values selected are indicated by a colored cross). The main plot represents in different colors the data rate generated by the data set headers when using different values of *TdiOffResA* and *TdiOffResS*, while the secondary plots are cuts of this main plot. Numerical results are also output by this routine, including the independent field sizes as well as the total data rate for the headers.

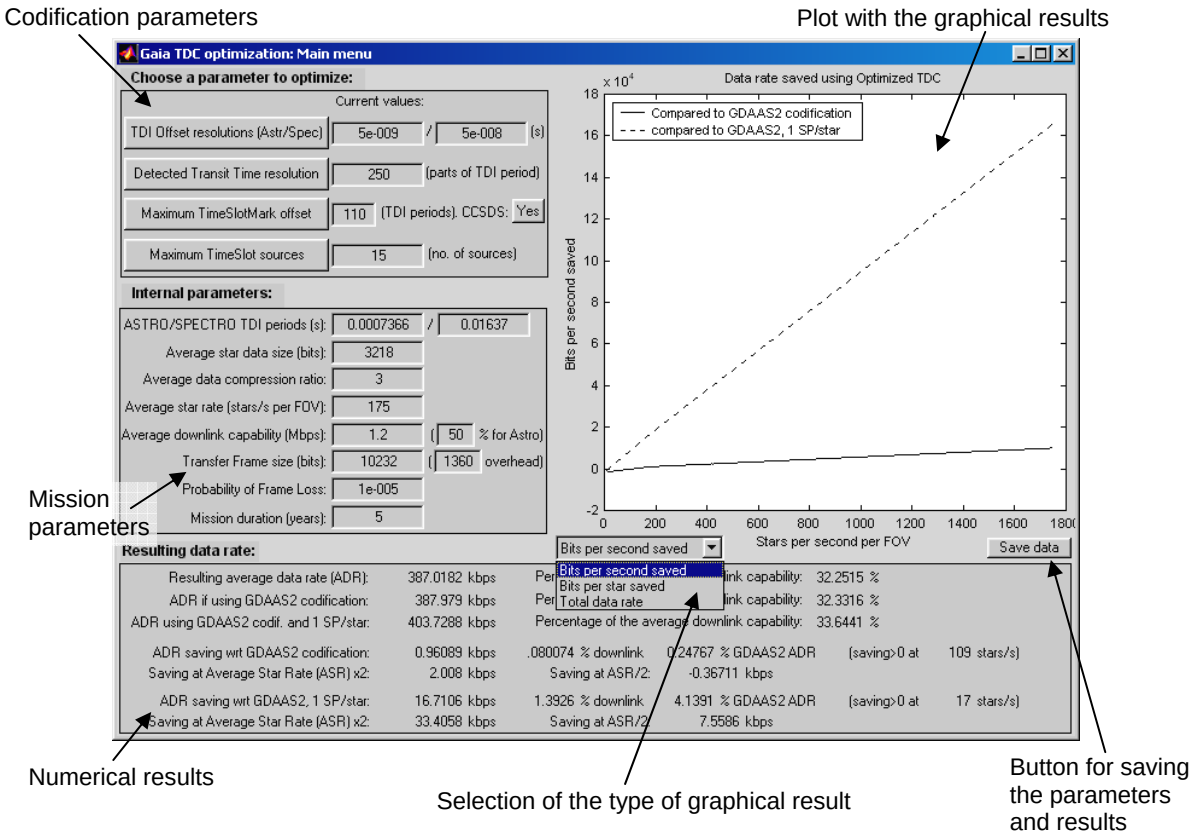


Figure C.1: Elements of the main window of the averaged simulations software

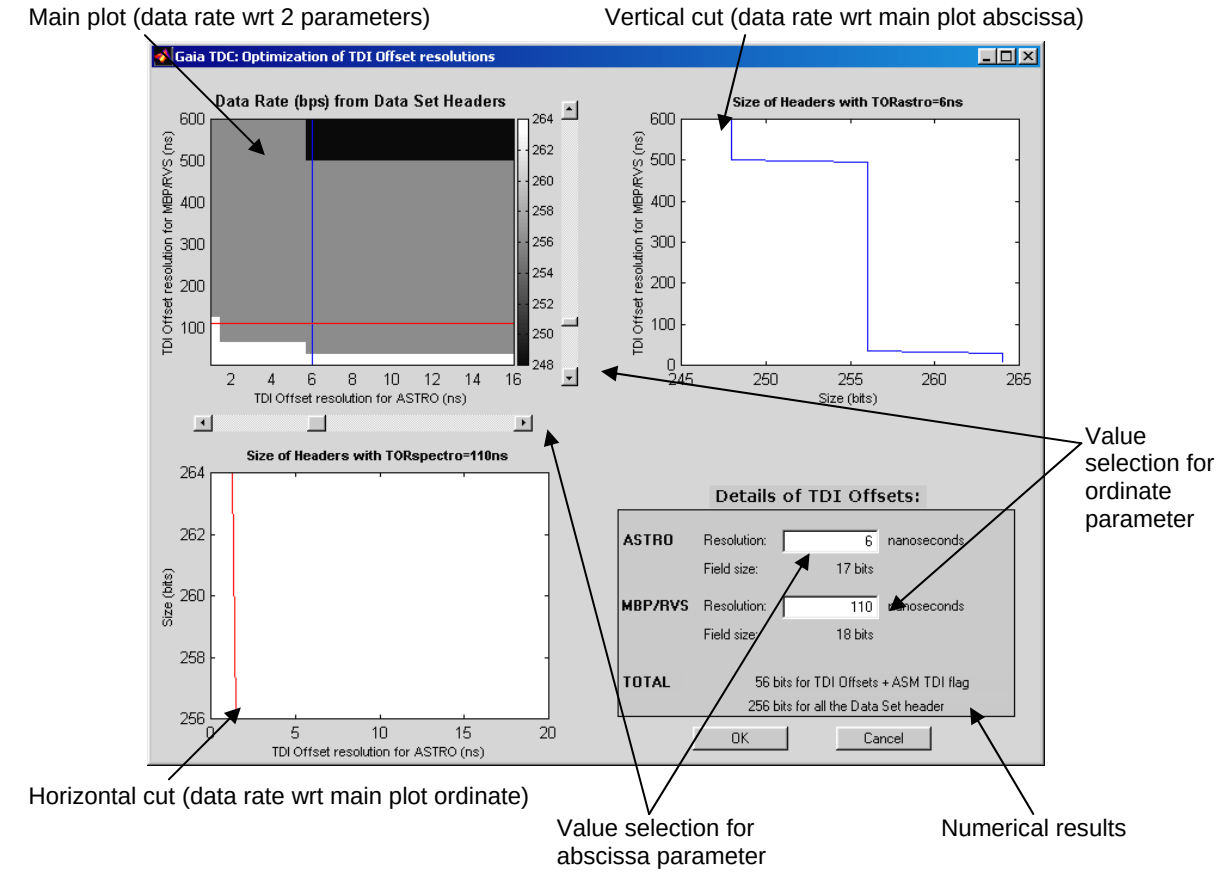


Figure C.2: Elements of the optimization routine for TdiOffRes

The modification of the TDI offset resolutions does not affect very much the final data rate. On the other hand, the three other parameters have a more important effect and, furthermore, they have a tight interrelation. This is why we developed a single optimization routine, `gaia_otdc_oast.m`, for optimizing the three parameters. The click on one or another parameter (in the main window) simply executes this routine with one or another configuration. Figure C.3 shows its execution after a click on *Maximum TimeSlotMark Offset*, where we can see a similar structure than used for the TDI offset resolutions: a main plot showing the data rate as a function of two parameters, two secondary plots with cuts of the main plot, selectors of the parameter values, and numerical results.

