UNIVERSITAT JAUME I

DPTO. DE LENGUAJES Y SISTEMAS INFORMÁTICOS

UNIVERSITAT JAUME·I

Viewpoint-driven Simplification of Polygonal Models using Information Theoretic measures

PhD Dissertation

Pascual Castelló Boscá

Advisors: Miguel Chover Sellés, Mateu Sbert i Casasayas

Castellón

October, 2007

Viewpoint-driven Simplification of Polygonal Models using Information Theoretic measures

Pascual CASTELLÓ BOSCÁ

Work done under the supervision of Professor
Miguel CHOVER SELLÉS and Professor Mateu SBERT I CASASAYAS
and presented to the Universitat Jaume I
in partial fulfillment of the requirements for the degree of Doctor
of Philosophy by the Universitat Jaume I

Castellón, October, 2007

This research was supported in part by the projects ADVANCED TOOLS FOR DEVELOPING HIGHLY REALISTIC COMPUTER GAMES (project IST-2-004363), subsidized by the European Union and MEJORA Y APLICACIÓN DE LA TECNOLOGÍA DE JUEGOS EN REALIDAD VIRTUAL Y CONTENIDOS WEB (TIC2004-7451-C03-03), subsidized by the Science and Technology Interministerial Commission.

To my family and friends

Resumen

Los modelos poligonales actualmente dominan el campo de los gráficos interactivos. Esto es debido a su simplicidad matemática que permite que los más comunes algoritmos de visualización se implementen directamente en el hardware. Sin embargo la complejidad de estos modelos (medidos por el número de polígonos) crece más rápido que la capacidad del hardware gráfico para visualizarlos interactivamente. Las técnicas de simplificación de polígonos ofrecen una solución para tratar estos modelos complejos. Estos métodos simplifican la geometría poligonal reduciendo el coste de visualización del modelo sin una pérdida del contenido visual del objeto. Esta idea aún sigue vigente aunque es una idea ya antigua en gráficos por ordenador. Durante los últimos años ha surgido un gran abanico de métodos de simplificación. La mayoría ha abordado el problema de la simplificación desde el punto de vista geométrico. Es decir, elaborando métricas que permiten guiar la simplificación calculando el error cometido en cada paso utilizando una medida puramente geométrica. Recientemente se han desarrollado nuevos métodos que intentan guiar el proceso de simplificación mediante una medida de similitud visual. En otras palabras, que los modelos simplificados se vean de forma parecida cuando se visualizan. El error geométrico es uno de los factores que influye en la similitud visual pero no es el único. Otros factores como las siluetas, las propias oclusiones y trasparencias, los atributos de superficie, etc. influyen notablemente. En esta tesis se presenta un nuevo método de simplificación de mallas de polígonos. Este método realiza una simplificación guiada por el punto de vista, acometiendo una simplificación cuyo objetivo es garantizar la similitud visual. Esto permite que muchas aplicaciones cuyo objetivo sea la visualización interactiva como por ejemplo los juegos de ordenador se beneficien en buena medida. Se han propuesto diferentes métricas para conducir el método de simplificación desarrollado, todas ellas están basadas en la Teoría de la Información. El empleo de una métrica u otra permite llevar a cabo tres grandes tipos de simplificaciones. En el primer grupo se consigue una simplificación cuyo factor primordial es la similitud visual lo que conduce a la obtención de modelos simplificados cuya geometría oculta ha sido simplificada en gran medida o en su totalidad. En el segundo grupo de métricas se aborda la simplificación con el objetivo de la similitud visual pero respetando en mayor medida la geometría oculta del modelo. Con lo que el error geométrico es menor que en grupo anterior a costa de un mayor error visual. Finalmente, el tercer grupo permite que se pueda acometer una simplificación que intenta preservar las regiones visualmente salientes de la malla mediante la aplicación del concepto de *saliency* de malla, definido a partir de la divergencia de *Jeshen-Shannon.* Estas pequeñas regiones se mantienen mejor si se hace uso de este método, de otra forma serían eliminadas ya que tienen un coste de simplificación bajo.

Palabras clave: simplificación poligonal automática, modelado multirresolución, nivel de detalle, selección del punto de vista, percepción visual, Teoría de la Información.

Acknowledgments

First of all, I would like to thank to my thesis advisors, Miguel Chover and Mateu Sbert, for their help and useful advice during the development of this thesis.

I wish to thank my colleagues in the computer graphics group of the Universitat de Girona (UdG), who helped me at the beginning of the work on this thesis. I would also like to thank Miquel Feixas at UdG for spending many days discussing research ideas with me and providing many insightful comments and suggestions on this research.

Finally, many thanks to my friends and colleagues at the computer graphics group at the Universitat Jaume I (UJI) for their encouragement and support.

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Background and Motivation

Traditionally, polygonal models have been used in computer graphics as the fundamental primitive for representing 3D objects. Polygonal meshes allow objects of arbitrary geometry to be represented properly at any desired degree of accuracy. In addition, polygonal surfaces are easy to build from other surface representations. Yet, perhaps the most important reason is that polygons are the main, and sometimes the only, primitive supported by common graphics hardware and software rendering systems. Moreover, most algorithms and methods for manipulating and creating surfaces have been designed for polygonal meshes.

Nowadays, many devices used in medical and scientific visualization create polygonal models. These include, for instance, laser range scanners, computer vision systems and medical imaging devices. These devices create models which often consist of thousands or even millions of polygons. A great number of polygons are necessary to represent curved surfaces. These massive data sets normally exceed the capabilities of current display units and mesh processing tools. Hence, in order to deal with these complex polygonal models, mesh simplification is used. *Mesh simplification* or *mesh decimation* consists in generating approximations to highly detailed surfaces with a smaller number of polygons. Figure 1.1 shows two different approximations of a polygonal model. In 1976, Clark [Cla76] proposed the use of simplified models to improve the frame rate, that is, the measurement of the frequency (rate) at which an imaging device produces unique consecutive images called frames. Frame rate is most often expressed in frames per second (fps). Different representations can be created for an object and the most adequate approximation can be chosen depending on the distance, visibility, importance, etc. Objects which are close to the viewer need a greater amount of detail than distant objects. This concept gives rise to *multiresolution modeling* [Gar99, RLB+02] and a lot of algorithms and methods have been developed in interactive graphics for constructing and manipulating *levels of detail* (LOD). Figure 1.2 shows three levels of detail for a polygonal model. Level of detail is very important in real-time systems. Every system has a fixed frame buffer fill, transformation and lighting throughput, and bus bandwidth. In order to maintain a constant frame rate, it is necessary to use multiresolution models.

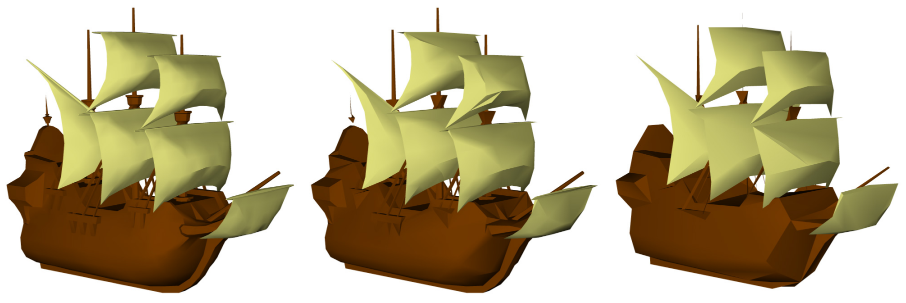

(a) Original (b) 68% (c) 85%

Figure 1.1: Two different approximations automatically created for a polygonal model. Models (b) and (c) are represented with fewer polygons than the original model (a)

Other data sources for polygonal models include modeling software for computer-aided design (CAD) and computer games. These models are also very complex with highly detailed surfaces that are produced by surface reconstruction and isosurface extraction methods. Indeed this creates very dense meshes with a uniform distribution of points over the surface. Moreover, many CAD models involve large assemblies of many small objects which may be partially hidden. However, level of detail is still created manually. Although polygonal simplification is a well-studied problem (there are a lot of different approaches in the computer graphics literature), most of the work has been carried out by developing simplification methods that minimize the geometric error committed in the approximations. But in video games and visualization, the most important point is the visual quality. Current video game artists often do not use automatic simplification and continue to make the approximations by hand. Therefore, the recent work on simplification has been conducted to design methods that guide the simplification process by metrics that take into account the visual error. This is the main motivation of this thesis.

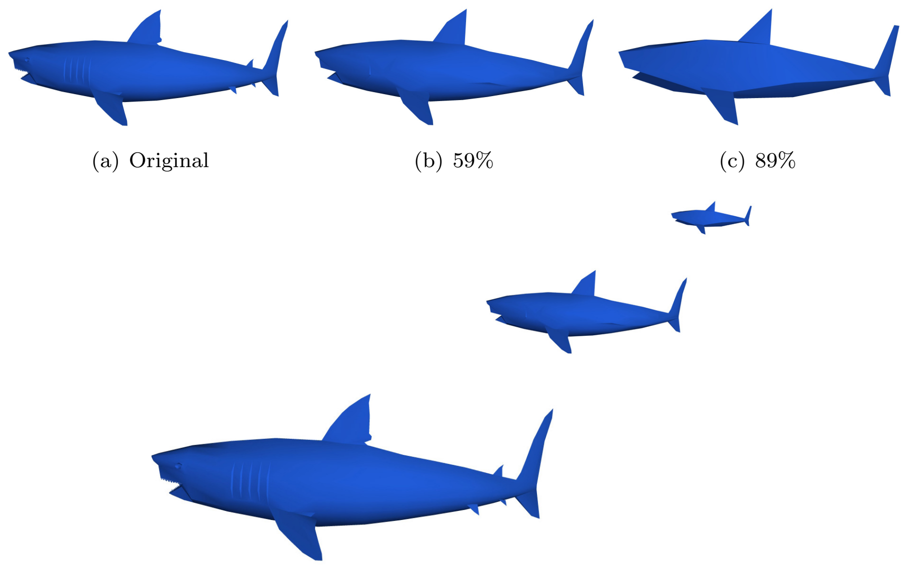


Figure 1.2: Three levels of detail for a polygonal model. The LOD drawn depends on the distance. Models (b) and (c) are represented with fewer polygons than the original model (a)

1.2 Problem Statement

The ultimate goal of the *automatic simplification* is to produce approximations with a desired number of polygons that minimizes the loss of quality with respect to the original model. At the moment, there is no panacea in simplification. That is, there is no algorithm or technique that is the best for all possible scenarios. Different domains require different error metrics and simplification methods.

In many medical and scientific applications, models must have precise geometric tolerances with respect to the original model. For those cases, simplification algorithms based on a geometric error metric are better suited. However, in many visualization applications, the most important requirement is visual similarity. In other words, what it is really important is what is visible. Other aspects of the model are irrelevant.

In this thesis, we present a new family of methods for polygonal simplification. These methods obtain approximations that preserve visual similarity. The domain considered here is that of applications for interactive rendering and visualization such as video games. Several different metrics are proposed to conduct the simplification. These methods and metrics will be described in the following sections.

1.3 Contributions

The main aim of this thesis is to propose a new method for performing polygonal simplification. This method can be used in conjunction with several error metrics based on Information Theory. Below we will provide a brief description of each contribution.