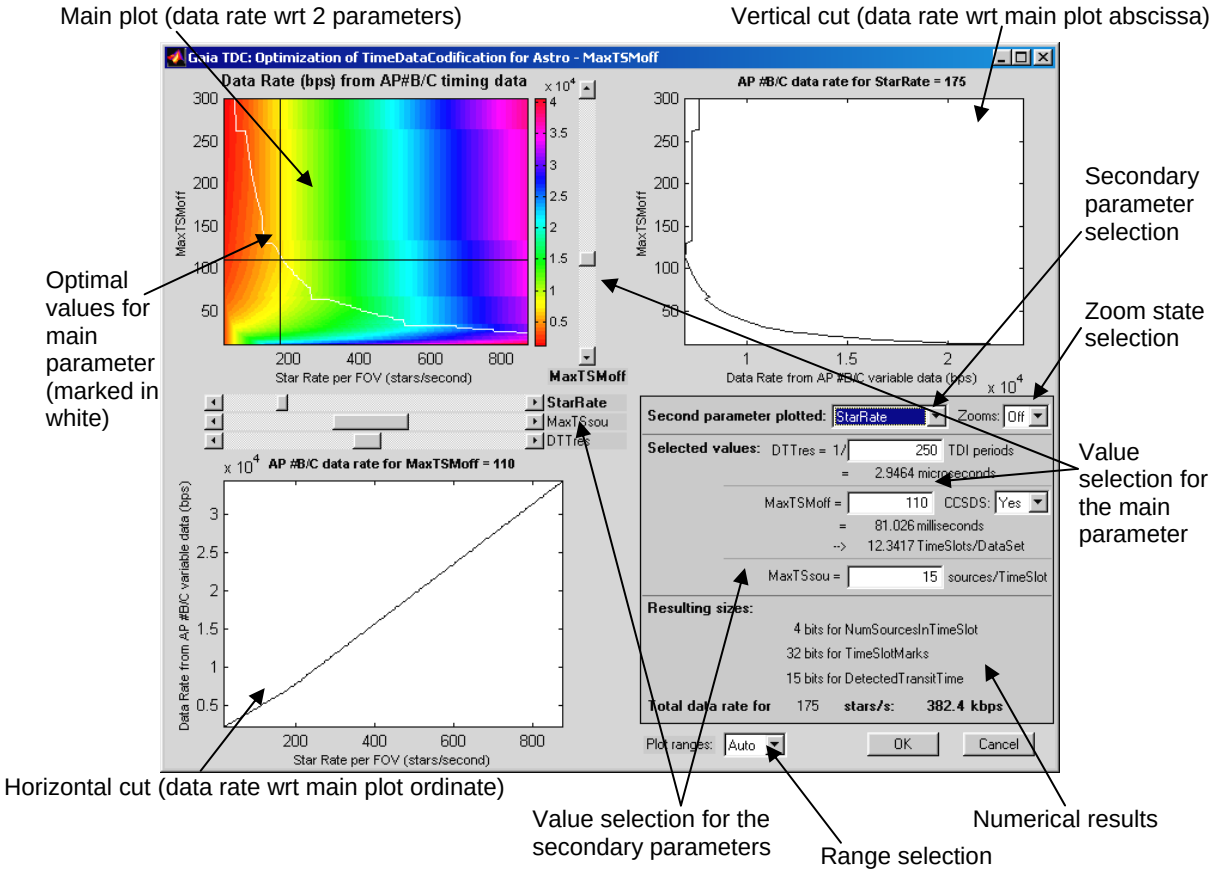


Figure C.3: Elements of the optimization routine for Astro codification parameters

The main difference in this dialog is that we have more parameters to move, whether entering their new values numerically, moving the sliders, or clicking on the main plot. Furthermore, after selecting the main parameter (the button pressed in the main window selects this), three different secondary parameters can be selected for plotting the graphical results: star rate and the two remaining codification parameters (*MaxTSsou* and *DTTres*, in the case illustrated in the figure). It is very important to note that the modification of *any* codification parameter also keeps recorded, this is, they are not only used for testing but are also kept when we exit the routine and return to the main window.

As we can see in the example, the effect of these parameters is much more complicated than in `gaia_otdc_otor.m`, so the optimal values of the main parameter are automatically calculated and shown in white in the main plot. Ideally, the codification system should always move on this white curve.

The dialog also allows turning on and off the zoom option of the plots. When zooms are off, we can select new parameter values by clicking on the main plot. Another feature is the automatic or fixed ranges for the three plots: usually they are automatic, but if we want to compare several parameter values we can fix them, although in this case we will lose some graphical resolution.