A survey of techniques for computing projected areas

The simplification algorithm presented here is based on viewpoint selection measures. These measures are used to quantify the error committed at each step of simplification and therefore guide the simplification process. All the viewpoint selection measures included in this thesis take the projected areas as input data. In general, the error metric has to be calculated at every step in the simplification algorithm. Thus, it is necessary to find a technique that allows this error metric to be computed quickly. The temporal cost of the simplification algorithm depends above all on this computation.

Firstly, several techniques that allow the projected areas of polygons to be computed in current graphics hardware are analyzed [CCSF05, CSCF06]. To compare these techniques, we compute viewpoint entropy which is defined from the projected areas of polygons. Viewpoint entropy is a metric that has been successfully used to obtain the best viewpoints in a scene or object. It will be used later in the viewpoint-driven simplification algorithm together with some other viewpoint selection measures.

As pointed out above, the results from this survey can be useful not only to compute viewpoint entropy but also to compute other metrics based on projected areas such as mutual information and the Kullback-Leibler distance. These metrics will be described in more detail in the following chapters.

Viewpoint entropy-driven simplification

Most existing simplification methods carry out the simplification process by minimizing some measure of geometric deviation between the two surfaces [SZL92, RB93, EDD$^+$95, HHK$^+$95, Hop96, CVM$^+$96, COM98, GH97, LE97]. In recent times, only a few methods attempt to perform a simplification using metrics, so that the two models appear similar when rendered [LT00, LH01, ZT02, LVJ05]. In fact, some of them combine a pure geometric metric with a measure of visibility [ZT02, LVJ05]. However, it is not clear how to weight these two factors in order to produce an approximation with a higher degree of visual quality.

In this thesis, a new error metric based on information-theoretic viewpoint selection measures, i.e. viewpoint entropy and mutual information, is proposed to guide the simplification process [CSCF07c, CSCF07b, CCSF07]. With this approach, the hidden interiors of the model are removed just at the beginning.

This metric is also able to preserve the silhouette of the model. Furthermore, a balance in the size of the polygons is achieved. The polygonal simplification algorithm applies the half-edge collapse as a decimation operation.

Viewpoint mutual information represents an improvement with respect to viewpoint entropy where each viewpoint is analyzed individually. However, with mutual information, all viewpoints are analyzed together. This allows mutual information to preserve the silhouette of the model better than viewpoint entropy. Therefore, the visual quality achieved with the mutual information is much higher.

Viewpoint-driven simplification using f-divergences

With viewpoint entropy and mutual information, all the hidden geometry is completely removed in the earliest stages of the simplification process. In fact, this increases the visual quality of the simplified meshes, but it also decreases the geometric fidelity. A high visual quality is a fundamental requirement for some kinds of applications. However, others demand exact geometric tolerances. Several new metrics [CSCF07a, CCSF07] based on f-divergences are proposed in this thesis. These divergences are measures of discrimination or distance between probability distributions. Here we take into account not only the distribution of projected areas of the polygons in the scene or object, but also the actual areas of the polygons. This enables the simplification algorithm to produce approximations that reduce the geometric error and even achieve a higher visual quality.

Viewpoint-driven simplification using mesh saliency

Mesh saliency was defined as a measure of regional importance for graphics meshes. This measure is able to capture what most would classify as visually interesting regions on a mesh. This concept was first introduced by [LVJ05]. In this chapter, a new measure of mesh saliency is proposed. This new mesh saliency, based on the Jensen-Shannon divergence, has been included in the viewpoint-driven simplification method presented here in this thesis [CSCF08]. This approach formulates the mesh saliency in terms of how polygons are seen from a set of viewpoints. The incorporation of mesh saliency into the simplification method allows the regions of the model with small polygons to be preserved better. Collapsing these regions would introduce a small error in our simplification method. However, if they present a large saliency, they will not be as much simplified. It must be considered that this is accomplished at the expense of a greater simplification in other areas of the model.

Viewpoint-driven simplification using the best viewpoints

The viewpoints chosen by the algorithm play a decisive role throughout the simplification process. This is due to the nature of the viewpoint selection measures. However, instead of using fixed camera positions that are determined a priori, in this chapter an algorithm that selects the n best-viewpoint positions is analyzed. This algorithm allows us to use fewer points to accomplish similar results to those obtained with a greater number of viewpoints. This fact makes it possible to reduce the temporal cost considerably.

Comparisons among the different viewpoint-driven methods and a quadric-based algorithm

The different viewpoint-driven simplification methods proposed in this thesis [CSCF07c, CSCF07b, CSCF07a, CCSF07] are compared to a high quality polygonal simplification method based on a pure geometric metric, *quadrics* [GH97]. The quadric error metric is considered to be one of the most important simplification metrics in the literature. This is mainly due to the fact that it is perhaps the best balance between speed, fidelity and robustness currently available for creating LODs. Its authors released the implementation as a software package called *QSlim*, available at
`http://graphics.cs.uiuc.edu/∼garland/software/qslim.html`.
The quadric error metric has been improved in many different approaches. For instance, by adding mesh attributes [GH98, Hop99] and some kind of visibility information [ZT02, LVJ05] and, more recently, by preserving topological restrictions [WHC04, KTiKN05] and physical features [KG03, JTY06, YSZ04].

1.4 Overview

The remainder of this thesis is organized in the following chapters:

- Chapter 2: Previous Work

  Automatic simplification has been a very active field of research in recent decades. In this chapter, the most notable work in the simplification framework is reviewed. Additionally, some basic concepts in Information Theory which will be used later are presented here.

- Chapter 3: Techniques for Computing Projected Areas

  The simplification algorithm proposed in this dissertation is based on viewpoint selection measures. All the viewpoint selection measures introduced throughout all the chapters are computed from the projected areas. In this chapter, several techniques that allow computation of the projected areas of polygons in current graphics hardware are analyzed. In this study, some recent graphics processing units (GPUs) are tested. The fastest technique will be applied later to the simplification algorithm.

- Chapter 4: Viewpoint-driven Simplification

A new simplification algorithm for polygonal meshes is introduced in this chapter. This method uses viewpoint entropy and mutual information to define an error metric that will guide the simplification process. These metrics allow for better preservation of the visual appearance of the model. This is mainly because the hidden interiors of a model are completely removed during the first steps of the algorithm. Mutual information significantly improves the visual quality of the approximations obtained with viewpoint entropy, this is because it is able to preserve the silhouettes better than viewpoint entropy.

- Chapter 5: Viewpoint-driven Simplification using f-divergences

  The viewpoint-driven simplification algorithm is extended with new error metrics based on f-divergences. These metrics consider the actual areas of the polygons as well as the projected areas. This new approach makes it possible to achieve high visual quality of the simplified models without decreasing the geometric fidelity. However, the visual quality of the models obtained with these new error metrics is a bit lower than with mutual information, but the geometric error is quite a lot lower. The hidden interiors of a model are preserved with this new proposal.

- Chapter 6: Viewpoint-driven Simplification using Mesh Saliency

  The simplification method proposed in this thesis is able to produce high quality simplifications. However, some small parts of the model composed of very small polygons, such as the nose and the ears on a face, could be further simplified. By incorporating the mesh saliency concept, it is possible to preserve these regions. In this chapter, the viewpoint-driven method is extended with mesh saliency. The results show that these small yet very important regions are preserved better by using mesh saliency, but other parts of the model could have a slightly worse simplification.

- Chapter 7: Viewpoint-driven Simplification using the Best Viewpoints

  The temporal cost of the viewpoint-driven simplification method is high. This is due to the technique used to compute the different measures based on Information Theory. The cost of these measures is directly dependent on the number of viewpoints. Therefore, instead of choosing several viewpoints distributed uniformly around the model, here we analyze what the simplification would be like if only a few best viewpoints were considered. The results of the experiments show that the visual quality that can be accomplished with a few best points is similar to that achieved with a greater number of uniformly distributed points.

- Chapter 8: Conclusions and Future Work

  The main conclusions and contributions of this thesis are presented in this chapter, as well as some proposals for a future research.

Chapter 2
Previous Work

2.1 Introduction

Research into automatic polygonal simplification has been very active in recent years. The computer graphics literature is replete with excellent simplification methods. In this chapter, the most relevant algorithms in the field of automatic simplification will be covered. Finally, some concepts on Information Theory, which is used throughout this thesis, will be introduced.

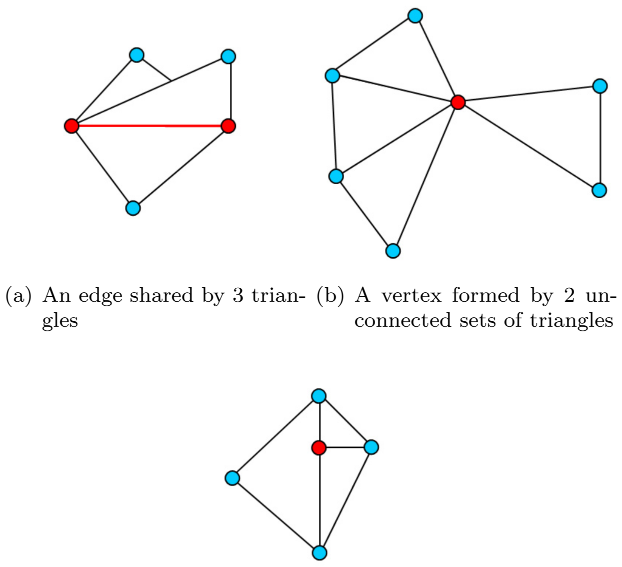
2.2 Polygonal Simplification

The idea of representing objects within a scene at several resolutions is very old and commonly used in computer graphics. For example, Clark [Cla76] described its benefits back in 1976 and since then flight simulators have long used handmade multiresolution models to ensure a constant frame rate. In more recent years, a lot of research has focused on automatic simplification but it is important to mention that today no algorithm excels at simplifying all models. Some work best with terrain data sets, while others fare better with medical and scientific data or with CAD models.

There are several different conceptual approaches to polygonal simplification. Different taxonomies for simplification algorithms will be presented here. These taxonomies describe some ways in which algorithms can differ from or be similar to each other. The literature includes some good surveys [HG97, CMS98, Gar99, Lue01, LRC$^+$04] in which the state of the art is analyzed. Recently, a study of the most important geometric error metrics was carried out in [vKP06].

2.2.1 Topology

Topology refers to the structure of the connected polygonal mesh. The *genus* is the number of holes in the mesh surface. For instance, a sphere and a cube have a genus of zero, while a doughnut and a coffee cup have a genus of one. A mesh is *manifold* if every edge is shared by exactly two triangles. A mesh is *manifold with boundary* if it allows edges with only one triangle.



(a) An edge shared by 3 triangles (b) A vertex formed by 2 unconnected sets of triangles

(c) A T-vertex

Figure 2.1: Examples of non-manifold triangle meshes

Manifold meshes are desirable for many algorithms and applications, but unfortunately in practice many models are not perfectly manifold. They may present many artifacts such as non-manifold edges or vertices (see Figure 2.1). This is a particularly relevant problem in CAD models.

Simplification algorithms can be classified into:

- *Topology-preserving* algorithms. A topology-preserving algorithm maintains manifold connectivity and does not close holes. This constraint limits the simplification. Thus, drastic simplification is not always possible. Many topology-preserving algorithms are simply topology-tolerant, which means that they ignore the regions with non-manifold connectivity and leave them unsimplified.

- *Topology-modifying* algorithms. A topology-modifying algorithm does not necessarily preserve manifold topology and may therefore close holes. But drastic simplification can be accomplished with this kind of algorithms. The price to be paid for this, however, is poor visual quality. Many topology-modifying algorithms are topology-insensitive, and therefore do not take into consideration the initial mesh connectivity.

2.2.2 Mechanism

Four basic polygonal removal strategies, i.e. sampling, adaptive subdivision, decimation and vertex merging, are used by almost every method in the field of polygonal simplification.

- *Sampling* begins by taking geometric samples of the initial model. These samples can be points, manifold surfaces or voxels. Then, these algorithms attempt to create a polygonal simplification that matches the sample data. The greater the number of samples taken, the more accuracy is obtained.

- *Adaptive subdivision* methods find a simple base mesh that can be recursively subdivided to approximate the initial model. This proposal works best when it is easy to find a base model, for instance, in terrain models.

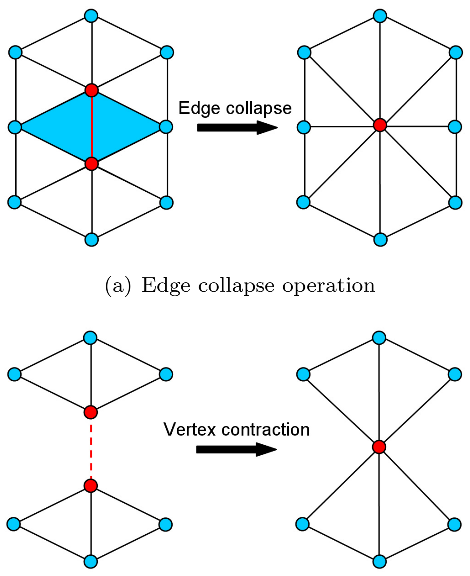
- *Decimation* techniques consist of iteratively removing vertices or faces from the model and retriangulating the resulting hole. These algorithms are easy to code and fast, but may not preserve the topology.

- *Vertex-merging* algorithms work by collapsing two or more vertices of a triangle mesh into a single vertex. Within this approach, there are basically two main operations (see Figure 2.2), namely, *edge collapse*, which means merging two vertices that share an edge, and *vertex contraction*, which means merging pairs of vertices that do not need to be connected. This sort of algorithms can modify topology to enable drastic simplification.

2.2.3 Static, dynamic and view-dependent simplification

Static simplification consists in creating several discrete approximations of an object in a pre-process, each representing a different level of detail. At run-time, the approximation that is most appropriate for representing the object is chosen. This approach has many advantages because current graphics hardware can render these approximations using several acceleration techniques, such as display lists and vertex buffer objects.

Dynamic simplification creates a spectrum of data structure encoding a continuous spectrum of levels of detail. A demanded LOD is extracted from



(a) Edge collapse operation

(b) Vertex contraction operation

Figure 2.2: Vertex-merging operations. The edge collapse operation removes two triangles, which are marked in blue

this structure at runtime. The main advantage of this scheme is the better granularity that is achieved.

View-dependent simplification is an extension of dynamic simplification. Not all parts of an object show the same level of simplification. For instance, nearby and exterior regions may appear at a higher resolution than distant and interior regions. Models representing large objects such as terrain models are the most suitable for this approach. However, View-dependent simplification has a clear drawback, i.e. the additional runtime load needed to choose and extract an appropriate LOD.

2.2.4 Out-of-core simplification

Mesh simplification is frequently applied to very large data sets which are too complex to fit completely into the main memory. In computer science and applications, out-of-core refers to algorithms which process data that is too large to fit into a computer's main memory at one time. *Out-of-core algorithms* have been proposed in order to prevent performance degradation due to virtual memory swapping. The goal here is to design simplification methods which avoid random access to parts of the mesh during the simplification process.

2.3 Some relevant algorithms

Geometric fidelity has been a common issue in the simplification field, and visual fidelity [LT00, ZT02, LVJ05] later emerged as a very important topic. Recently, the polygonal simplification problem has been addressed by some new approaches such as preserving global geometry features [WHC04], retaining physical features [JTY06], considering a very specific class of man-made models [JWRS06] and maintaining the topology better during the simplification process [KTiKN05].

In this section, we will review some of the most important algorithms that have been published in the field of polygonal simplification. Although this list is probably incomplete, it is not our intention to provide an exhaustive catalog, but to select just the most relevant papers.

2.3.1 Triangle mesh decimation

Schroeder et al. [SZL92] proposed one of the first simplification methods for polygonal models. Their algorithm, based on vertex removal, is designed to work on the output of the marching cubes algorithm [LC87] for extracting isosurfaces from volumetric data. Basically, this method operates by making multiple passes over all the vertices in the model. A vertex is removed if the distance to the average plane formed by its incident triangles exceeds a certain threshold. This error estimation is very simple to compute, but generates low-quality approximated models. This *decimation* algorithm is also topology-tolerant, accepting non-manifold vertices but not simplifying around them. Later on, in [Sch97], Schroeder extended this algorithm to make it topology-modifying.

2.3.2 Vertex clustering

The *vertex-merging* algorithm was first proposed by Rossignac and Borrel [RB93]. Their algorithm is topology-insensitive. It begins by assigning an importance to each vertex and then it overlays a 3D grid on the model and collapses all vertices within each cell to the single most important vertex. The resolution of the grid determines the quality of the resulting simplification. Low and Tan [LT97] introduced a different clustering approach, called floating-cell clustering, which leads to more consistent simplification. Since the importance of vertices controls the positioning of clustering cells, unpredictable simplification artifacts are greatly reduced.

Lindstrom [Lin00] also extended this algorithm in a different direction, that is, by performing an out-of-core simplification combined with quadric error metrics [GH97]. Lindstrom and Pascucci [LP01] improved on this approach by removing the requirement for the output model to fit into main memory by using a multi-pass approach. Shaffer and Garland [SG01] proposed a scheme that combines an out-of-core vertex clustering step with an in-core iterative decimation step.

Recently, Kanaya et al. [KTiKN05] have extended the Rossignac and Borrel algorithm in [RB93] to make it topology-preserving.

2.3.3 Multiresolution analysis of arbitrary meshes

Multiresolution analysis of arbitrary meshes, or MRA, is an *adaptive subdivision* algorithm developed by Eck et al. [EDD$^+$95]. This algorithm uses a compact wavelet representation to guide a recursive subdivision process. A base mesh is created by growing Voronoi-like regions across the original triangles of the surface. When these regions cannot grow anymore, the Voronoi sites form a Delaunay-like triangulation which forms the base mesh. But it is not always easy to find a base mesh for general polygonal models. Another disadvantage is that manifold topology is required in the original model and this limits the potential for drastic simplification.

2.3.4 Voxel-based object simplification

Voxel-based object simplification by He et al. [HHK$^+$95] is a *sampling* algorithm that attempts to simplify topology in a gradual controlled way using a signal processing approach. This algorithm samples a volumetric representation of the model, superposing a 3D grid of voxels over the polygonal geometry. It assigns each voxel a value of 1 or 0, depending on whether the sample point lies inside or outside the object. Next, a low-pass filter is applied and the volume is resampled. The result is another volumetric representation of the model with a lower resolution. The main drawback of this algorithm is that it performs poorly on models with sharp edges and squared-off corners. Therefore, it is only useful on mechanical CAD models. Furthermore, this algorithm is not topology-tolerant because deciding whether the sample points lie inside or outside the object requires closed meshes with manifold topology.

2.3.5 Simplification envelopes

The simplification envelopes algorithm by Cohen et al. [CVM$^+$96] creates two copies of the surface of the model: the inner envelope and the outer envelope. These two surfaces are not allowed to self-intersect. Once created, these envelopes can guide the simplification process. After that, this *decimation* algorithm can remove triangles or vertices iteratively and retriangulate the resulting holes. This approach allows topology to be strictly preserved and self-intersections to be avoided, but it limits the capability for drastic simplification.

2.3.6 Appearance-preserving simplification

Appearance-preserving simplification by Cohen et al. [COM98] is an attempt to guide the simplification process by considering not only the surface position but also color and curvature. Their algorithm decouples this information and stores it in normal and texture maps. This *vertex-merging* algorithm uses edge collapses, guided by a texture deviation metric and applies the deviation metric to both the texture and normal maps. If an edge collapse causes color, normals or position to change more than a user-specified distance, it is simply not allowed. These constraints make this approach less suitable for drastic simplification.

2.3.7 Quadric error metrics

This algorithm by Garland and Heckbert [GH97] is perhaps the most important geometric-based algorithm in the literature, probably because it represents the best balance between speed, quality and robustness. This *vertex-merging* algorithm uses vertex contractions. But its major contribution is a new way to represent the error introduced by a vertex contraction operation, called the *quadric error metric*. This metric represents the sum of the squared distances from the vertex to the planes of neighboring triangles, that is, triangles that share the vertex. Therefore, the error introduced by vertex contraction can be quickly derived from the sum of the quadric error of the vertices being merged. This algorithm stores all candidate vertex contractions in a heap. The vertex contraction with the lowest error from the top of the heap is removed and all vertex contractions involving the merged vertices are then updated. This process is repeated until the desired number of triangles is obtained.

Quadric error metrics are fast to compute. Thus, the algorithm has a very low temporal cost. In addition, it does not require manifold topology and is able to close holes because disconnected vertices may merge. This allows the algorithm to perform drastic simplifications. The implementation of this algorithm was released as a software package called QSlim.

Later on, the algorithm was extended to deal with color and texture in [GH98, EM99]. Hoppe probably presented the best extension of quadrics to handle *mesh attributes* [Hop99].

Many incremental algorithms use some kind of a heap data structure with the best removal operation on top. Whenever removal candidates have to be re-evaluated, they are deleted from the heap and re-inserted with their new value. Thus, the complexity of the update-step increases only by $O(\log n)$ for large meshes if the criteria evaluation itself has constant complexity. Wu and Kobbelt [WK02] proposed the probabilistic optimization technique of Multiple-Choice algorithms. With this approach, a global heap data structure is not required and as a result the memory overhead is reduced and the algorithmic structure is simplified. This optimization technique was applied to the QSlim algorithm, thus reducing the temporal cost.

The quadric error metric has been used in a lot of algorithms since Garland and Heckbert first proposed it. Recently, this metric has been extended to any dimension in [GZ05]. In fact, the most recent algorithms use the quadric error metric as the best quality metric to measure the error introduced by a vertex-merging operation.

For instance, it has been used to develop new methods which retain physical features in [JTY06] and preserve global geometry features in [WHC04]. Kho and Garland [KG03] also extended QSlim to allow the user to select some particular regions to be simplified.

Additionally, the QSlim algorithm has been extended to deal with *visual similarity*. Zhang and Turk [ZT02] defined a visibility function between the surfaces of a model and a surrounding sphere of cameras. In order to guide the simplification process, their visibility measure was combined with the quadric error metric. Recently, Lee et al. [LVJ05] introduced the idea of mesh saliency as a measure of regional importance for polygonal meshes and a saliency map was generated and incorporated in the QSlim algorithm.