C.1.1.2. Adaptive system design: gaia_otdc_adap.m

This improved software calculates the data rate output by the data set headers and Astro instrument when using an *adaptive* system combined with a *star rate histogram*, this is, it applies the corresponding codification parameters to every star rate range and integrates the total telemetry saving, offering afterwards an average (but more realistic) value. All of the codification parameters except *MTO* (maximum TSM offset), as well as the mission parameters, are fixed at the beginning of the code for simplicity of execution using the optimal values previously found with `gaia_otdc_main.m`. Figure C.4 shows a snapshot of this.

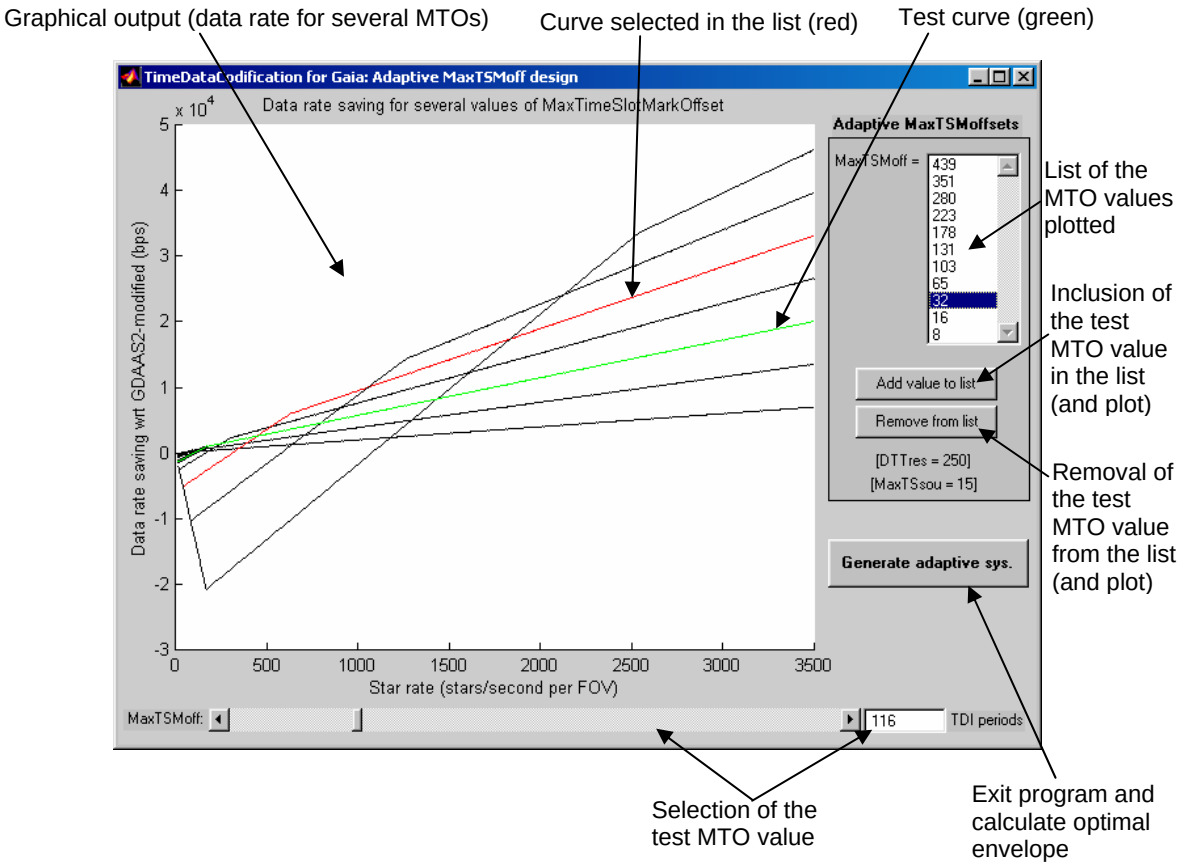


Figure C.4: Software for simulating the adaptive codification system

The software calculates a first set of optimal *MTO* values that are automatically included in the list, using them for plotting the first set of telemetry saving curves. Afterwards, the user has to test several *MTO* values by moving the slider (or entering numerical values), adding them to the list when they appear to be optimal. The plot can be zoomed by dragging the mouse over it, for better appreciating the savings at low star rates. Finally, redundant values should be removed from the list, thus obtaining the minimum required set of optimal *MTO* values. When finished, the user exits the program and it offers graphical and numerical outputs of the star rate histogram used, the data rate saved, and the bits per star saved.