Finally, this successful metric has also been used to build simplification algorithms within the *out-of-core* approach. For example, Wu and Kobbelt [WK03] proposed a streaming approach to out-of-core mesh decimation-based edge collapse operations in connection with the quadric error metric. The basic idea is to sequentially stream the mesh data and incrementally apply decimation operations on an active working set that is kept in the main memory. Isenburg et al. [ILGS03] introduced mesh processing sequences, which represent a mesh as a fixed interleaved sequence of indexed vertices and triangles. This algorithm can be used in the vertex clustering algorithm [Lin00] and in Wu and Kobbelt's streaming approach. Recently, Vo and Callahan [VC07] proposed a two-step approach for streaming simplification of large tetrahedral meshes. Their algorithm arranges the data on disk in a streaming, I/O-efficient format that allows coherent access to the tetrahedral cells. A quadric-based simplification is sequentially performed on small portions of the in-core mesh. The output is a coherent streaming mesh which facilitates future processing.

2.3.8 Image-driven simplification

Lindstrom and Turk [LT00] were the first to address the problem of *visual similarity* by developing a pure image-based metric. This *vertex-merging* algorithm is based on the edge collapse operation. Basically, their method determines the cost of an edge collapse operation by rendering the model from several viewpoints. The algorithm compares the rendered images to the original ones and adds the mean-square error in luminance across all the pixels in all images. Then all edges are sorted by the total error induced in the images and after that the edge collapse that produces the least error is chosen. Lindstrom et al. used 20 viewpoints in their implementation to compute that error.

The main advantage of their method is that the metric provides a natural way to balance the geometric and shading properties such as color and texture without requiring the user to perform an arbitrary weighting of them. In contrast, its main disadvantage is the high temporal cost. To address this issue, the authors proposed performing two passes. In the first pass, they simplify with a fast geometric-based method and in the second pass, they apply the image-driven simplification.

2.3.9 Progressive meshes

Progressive meshes [Hop96] is a continuous multiresolution model for general polygonal manifold meshes. It is based on the edge collapse operation and its complementary process the vertex split. The vertex split replaces a vertex with an edge thus, creating an additional vertex and two additional triangles. A progressive mesh consists of a simple base mesh, created by a sequence of edge collapse operations and a series of vertex split operations. Applying every vertex split to the base mesh will recapture the original model.

Later on, Hoppe extended progressive meshes to support view-dependent simplification [Hop97] and presented its efficient implementation in [Hop98]. In addition, he described a simplification method. This method uses an energy function to be minimized and a heap structure in which the edge collapses are stored. The energy function considers the position of the vertices, the color, normal and textures, and allows all these attributes to conduct the simplification process.

The view-dependent version of the algorithm simplifies the regions of the mesh that lie out of view by using a view-frustum test. It also simplifies the regions not facing the viewer and finally a screenspace error guarantees that the geometric error is never greater than a user-specified tolerance.

Preserving topology avoids holes but limits drastic simplification and representing non-manifold objects as a progressive mesh might present problems.

2.3.10 Hierarchical dynamic simplification

Hierarchical dynamic simplification or HDS is a *vertex-merging* algorithm developed by Luebke and Erikson [LE97]. The purpose of this algorithm is to provide a view-dependent simplification of arbitrary polygonal scenes. Instead of representing the scene as a set of objects, each at several LODs, in this algorithm the whole model is represented using a vertex tree. This hierarchy of vertex clusters is used to generate a simplified scene. In fact, any algorithm based on vertex merging can be used with HDS.

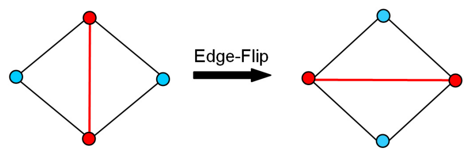
In order to perform a view-dependent simplification, the algorithm uses a screenspace error threshold and a silhouette test. Moreover, it implements a triangle-budget simplification by keeping a heap of vertex clusters. HDS is a topology-tolerant algorithm, suitable for drastic simplification.

2.3.11 Perceptually-driven simplification

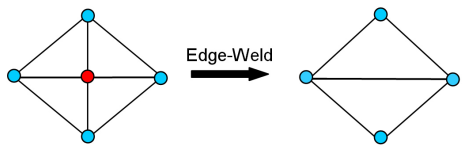
Luebke and Hallen [LH01] proposed the first simplification algorithm based on principles of visual perception. Simplification in this method is guided by perceptual metrics which derive from the contrast sensitivity function, or CSF, which is a simple measure of low-level perceptibility of visual stimuli in humans. This approach was applied to view-dependent polygonal simplification and addresses several interesting problems in regulating level of detail such as silhouette preservation and imperceptible simplification. An optional gaze-direct component uses eye tracking to obtain further simplification by reducing fidelity in the viewer's peripheral vision.

Williams et al. [WLC$^+$03] extended the perceptual simplification framework of Luebke and Hallen to textured and lit models. Moreover, a parametric texture deviation was used to measure distortion more accurately. The data structure was also changed to Multi-Triangulation by [FMP97, FMP98].

2.3.12 Four-Face cluster mesh simplification



(a) Edge swap



(b) Degree 4 vertex removal

Figure 2.3: Local modifications in Four-Face cluster mesh simplification algorithm

Velho [Vel01] proposed a mesh simplification method, called Four-Face cluster (FFC) mesh simplification, that produces a sequence of *Edge-Weld* operations intercalated with *Edge-Flip* operations (see Figure 2.3). Edge-Weld and Edge-Flip are two examples of topological operators based on the Stellar Theory [Ale30]. Edge-Flip swaps an internal edge of a two-face cluster. Edge-Weld removes a valence 4 vertex from a four-face cluster and replaces it with the internal edge of a two-face cluster. Edge-flips are required to change the valence of a vertex to 4. Stellar theory proves that those two operators form a complete set for changing the connectivity of the mesh without modifying its topology. As a mesh quality criteria FFC employs the quadric metric [GH97] and a triangle compactness measure. Furthermore, it produces a hierarchical multiresolution structure, which allows viewpoint-dependent simplification to be performed.

Vieira et al. [VLV$^+$04] enhanced the Four-Face Clusters algorithm by substituting the priority queue by the Multiple-Choice technique [WK02] and introducing the Corner-Table data structure [Ros01] to represent the hierarchy of meshes obtained during the simplification process.

2.4 Information Theory: Concepts

2.4.1 Introduction

Information Theory deals with the transmission, storage and processing of information and is used in fields such as physics, computer science, mathematics, statistics, economics, biology, linguistics, neurology and learning [Bla87, CT91, vdL97]. For instance, it is successfully applied in areas closely related to computer graphics, such as image processing [IVA$^+$96, Stu97, Plu01], pattern analysis and recognition [GDBA94, GDA98, DV00, KB99], scene complexity and discretization [FdABS99b, FdABS99a, Fei02] or computer vision and robot motion [VWMW97, TOS98]. Recently, it has also been used in computer graphics to select the best viewpoints, as well as for scene understanding and virtual exploration [VFSH01, VFSH03, SPFG05] and volume visualization [BS05, TFTN05, JS06, VFSG06].

2.4.2 Entropy

Shannon entropy [Sha48] is the classical measure of information, where *information* is simply *the outcome of a selection among a finite number of possibilities*. Entropy also measures *uncertainty* or *ignorance*.

Thus, the Shannon entropy $H(X)$ of a discrete random variable X with values in the set $S = \{x_1, x_2, ..., x_n\}$ is defined as

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i \tag{2.1}$$

where $n =\mid j \mid$, $p_i = Pr[X = x_i]$ for $i \in \{1, ..., n\}$, the logarithms are taken in base 2 (entropy is expressed in bits), and the convention that $0 \log 0 = 0$ is used, which is justified by continuity. The notation $H(X)$ or $H(p)$ for the entropy can be used interchangeably, where p is the probability distribution $\{p_1, p_2, ..., p_n\}$, also represented by p_i. As $-\log p_i$ represents the information associated with the result x_i, the entropy gives us the average information or uncertainty of a random variable. Information and uncertainty are opposites, since uncertainty is considered before the event and information afterwards. So, information reduces uncertainty. Note that the entropy depends only on the probabilities.

Some other properties [Sha48] of the entropy are

1. $0 \leq H(X) \leq \log n$
    - $H(X) = 0$ if and only if all the probabilities except one are zero and this exception has a value of unity, i.e., when we are certain of the outcome.
    - $H(X) = \log n$ when all the probabilities are equal. This is the most uncertain situation.
2. If we equalize the probabilities, entropy increases, that is, the maximum entropy is reached when all probabilities are equal and their value is $\frac{1}{n}$.

If we consider another random variable Y with probability distribution q_i corresponding to values in the set $S' = \{y_1, y_2, ..., y_m\}$, the *joint entropy* of X and Y is defined as

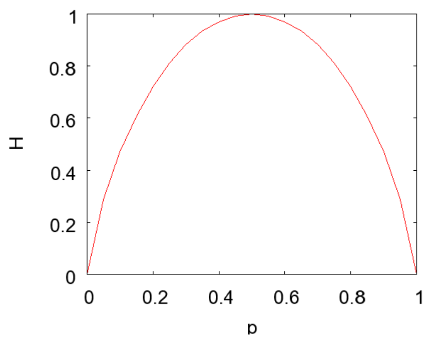


Figure 2.4: The binary entropy function

$$H(X,Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}\log p_{ij} \tag{2.2}$$

where $m = \mid S' \mid$ and $p_{ij} = Pr[X = x_i, Y = y_j]$ is the joint probability. When $n = 2$, the *binary entropy* (see Figure 2.4) is given by

$$H(X) = -p\log -(1-p)\log(1-p) \tag{2.3}$$

where $\mathbf{p} = \{p, 1-p\}$.

Also, the *conditional entropy* is defined as

$$H(X \mid Y) = -\sum_{j=1}^{m}\sum_{i=1}^{n} p_{ij} \log p_{i|j} \tag{2.4}$$

where $p_{i|j} = Pr[X = a_i \mid Y = b_j]$ is the conditional probability.

The Bayes' theorem expresses the relation between the different probabilities:

$$p_{ij} = p_i p_{j|i} = q_i p_{i|j} \tag{2.5}$$

If X and Y are independent, then $p_{ij} = p_i q_j$. The conditional entropy can be thought of in terms of a *channel* whose input is the random variable X and whose output is the random variable Y. $H(X \mid Y)$ corresponds to the uncertainty in the channel input from the receiver's point of view, and vice versa for $H(Y \mid X)$. Note that in general $H(X \mid Y) \neq H(Y \mid X)$.

The following properties are also fulfilled:

1. $H(X, Y) \leq H(X) + H(Y)$
2. $H(X, Y) = H(X) + H(Y \mid X) = H(Y) + H(X \mid Y)$
3. $H(X) \geq H(X \mid Y) \geq 0$

2.4.3 Mutual Information

The *mutual information* between two random variables X and Y is defined as

$$\begin{aligned} I(X,Y) &= H(X) - H(X \mid Y) \\ &= H(Y) - H(Y \mid X) \\ &= -\sum_{i=1}^{n} p_i \log p_i + \sum_{j=1}^{m}\sum_{i=1}^{n} p_{ij} \log p_{i|j} \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij} \log \frac{p_{ij}}{p_i q_j} \end{aligned} \tag{2.6}$$

Mutual information represents the amount of information that one random variable, the output of the channel, gives (or contains) about a second random variable, the input of the channel, and vice versa; that is to say, how much the knowledge of X decreases the uncertainty of Y and vice versa. Therefore, $I(X, Y)$ is a measure of the information shared between X and Y.