C.1.2. List of functions used in the simulations

These are the functions developed for implementing the data rate models described in 7.3.1:

- `gaia_otdc_dshr.m`: Calculation of the data rate from the AP #A source packets (data set headers). The parameters *TdiOffResA* and *TdiOffResS* are the only affecting the result (besides the TDI period values), and can be entered as vectors (for testing several values). The output is the total bits generated per second.
- `gaia_otdc_adbw.m`: Calculation of the data rate from APs #B and #C (Astro-1 and Astro-2). Several codification and mission parameters affect the result, specially *DTTres*,

MaxTSMoff, *MaxTSsou* and the star rate. Any two of these parameters can be entered as vectors for testing several values. The results include the total data rate, the data rate affected by the codification parameters, and several field sizes.

- `gaia_otdc_g2dr.m`: Calculation of the data rate from a GDAAS2-type codification, including both the data set headers and the Astro data. We can select the use of a *worst* codification (generating one source packet per star) or an *equivalent* codification. Several parameters affect the result, and only *StarRate* can be entered as a vector. The output is the total data rate.

The following are the GUI-based (graphical user interface) functions used for optimizing the codification parameters. All of them are based on a recursive operation: the function first creates and initializes the GUI dialog and, depending on the events produced by the user action, the function calls itself with different parameters (for updating the output, modifying GUI elements, etc.). All of them make intensive use of the previous functions for calculating the total data rates and data rate savings:

- `gaia_otdc_main.m`: Main program for optimizing the parameters using averaged calculations.
- `gaia_otdc_otor.m`: Sub-program called by `gaia_otdc_main.m` for optimizing the TDI offset resolutions.
- `gaia_otdc_oast.m`: Sub-program called by `gaia_otdc_main.m` for optimizing the Astro codification parameters.
- `gaia_otdc_adap.m`: Main program for selecting the optimal *MTO* values to be used in an adaptive codification system.

C.2. PRELIMINARY DATA COMPRESSION AND TM CODEC

C.2.1. Simulation of preliminary data compression systems

In order to test the preliminary designs described in chapter 11, a software application was developed in C language which included several libraries –one for every data pre-compression system. This software can be considered the predecessor of the TM CODEC, which actually reused some part of its operating philosophy. The following are the main modules developed for testing the preliminary data compression systems:

- `gaia_def.h`: Header file containing definitions of the Gaia design, such as dimensions of the focal plane or bit sizes of every data field.
- `gaia_err`: Small library for elementary error handling.
- `gaia_bit`: I/O library for making possible bit-level accesses to disk files.
- `txt2bin.c`: Text-to-binary converter. It reads an ASCII file generated by GASS (with the adequate format) and converts it to a binary file using the data field sizes indicated in gaia_def.h. In this way, the output file reveals the *actual* size of the telemetry data generated and can be compressed with standard systems if necessary. This was actually the main goal of this program: to obtain raw TM sizes and compression ratios achieved with standard systems.
- Data pre-compression routines (as standalone C programs), each receiving binary files converted by txt2bin and saving the pre-compressed result to another binary file. The following routines were implemented and tested:
  - FAE (*Fully Adaptive Encoder*), as described in chapter 10.
  - FAMbE (Fully Adaptive Model-based Encoder), as described in chapter 10.
  - FADE (Fully Adaptive Differential Encoder), as described in chapter 10.

- FAMbDE (Fully Adaptive Model-based Differential Encoder), combining FADE with FAMbE. The model is not only used for absolute patches but also for differential ones (although the latter use a less steep model).
- FAPADE (Fully Adaptive Patch-Average Differential Encoder), based on FAMbDE. It calculates an "average patch" (using AF01-AF16), looks for the actual patch that is most similar to this average, and differentially codes every patch wrt this one.
- FASADE (Fully Adaptive Sample-Average Differential Encoder), based on FAPADE but composing the "average patch" from separate samples.
- FARODE (Fully Adaptive Re-Ordering Differential Encoder), based on FAMbDE but reordering the AF01-16 patches looking for the smoothest possible changes.
- FAMbDE3 (FAMbDE-3 differentiators), combining FAMbDE and FAPADE but using 3 independent differentiators, always coding the smallest of the three differences.

C.2.2. Telemetry CODEC

The TM CODEC version used in this work is 1.1 (static), released on 2005, January 26th. The package is composed of the following applications:

- txt2bin 1.0.2: Converts GASS v2.2 TM data (in ASCII files) to binary (and quantized) files. It also statistically analyses the data, offering a set of files with the field density curves, telemetry curves and flux histograms.

  It assumes some limitations in the Sampling Scheme (e.g., WinMode cannot be 2, AF11mode cannot be 1). Nevertheless, it is prepared for the full scheme, so that it could be updated with very few changes in the code.
- bin2txt 1.0.2: Restores binary and quantized data to GASS format again (ASCII files). With this, one can visually verify the quantization errors in the TM data (mainly in the flux data).
- GaiaSPaP 1.2.2: Stream Precompressor and Partitioner. It prepares GASS binary data (converted by txt2bin) so that much higher compression ratios are achieved when applying standard compression systems afterwards. It also statistically analyses the data, offering the entropy and Shannon limit, as well as files with the histograms of the main data fields.
- GaiaSRaD 1.0: Stream Recombiner and Decoder. It restores the set of sub-streams generated by GaiaSPaP into a single file again, with the original GASS format again (but quantized and in binary format). bin2txt can be applied afterwards in order to obtain the ASCII file again.

Also some auxiliary libraries are included:

- Gaia Miscellaneous Tools 1.0
- Gaia EDL Tools (Error, Display and Log) 1.0
- Gaia File 1.3
- XML Parser 1.2

The full package can be executed with the following scripts:

- GaiaPrecomp 1.0: It inputs a GASS TM file and executes txt2bin plus GaiaSPaP, thus obtaining the following products with a single command:
  - Binary (and quantized) file
  - Statistical log files
  - Pre-compressed sub-streams
  - Pre-compressed combined stream (obtained with a TAR of the several sub-streams)
- GaiaStdComp 1.0: It inputs a .bin file (obtained with txt2bin or GaiaPrecomp) and applies several standard compressors on it.

- GaiaSpapComp 1.0: It receives the set of sub-streams and combined stream obtained with GaiaPrecomp and applies several standard compressors on it. The resulting files are:
  - Compressed sub-streams
  - Compressed combined stream (standard compressors applied to the TAR file obtained from the sub-streams)
  - Combined compressed stream (TAR file obtained from the compressed sub-streams)
- GaiaRestore 1.0: It takes the set of sub-streams (obtained with GaiaSPaP or GaiaPrecomp) and applies GaiaSRaD and bin2txt to it, thus offering an ASCII file again (which can be used to verify the quantization errors).

Finally, the package also includes two MatLab® programs, one for plotting the histograms, telemetry and field density curves obtained with txt2bin, and the other for plotting the probability density functions obtained with GaiaSPaP.

C.2.3. Dynamic Telemetry CODEC preliminaries

Although the complete Dynamic TM CODEC has not been implemented yet, we started developing part of its core: the telemetry model and data loaders. As explained in chapter 9, the Dynamic TM CODEC will be configured using XML files. Complex data structures for storing this have been developed, offering a very high flexibility in the configuration, operation and data processing. Then, the TM model loader (which is one of the crucial parts of the Dynamic TM CODEC) loads and parses the XML configuration file to memory, also offering tracking and processing routines and hence easing the overall implementation –thanks to a completely modular design. An additional library has been developed to verify the correct load of this telemetry model; this library parses the data structures stored in the computer memory and generates a human-readable HTML file, an example of which is shown in figure C.5. Here we can appreciate in part the complexity of the data structures that it can manage, although conversion and data processing routines have not been included.

Once we achieved loading these complex data structures from the TM Model to memory we started developing a TM Data Loader, this is, a library in charge of tracking the TM Model and loading to memory the several data fields from the TM file. Obviously, it does not load all of the TM data to memory, but it loads them data set by data set. In this way, complex data processing operations can be performed –following the *script* entered with XML files. The remaining steps towards a full implementation of a Dynamic TM CODEC are described in chapter 9 (sections 9.4.3.5 to 9.4.3.10).