Mutual information $I(X, Y)$ has the following properties:

1. $I(X, Y) \geq 0$ with equality if, and only if, X and Y are independent.

2. $I(X,Y) = I(Y,X)$
3. $I(X,Y) = H(X) + H(Y) - H(X,Y)$
4. $I(X,Y) \leq H(X)$

The relationship between all the above measures can be expressed by the Venn diagram, as shown in Figure 2.5.

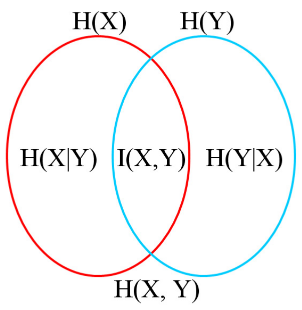


Figure 2.5: Venn diagram of a discrete channel

The *relative entropy* or *Kullback-Leibler distance* between two probability distributions $\mathbf{p} = \{p_i\}$ and $\mathbf{q} = \{q_i\}$, which are defined over the set S, is defined as

$$KL(\mathbf{p} \mid \mathbf{q}) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}, \tag{2.7}$$

where, from continuity, we use the convention that $0 \log 0 = 0$, $p_i \log \frac{p_i}{0} = \infty$ if $p_i > 0$ and $0 \log \frac{0}{0} = 0$.

The relative entropy is “a measure of the inefficiency of assuming that the distribution is $\mathbf{q}$ when the true distribution is $\mathbf{p}$” [CT91].

The relative entropy satisfies the information inequality $KL(\mathbf{p} \mid \mathbf{q}) \geq 0$, with equality only if $\mathbf{p} = \mathbf{q}$. The relative entropy is also called discrimination and is not strictly a distance, since it is not symmetric and does not satisfy the triangle inequality. Moreover, we must emphasize the fact that mutual information can be expressed as

$$I(X,Y) = KL(\{p_{ij}\} \mid \{p_i q_j\}). \tag{2.8}$$

2.5 Summary

In this chapter, the most important work carried out on polygonal simplification and some basic concepts of Information Theory needed in this thesis have been reviewed. As shown, the research in the simplification framework seems to be mature. There are many good methods and algorithms, but most of the work has focused on geometric fidelity. Only a few approaches have recently dealt with the problem of visual fidelity. Although exact geometric tolerances are useful for some kinds of applications, not all kinds of applications need those requirements. For instance, video games, driving simulation and walkthroughs are applications in which the main requirement is visual similarity.

Information Theory is a very powerful tool that can be used to provide measures that can be applied to the field of polygonal simplification. Indeed, Information Theory has been used in many different areas in computer graphics. In recent years, it has been used to develop metrics to select the best viewpoints. In the following chapters, we will review these metrics and see how they can be applied within the simplification framework. Furthermore, these metrics allow for the development of simplification algorithms that consider visual as well as geometric information.

Chapter 3 Techniques for Computing Projected Areas

3.1 Introduction

Recently, several methods have been developed to compute the goodness of a viewpoint. What these methods have in common is the use of the concept of *viewpoint complexity* [BDP99, BPD00, SFR^{+}02, VFSH01, VFSL02, Váz03, RFS00]. The notion of viewpoint complexity is used in several areas of Computer Graphics, such as scene understanding and virtual-world exploration, radiosity and global illumination, image-based modeling and rendering.

In scene understanding and virtual-world exploration, viewpoint entropy which is a measure of viewpoint complexity, is used to automatically calculate suitable positions and trajectories for a camera exploring a virtual world [BDP99, BPD00, Ple03, GAG04, VS03].

In Monte Carlo radiosity and global illumination, viewpoint complexity is used to improve the subdivision of the scene into polygons and the adaptive ray casting [JPS99, Fei02, RFS03].

In image-based modeling, viewpoint entropy is used to compute a minimum optimized set of camera positions [VS02].

Among the metrics that have been introduced for the calculation of viewpoint complexity, viewpoint entropy has been the most fruitful metric to date [VFSH03]. It has recently been embedded in the field of volume visualization to compute the best n-views of a volumetric object. However, the computation of viewpoint entropy can be very costly, especially when a very complex scene and multiple viewpoints have to be evaluated. This is due to the fact that viewpoint entropy is defined from the projected areas of all the polygons in the scene.

In this chapter, we will assess different alternatives for calculating the projected areas of polygons. These alternatives will be tested against several geometric models of increasing complexity. To determine the fastest technique, we will use the calculation times of viewpoint entropy. For this calculation, we will use the facilities of modern hardware cards such as OpenGL histogram and Occlusion query as well as the new symmetric bus PCI Express [WST03, SA06].

The projected areas of polygons are also needed to compute other metrics based on Information Theory used in this thesis. These metrics are proposed to measure the simplification error. To achieve a desired approximation, lots of simplification operations must be performed. Therefore, it is very important to find the fastest technique in order to obtain the projected areas because the temporal cost of the simplification algorithm is directly related to its calculation. This is the reason for the study we carry out in this chapter.

The results of this chapter were first published as a short paper in [CCSF05], which was later extended and published as a full paper in [CSCF06]. Previously, a study on fast rendering techniques for geometry was carried out in [CRC05].

3.2 Viewpoint Entropy

Viewpoint entropy, based on the definition of Shannon entropy, was introduced in [VFSH01, Váz03] as a measure of the information provided by a point of view. The Shannon entropy equation (2.1) is used as a basis to define viewpoint entropy. The relative area of the polygons projected over a sphere of directions centered on the viewpoint is taken as a probability distribution. Thus, given a viewpoint v, the entropy of v is defined by

$$H_v = -\sum_{i=0}^{N_f} \frac{a_i}{a_t} \log \frac{a_i}{a_t}, \tag{3.1}$$

where N_f is the number of polygons in the scene, a_i is the projected area of polygon i over the sphere, a_0 represents the projected area of the background in open scenes, and $a_t = \sum_{i=0}^{N_f} a_i$ is the total area of the sphere. Maximum entropy is obtained when a certain viewpoint can see all the polygons with the same projected area. So, in an open scene the maximum entropy is $\log(N_f + 1)$ and in a closed scene it is equal to $\log N_f$. The best viewpoint is defined as the one that has maximum entropy, i.e., it has the maximum amount of captured information. In molecular visualization, both maximum and minimum entropy views show relevant characteristics of a molecule [VFSL06].

3.3 Techniques for Computing Projected Areas

In order to compute viewpoint entropy, we need to know the number of pixels covered by each visible polygon from a particular camera position. This number will give us the projected area. Next, we analyze several techniques that allow us to compute such areas.

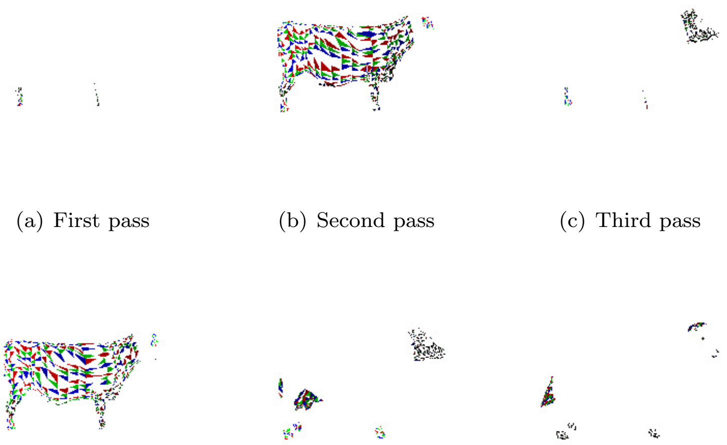
3.3.1 OpenGL Histogram

The OpenGL histogram was first used to compute viewpoint entropy in [VFSL04]. The OpenGL histogram allows us to analyze the color information of an image. Basically, it counts the appearances of a color value of a particular component. However, we can also use it to calculate the area of polygons that are visible from a viewpoint, without reading the buffer. Since version 1.2, OpenGL includes an extension called `glHistogram`. This extension is part of the image processing utilities. The OpenGL histogram is hardware-accelerated, although there are only a few graphics cards that currently support it (for instance, 3DLabs WildCat) and it is often implemented in software.

In order to obtain the area of each visible polygon, a different color is assigned to each polygon. An important limitation is that histograms have a fixed size, normally with 256 different values. This is the most common value in many graphics cards. The `glGetHistogram` command returns a table that counts each color value separated into channels. If the 4 $RGB\alpha$ color channels are used, a 256-item table of 4 integer values will be returned, where each integer is the number of pixels this component has. Thus, if we wish to detect a polygon, this should be codified using a single channel. This gives us a total of 1020 different values, that is, for channel R (1,0,0,0) up to (255,0,0,0), for channel G (0,1,0,0) up to (0,255,0,0), for channel B (0,0,1,0) up to (0,0,255,0) and finally for channel α (0,0,0,1) up to (0,0,0,255). The value (0,0,0,0) is reserved for the background.

Obviously the main drawback of this technique is that several rendering passes are needed for objects with more than 1020 polygons. At each pass, we will obtain the area of 1020 different polygons. Using histograms with a higher number of items and performing an off-screen rendering will increase the number of colors and therefore lower the number of rendering passes that are needed. However, this possibility is beyond the capabilities of the OpenGL specification and is hardware-dependent. It was not possible for us to use a larger-sized histogram in the several graphics cards tested. Figure 3.1 shows an example of the entropy calculation using the OpenGL histogram.

According to some comparative tests that we have run on several GPUs, the entropy calculation with this technique has a time complexity that is linear in the number of polygons and the image resolution, i.e. the number of pixels.



(a) First pass (b) Second pass (c) Third pass

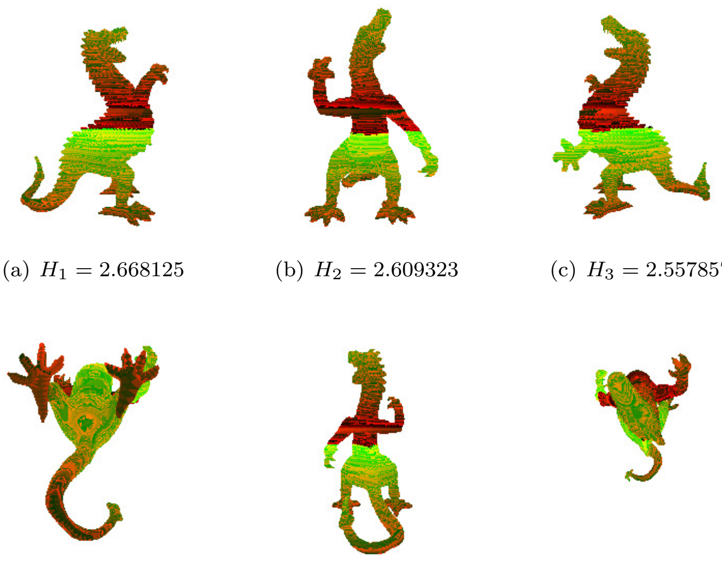
(d) Fourth pass (e) Fifth pass (f) Sixth pass

Figure 3.1: Example of entropy calculation from a viewpoint for the Cow model using the OpenGL Histogram. The total number of triangles for this model is 5804. Therefore, the total number of rendering passes is $5804/1020 \approx 6$. $H_1 = 1.941685$

3.3.2 Hybrid Software and Hardware Histogram

The OpenGL histogram allows us to obtain the area of each visible polygon. However, as pointed out in the previous section, several rendering passes are needed for objects with more than 1020 polygons. Recently, new symmetric buses such as the PCI Express have appeared. In this new bus, the buffer read operation is not as costly as before and it is therefore possible to obtain a histogram without making several rendering passes. The way to accomplish this is very simple. A different color is assigned to each polygon and the whole object is sent for rendering. Next, a buffer read operation is performed, and then this buffer is analyzed pixel by pixel to retrieve data about its color. Using an $RGB\alpha$ color encoding with a byte value for each channel, up to 256^4 polygons can be calculated with only one single rendering pass. Figure 3.2 shows an example of the entropy calculation using this method.

With this technique, the time complexity for the calculation of viewpoint entropy is linear in the number of polygons and the number of pixels. Note that most of the work is performed by the CPU. Therefore, faster processors will improve the performance of this technique.



(a) $H_1 = 2.668125$ (b) $H_2 = 2.609323$ (c) $H_3 = 2.557857$

(d) $H_4 = 2.387822$ (e) $H_5 = 1.964738$ (f) $H_6 = 1.224885$

Figure 3.2: Entropies from 6 different viewpoints for the Dragon model obtained with the Hybrid Software and Hardware Histogram. The maximal entropy viewpoint corresponds to (a)

3.3.3 Occlusion Query

This OpenGL extension is normally used to identify which objects in the scene are hidden by others, and therefore should not be sent to be rendered. In fact, what we do is to render just the bounding box of an object and, if it is not visible, the object is not sent for rendering. However, it can also be used to compute the area of the polygons that are visible from a particular camera position.

The OpenGL `ARB_occlusion_query extension` returns the number of visible pixels. In order to compute the area of each visible polygon of an object with this technique we will proceed as follows. First, the whole object is sent for rendering and the depth buffer is initialized. Second, each polygon is sent for rendering independently. With this procedure it is necessary to make $n + 1$ rendering passes, n being the number of polygons in an object. It must be mentioned that the whole geometry is only rendered in the first pass. In the following passes, one single polygon is rendered. However, a high number of renderings steps can significantly penalize this technique. In order to improve the results this extension can be used asynchronously, unlike its predecessor, `HP_occlusion_query`. That is, it does not use a ‘stop-and-wait’ execution model for multiple queries. This allows us to issue many occlusion queries before asking for the result of any one of them. But we must be careful with this feature because, as we mentioned above, this extension was not designed to deal with thousands of multiple queries. Thus, we may have some limitations depending on the graphics card.

The running time for this technique is linear in the number of polygons. And, as we have confirmed in tests run on several GPUs using different resolutions, image resolution does not increase the time complexity.

3.4 Comparison

We calculated viewpoint entropy from 6 camera positions, which were regularly distributed over a sphere that covers the object, using the different techniques described above. In order to compare them, we measured the time needed to compute the entropy from those cameras. As test models, we used several models of different complexities (see Figure 3.3). All models were rendered in a 256×256 pixel resolution using OpenGL vertex arrays. Two different PC configurations were used: a Xeon 2.4 GHz with 1GB RAM with an ATI X800XT 256MB and a Pentium IV 3.0 GHz with 1GB RAM with an NVIDIA GeForce 6800GT 256MB. We must emphasize that of the two GPUs that were analyzed, only the NVIDIA card supports the OpenGL histogram.

Model	**Vertices**	**Triangles**	**Rendering Passes**	**OpenGL Histogram(ms)**	
				GeForce 6800 GT	**Radeon X800 XT**
Teddy	1598	3192	4	2811.45	-
Cow	2904	5804	6	4227.28	-
Teapot	3644	6320	7	4927.65	-
Octopus	4242	8468	9	6339.67	-
Unicycle	6973	13 810	14	9886.66	-
Roman	10 473	20 904	21	14 888.38	-
Sphere	15 314	30 624	31	22 136.10	-
Bunny	34 834	69 451	69	50 445.86	-
Dragon	54 296	108 588	107	80 029.94	-

Table 3.1: Results for the calculation of viewpoint entropy with the OpenGL Histogram. The times are measured in milliseconds

Table 3.1 shows the results obtained with the OpenGL histogram. These times are too high to allow an interactive calculation, even for objects with a low complexity. This is fundamentally due to the several rendering passes of the whole object that we make when using objects consisting of several thousand polygons. The main cost component is the OpenGL histogram operation.

Table 3.2 shows the results obtained with the hybrid software and hardware histogram. As shown in this table, the times are quite low even if the complexity is increased, mainly because we make one single rendering pass and the buffer read operation has a very low cost.

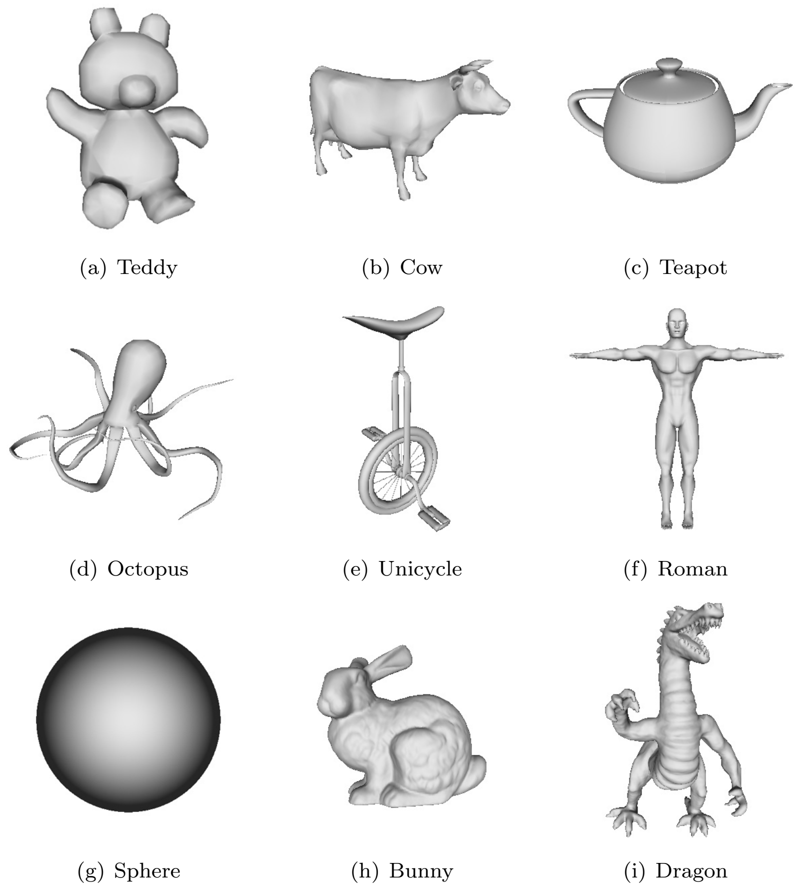


Figure 3.3: Models used in our experiments

Model	**Vertices**	**Triangles**	**Rendering Passes**	**SW+HW Histogram(ms)**	
				GeForce 6800 GT	**Radeon X800 XT**
Teddy	1598	3192	1	11.66	16.62
Cow	2904	5804	1	13.36	19.10
Teapot	3644	6320	1	14.84	19.37
Octopus	4242	8468	1	17.28	20.69
Unicycle	6973	13 810	1	18.53	23.24
Roman	10 473	20 904	1	24.12	29.85
Sphere	15 314	30 624	1	36.65	38.09
Bunny	34 834	69 451	1	57.91	67.04
Dragon	54 296	108 588	1	79.35	88.75

Table 3.2: Results for the calculation of viewpoint entropy with the Hybrid Software and Hardware Histogram. The times are measured in milliseconds

Model	Vertices	Triangles	Rendering Passes	Occlusion Query(ms) GeForce 6800 GT	Radeon X800 XT
Teddy	1598	3192	3193	26.88	25.19
Cow	2904	5804	5805	47.49	44.41
Teapot	3644	6320	6321	50.88	48.31
Octopus	4242	8468	8469	67.17	64.48
Unicycle	6973	13 810	13 811	109.88	100.78
Roman	10 473	20 904	20 905	162.75	151.31
Sphere	15 314	30 624	30 625	238.09	221.42
Bunny	34 834	69 451	69 452	553.36	460.74
Dragon	54 296	108 588	108 589	829.75	665.33

Table 3.3: Results for the calculation of viewpoint entropy with the Occlusion Query. The times are measured in milliseconds

Table 3.3 shows the results obtained with the Occlusion Query technique. As can be observed, the measured times increase proportionally in relation to the complexity of the model being analyzed. In the same way as the previous technique, the ratio remains unchanged here because the number of rendering passes is proportional to the number of triangles. A complete rendering of the object is only performed at the first pass.

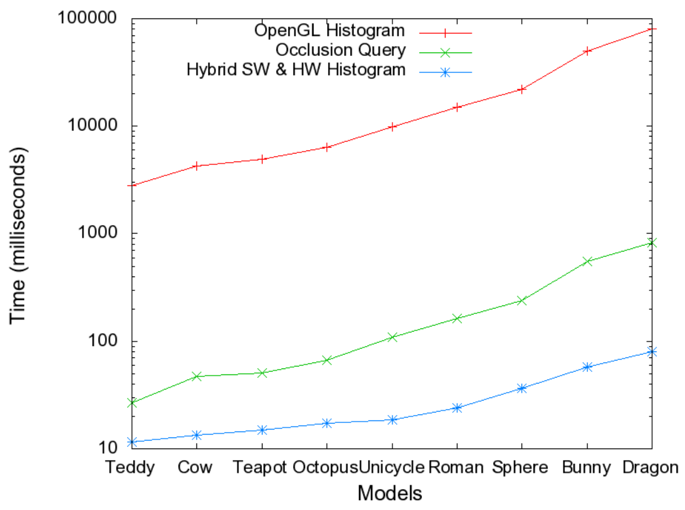


Figure 3.4: Comparison of results obtained with the different analyzed techniques

Finally, as a summary, Figure 3.4 shows a comparison of the performance of the different techniques. These results were obtained with the NVIDIA card described previously. We used an NVIDIA card because it fully supports all the techniques. In all cases, if we examine the ratio of temporal costs among the techniques with the ATI card, it can be seen that they are practically the same as with the NVIDIA card.

These results clearly show that by using the hybrid software and hardware histogram technique the entropy can be calculated in real time even for complex objects (100 000 triangles), because times increase very slowly as complexity goes up. The next best technique is the Occlusion Query. Note that its cost grows as the object complexity increases, and is unfeasible for complex objects in real time. Lastly, the OpenGL histogram technique is worse than the other two. This technique is useless for real-time application, unless we use objects with a low degree of complexity (1000 triangles).