Telemetry Model information:
- **File:** GASSv2.xml
- **Identifier:** GASS2
- **Description:** TM Model for GASS v2.1 (Gaia-2), as defined in UB-GDAAS2-TN-003 v2.1 (2004-Feb-11).
- **Type:** Datasource

Telemetry Model tree:

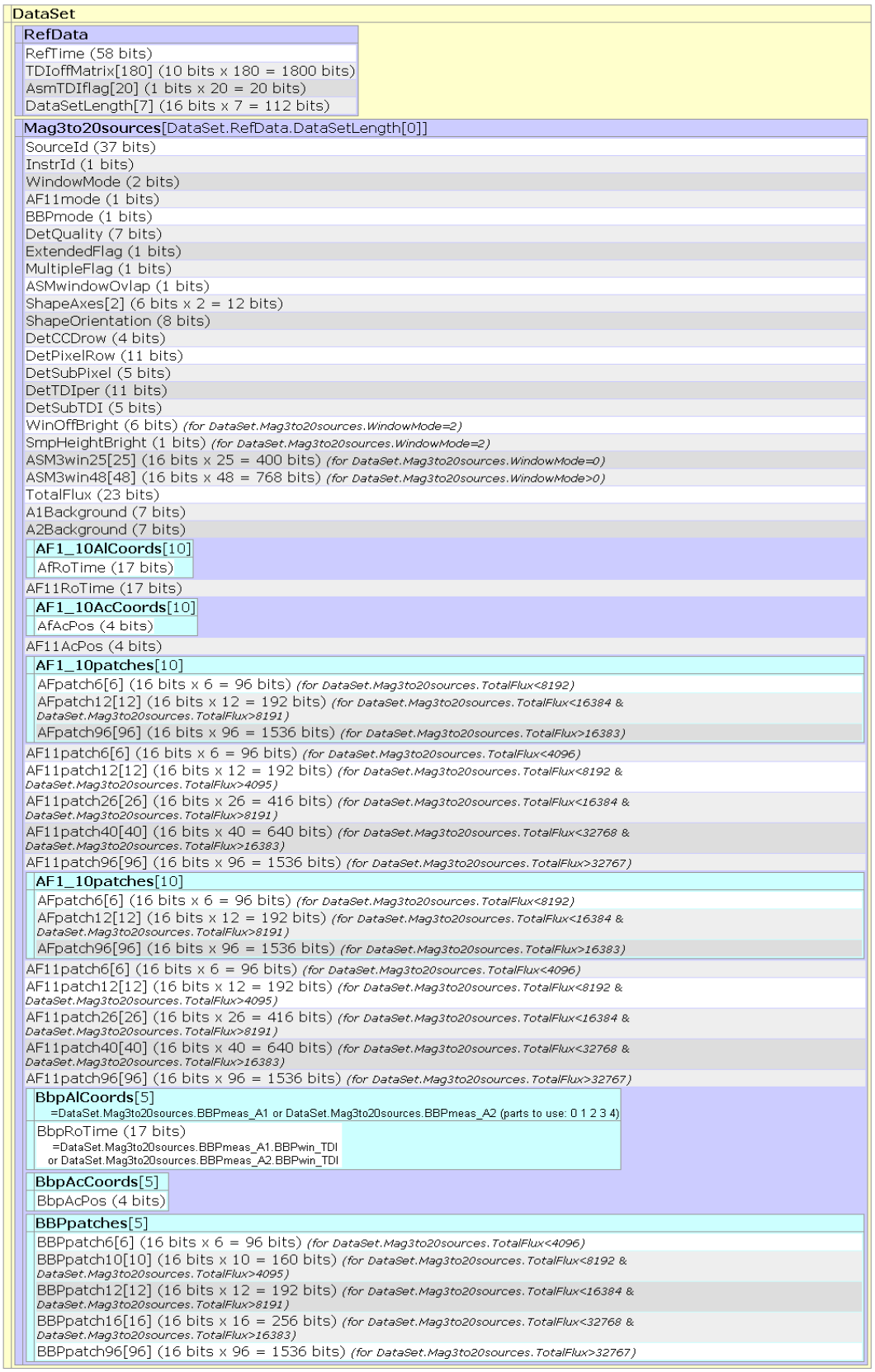


Figure C.5: Sample of an HTML-converted TM model after being parsed and loaded