3.5 Conclusions

In this chapter, we have studied several hardware-assisted techniques for computing the projected area of polygons in an efficient way. These techniques were applied to the calculation of viewpoint entropy. Among the different techniques analyzed, viewpoint entropy computed with the hybrid software and hardware histogram method accomplishes the best performance, followed by the occlusion query-based technique. By using the hybrid software and hardware histogram technique the entropy calculation can be performed practically in real time even for complex objects, while the occlusion query technique allows us to obtain only interactivity.

We must take into account that the performance of the hybrid software and hardware histogram technique depends on the analysis of pixels performed by the CPU and the read operation of the PCI Express bus. We also carried out some tests using higher resolutions, for example 960×960, and we observed that the times for the occlusion query are constant, even at higher resolutions the hybrid software and hardware histogram technique gives better results than occlusion queries. The proportion is not as high as before, but it is still significantly better. For our purposes, we believe that the resolution used in our experiments (256×256) is enough to obtain accurate results.

CHAPTER 4

Viewpoint-Driven Simplification

4.1 Introduction

Polygonal models currently dominate interactive computer graphics. Polygons are the simplest primitive and allow for regular rendering algorithms that fit the hardware well. Unfortunately, the complexity of these models seems to grow faster than the ability of graphics hardware to render them interactively. Polygonal simplification offers one solution. A common use of polygonal simplification is to generate *levels of detail* (LODs) of the objects in the scene. Levels of detail are used to create *multiresolution model representations* [Gar99] which allow the surface to adapt at run-time.

Most common polygonal simplification methods use a technique based on a geometric distance as a measure of the quality between an original mesh and the one obtained from simplification. With these methods we can achieve meshes that are very similar to the original. One of the most important advantages of geometry-oriented methods is their low temporal cost. This fact makes them suitable for scanned models, since these models are composed of thousands or even millions of polygons. In addition, geometric methods are very useful in applications that require exact geometric tolerances with regard to the original model. Examples of such applications include collision detection and path planning for part insertion and removal.

In contrast, image-based simplification methods carry out a simplification that is guided by differences between images more than by geometric distances. In other words, their goal is to create simplified meshes that appear similar according to visual criteria. These methods have a high temporal cost compared to geometric ones. The applications that can benefit from image-based methods are those in which the main requirement is visual similarity. Examples of such applications are video games, driving and walkthroughs.

In this chapter, a viewpoint-driven simplification method is introduced. This method defines a new simplification error metric which is based on viewpoint selection measures. Two information-theoretic viewpoint selection measures are used within this simplification scheme: *viewpoint entropy* [VFSH01], which is a measure of the geometric information of a scene or object seen from a certain point of view, and *viewpoint mutual information* [VFSG06, FSG06], which is a measure of the correlation between a viewpoint and the polygons in an object or scene. It is important to remark that, in principle, any other viewpoint selection measure could also be applied to this approach. As a decimation criterion, the simplification algorithm uses the *half-edge collapse* [HDD$^+$93, KCS98]. The error introduced by a decimation operation is calculated from the variation in viewpoint selection measures. Experimental results show that the method presented here yields better visual performance than QSlim-based simplifications [GH97]. Furthermore, it also offers very good results even at early stages of simplification, where it achieves a higher simplification in hidden interiors.

This chapter is organized as follows. In Section 4.2, the recent work carried out in polygonal simplification is reviewed. In Section 4.3, some viewpoint selection measures are introduced. In Section 4.4, the simplification error metric to measure the cost of an edge collapse is defined. In Section 4.5, the simplification algorithm is described. In Section 4.6, the results of the experiments are shown and finally, in Section 4.7, conclusions are presented.

A preliminary version of the polygonal simplification presented here was first published as a full paper in [CSCF07c].

4.2 Related work

The most important improvement in geometry-oriented simplification methods in recent years was the incorporation of mesh attributes such as color, normals and textures. For example, Hoppe extended his initial work [Hop96] by proposing a new quadric metric that includes colors and texture coordinates [Hop99], and the QSlim algorithm [GH97] was also extended with those attributes [GH98]. Cohen et al. [COM98] developed an algorithm based on edge collapses that samples the vertex position, normal and color attributes of the original mesh and then converts them into normal and texture maps. This algorithm is based on a texture deviation metric. More recently, a general method to incorporate texture information for edge collapse-based simplification algorithms has been proposed in [GCC07].

Lindstrom and Turk [LT00] were the first to address the problem of *visual similarity* by developing a purely image-based metric. Basically, their method determines the cost of an edge collapse operation by rendering the model from several viewpoints. The algorithm compares the rendered images to the original ones and adds the mean-square error in luminance across all the pixels of all the images. Then, all edges are sorted by the total error induced in the images and after that the edge collapse that produces the least error is chosen. Lindstrom et al. used 20 viewpoints in their implementation to compute that error. The main advantage of this method is that the metric provides a natural way to balance the geometric and shading properties without requiring the user to perform an arbitrary weighting of them. On the other hand, its main disadvantage is the high temporal cost.

Karni and Gotsman [KG00] proposed a metric to capture the visual difference between two approximations, which consisted of the average of the norm of the geometric distance between models and the norm of the Laplacian difference. By introducing the Laplacian component, some visual properties appreciated by the human eye, such as smoothness, are captured better.

Luebke and Hallen [LH01] presented a method to perform a view-dependent polygonal simplification using perceptual metrics. These metrics derive from a measure of low-level perceptibility of visual stimuli in humans. Later on, Williams et al. [WLC$^+$03] extended this work for lit and textured meshes.

Zhang and Turk [ZT02] proposed a new algorithm that takes visibility into account. This approach defines a visibility function between the surfaces of a model and a surrounding sphere of cameras. The number of cameras increases both accuracy and calculation time. Zhang et al. used up to 258 cameras. In order to guide the simplification process, they combined their visibility measure with the quadric error metric introduced by Garland and Heckbert [GH97].

Recently, Lee et al. [LVJ05] introduced the idea of mesh saliency as a measure of regional importance for graphics meshes. This measure was incorporated into mesh simplification. Briefly, this approach consists of generating a saliency map, and then simplifying by using this map in the QSlim algorithm [ZT02]. The new edge collapse cost is that of the quadric multiplied by the saliency of this edge.

4.3 Information-Theoretic Viewpoint Selection Measures

Information-theoretic-based viewpoint selection metrics have been successfully applied in different areas of computer graphics, such as scene understanding, virtual exploration [VFSH01, Váz03] and volume visualization [BS05, TFTN05, JS06]. In this section, we review *viewpoint entropy* [VFSH01, Váz03] which has been used to compute the best viewpoints in a scene, and *viewpoint mutual information* which has been introduced to select the best views in volume rendering [VFSG06] and for polygonal meshes [FSG06, FSG07].

4.3.1 Viewpoint Entropy

From (2.1), viewpoint entropy [VFSH01, Váz03] has been defined from the relative area of the polygons projected over the sphere of directions centered at viewpoint v. Thus, *viewpoint entropy* was defined by

$$H_v = -\sum_{i=0}^{N_f} \frac{a_i}{a_t} \log \frac{a_i}{a_t}, \tag{4.1}$$

where N_f is the number of polygons of the scene, a_i is the area of the polygon i projected over the sphere, a_0 represents the projected area of the background in open scenes, and $a_t = \sum_{i=0}^{N_f} a_i$ is the total area of the sphere. Maximum entropy is obtained when a certain viewpoint can see all the polygons with the same projected area. The best viewpoint is defined as the one that has maximum entropy.

4.3.2 Viewpoint Mutual Information

In [VFSG06, FSG06], *viewpoint mutual information* was introduced to select the best views. An *information channel* $V \rightarrow O$, called a *viewpoint information channel*, between the random variables V and O was defined. This channel represents, respectively, a set of viewpoints and the set of polygons of an object. Viewpoints will be indexed by v and polygons by o. The marginal probability distribution of V is given by $p(v) = \frac{1}{N_v}$, where N_v is the number of viewpoints. That is, the same probability is assigned to each viewpoint, although other distributions could be used. The conditional probability $p(o \mid v) = \frac{a_o}{a_t}$ is defined by the normalized projected area of polygon o over the sphere of directions centered at viewpoint v. Conditional probabilities fulfill $\sum_{o \in \mathcal{O}} p(o \mid v) = 1$. Note that with this notation viewpoint entropy (4.1) can be rewritten as $H_v = -\sum_{o \in \mathcal{O}} p(o \mid v) \log p(o \mid v)$. Finally, the marginal probability distribution of O is given by $p(o) = \sum_{v \in \mathcal{V}} p(v) p(o \mid v) = \frac{1}{N_v} \sum_{v \in \mathcal{V}} p(o \mid v)$, which represents the average projected area of polygon o, i.e., the probability of a polygon o to be hit (seen) by a random ray cast from the viewpoint sphere.

From this channel, the *mutual information* (2.6) between V and O, which expresses the degree of *dependence* or *correlation* between a set of viewpoints and the polygons of the object, is given by

$$\begin{aligned} I(V,O) &= \sum_{v \in \mathcal{V}} p(v) \sum_{o \in \mathcal{O}} p(o \mid v) \log \frac{p(o \mid v)}{p(o)} \\ &= \frac{1}{N_v} \sum_{v \in \mathcal{V}} I(v, O), \end{aligned} \tag{4.2}$$

where

$$I(v,O) = \sum_{o \in \mathcal{O}} p(o \mid v) \log \frac{p(o \mid v)}{p(o)}, \tag{4.3}$$

called *viewpoint mutual information* (VMI), represents the degree of correlation between the viewpoint v and the set of polygons O, and it is a measure of the *quality* of viewpoint v. *Quality* is considered here equivalent to *representativeness*. The best viewpoint is defined as the one that has *minimum* VMI. High values of the measure mean a high degree of dependence between the viewpoint v and the object, indicating a highly *coupled* view. On the other hand, low values correspond to the most *representative* or *relevant* views, showing the maximum possible number of polygons in a balanced way.

Note that $I(v, O) = KL(p(O \mid v) \mid p(O))$, where $p(O \mid v)$ is the conditional probability distribution between v and the object and $p(O)$ is the marginal probability distribution of O, which in our case corresponds to the distribution of the average of projected areas. It is worth observing that $p(O)$ plays the role of the *target* distribution in the KL distance and also the role of the *optimal* distribution since the objective is that $p(O \mid v)$ becomes similar to $p(O)$ to obtain the best views. On the other hand, this role agrees with intuition since $p(O)$ is the average visibility of polygon o over all viewpoints, i.e., the *mixed distribution* of all views, and we can think of $p(O)$ as representing, with a single distribution, the knowledge about the scene.

4.4 Viewpoint-Based Error Metric

In this section, a new error metric based on viewpoint selection measures is presented. This metric can be used to evaluate the cost of a decimation operation. The edge collapse is chosen as the decimation operation, although any other simplification operation could be performed such as removing a vertex, replacing a cluster of vertices by a single one and contracting an edge.

Viewpoint selection measures express the accessible information about an object from a particular viewpoint. Given a particular viewpoint, we can consider the following: if the simplification is produced near the silhouette, probably it will change the shape of the object. Therefore, if the goal is to preserve the visual appearance of the model then we should try to reduce this change. In addition, in order to preserve the global appearance of the model, several equidistant viewpoints surrounding the model are required, so that the whole model will be fully covered. This distribution guaranties a uniform simplification.

Taking into account the facts mentioned above, the variation of a viewpoint selection measure for each viewpoint can provide us with an error metric to guide the simplification process. Thus the *simplification error deviation* for an edge collapse e from all viewpoints V is defined by:

$$C_e = \sum_{v \in \mathcal{V}} \mid I_v - I'_v \mid, \qquad (4.4)$$

where I_v represents the viewpoint selection measure before the edge collapse

e and I'_v afterwards.

We choose two information-theoretic viewpoint selection measures to test our simplification method, namely, *viewpoint entropy* (H_v) and *viewpoint mutual information* (VMI). Viewpoint entropy measures the degree of uniformity of the projected area distribution at viewpoint v and viewpoint mutual information gives us the degree of dependence between the viewpoint v and the set of polygons. Therefore, VMI can be interpreted as the average viewpoint quality. *Quality* is considered here equivalent to *representativeness*. Viewpoint mutual information has an important feature, that is, it can be used as a shape descriptor for object recognition [RFS05], which is suitable for capturing the shape variation. Viewpoint entropy tends to infinity when polygons are infinitely refined. This makes this measure very sensitive to the discretization of the object, in general, inappropriate for evaluating the quality of a viewpoint. Viola et al. [VFSG06] show that the main advantage of VMI over H_v is its robustness to deal with any type of discretization or resolution of the volumetric data. The same advantage can be observed for the polygonal data. Thus, while a highly refined mesh will attract the attention of H_v, VMI will be almost insensitive to changes in the mesh resolution. In general, if we compare both measures for different discretizations, mutual information will give similar results in viewpoint quality and H_v will show an erratic behavior. In conclusion, when a mesh is infinitely refined, VMI converges to a finite value while H_v diverges. As a consequence, VMI is more robust than viewpoint entropy when the object mesh is changed.

Both viewpoint entropy and mutual information are based on the distribution of areas of polygons seen from a viewpoint. The area of the background is also included as the polygon 0. These facts allow viewpoint entropy and mutual information to preserve the silhouette better. But perhaps the main implication of considering the projected areas is that the hidden geometry will be initially removed, because if a polygon is not seen from any point of view, it will not introduce any error.

In the previous chapter, several techniques for computing those projected areas were analyzed. More specifically, the OpenGL histogram, the hybrid SW-HW histogram and the occlusion query were studied. The fastest technique, using today’s hardware, was found to be the hybrid SW-HW histogram. This technique takes advantage of the PCI Express bus symmetry. In brief, a different color is assigned to each polygon and the whole object is sent for rendering. Next, a buffer read operation is performed, and then this buffer is analyzed pixel by pixel to retrieve data about its color. Using $RGB\alpha$ color encoding with a byte value for each channel, up to 256^4 polygons can be calculated with only one single rendering pass. We used this technique during the simplification process.

Figure 4.1 shows the original Test model and how the viewpoints are distributed around it. These viewpoints are associated with the vertices of the Cube in which the object is inscribed. Figure 4.2 shows the VMI for the origi-

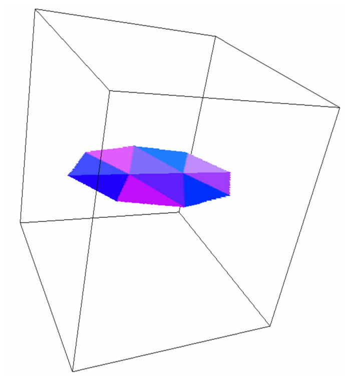
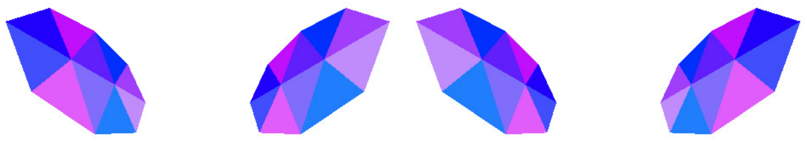


Figure 4.1: Example of 8 camera positions surrounding the Test Model. These camera positions correspond to the 8 vertices of the cube. This distribution allows the object to be completely covered because all the viewpoints are equidistant from each other

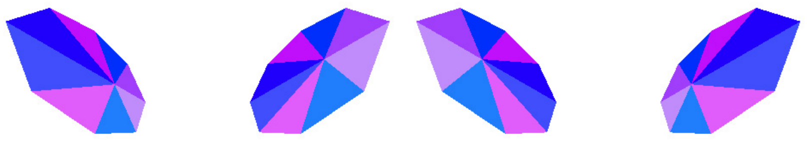
nal Test model using the 8 viewpoints shown in Figure 4.1. As can be seen, the different viewpoints have the same VMI. This is because all the object is seen from every viewpoint and each viewpoint sees the same as any other. Note that this is a very special situation because the object is quite simple and regular. Normally, in more complex models every viewpoint will have a different VMI.

Figures 4.3 and 4.4 illustrate how VMI can be employed to conduct the simplification. Figure 4.3 shows the Test model after performing the best edge collapse e and Figure 4.4 after performing the worst edge collapse e'. The best edge collapse belongs to the lowest simplification error C_e (4.4) and the worst edge collapse belongs to the highest. As can be observed all the VMI values for every viewpoint decreased after an edge collapse (see for instance Figure 4.2(a) compared to Figure 4.3(a) or 4.4(a)). This is because the visible area did not increase in both cases and also the complexity is always reduced during the simplification process. But in a more general case, it is possible that after an edge collapse some previously hidden parts of the mesh may now appear, thus increasing the visible area. If we pay attention to Figure 4.2(b) and compare this same viewpoint after the best edge collapse (see Figure 4.3(b)), it can be appreciated that although the number of triangles is reduced (T=8), the visible area remains the same. The simplification error for this viewpoint using VMI is $C_e = 0.004097 - 0.003651 = 0.000446$. If we analyze the same viewpoint in the worst edge collapse operation (see Figure 4.4(b)), it can be seen that although the number of triangles is less reduced (T=9), the total visible area is somewhat decreased. The simplification error for this viewpoint is $C_{e'} = 0.004097 - 0.003372 = 0.000725$, which is higher than the error committed in the best edge collapse.



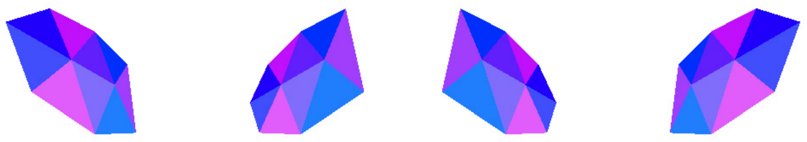
(a) (b) (c) (d)

Figure 4.2: Original Test model. T=10. $I(v, O) = 0.004097$ where $v = \{1, .., 8\}$. Only 4 viewpoints are shown because the rest are symmetric



(a) I(1,O)=0.003830 (b) I(2,O)=0.003651 (c) I(3,O)=0.003651 (d) I(4,O)=0.003830

Figure 4.3: Test model after performing the best edge collapse e using VMI. T=8. $C_e = 0.002573$



(a) I(1,O)=0.003238 (b) I(2,O)=0.003372 (c) I(3,O)=0.003372 (d) I(4,O)=0.003238

Figure 4.4: Test model after performing the worst edge collapse e' using VMI. T=9. $C_{e'} = 0.006228$

4.5 Simplification Algorithm

The simplification process, like many other simplification algorithms, is based on the edge collapse operation. However, we use the *half-edge collapse* operation. According to this, the remaining vertex for an edge collapse $e(u, v)$ is vertex u or v (see Figure 4.5). By using half-edge collapses it is possible to reuse the simplification process in order to generate multiresolution models. These models can use the current hardware in a more efficient way because no new vertices are added to the original model. Furthermore, the half-edge representation is useful for progressive transmission. The main disadvantage is a slight loss of quality of the final mesh, although the complexity of the simplification algorithm is reduced because we do not have to compute the position of the new vertex v' resulting from the edge collapse. In any case, the general edge collapse operation can be applied to our algorithm. However, a strategy is required to compute the position of the resulting vertex.

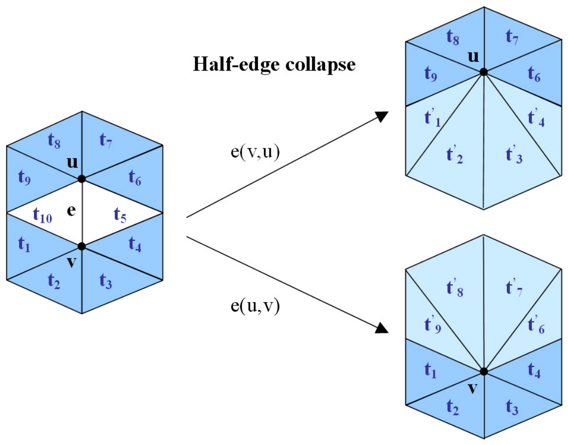


Figure 4.5: The half-edge collapse operation. In this example edge e is collapsed into vertex u (see $e(v, u)$), but it is also collapsed into v (see $e(u, v)$). Triangles t_{10} and t_5 are removed

Brute force selection of edges can introduce mesh inconsistencies. In order to avoid these artifacts, we only take into account the edges which have at most two adjacent polygons, that is 2-manifold edges. And we also consider boundary edges, i.e. edges which have one single adjacent polygon.

The best half-edge collapse is the decimation operation chosen in our algorithm. Note that the cost of collapsing vertex u to v may be different to the cost of collapsing v to u. In our strategy in order to determine the best orientation of an edge collapse, we would have to render the two possibilities and compute that error. However, this would considerably increase the number of renderings and consequently the number of framebuffer readings. Therefore the

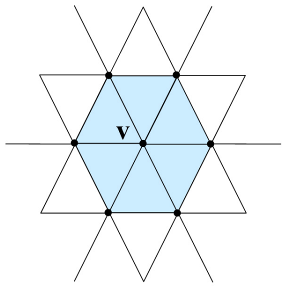


Figure 4.6: Edges adjacent to vertices adjacent to vertex v

temporal cost would be penalized. To avoid this, we used the approach developed by Melax [Mel98], which takes into account polygon normals. Within this approach, the two orientations $e(u, v)$ and $e(v, u)$ are calculated and finally the orientation that produces a minor change in the curvature of the local region around the edge collapse is applied. Hence the simplification error deviation is only computed for that orientation.

4.5.1 Evaluation of the Edge Collapse Error

In the previous section, the error introduced by an edge collapse was defined as the sum of the differences before simplifying and after simplifying.

To speed up its calculation, we can make use of the fact that both viewpoint entropy and mutual information can be iteratively calculated. Viewpoint entropy and mutual information are computed from the projected areas and the total projected area. The background is considered to be another polygon, and thus the total projected area is always the image resolution. Moreover, only a few polygons change after an edge collapse. Therefore viewpoint entropy and mutual information can be computed at the beginning for the entire object and then their initial values can be successively updated. In our implementation we have exploited this feature.

Our algorithm maintains a heap of edge collapses, sorted by the simplification error cost. In fact, it is an iterative method, so the edge collapse operation is applied until the desired approximation is obtained. At each operation, the edge collapse e that has the least deviation C_e (4.4) is chosen. However, it is important to determine some parameters, since the quality of the results may change. We performed measurements with 20 regularly distributed viewpoints and rendered 256×256 resolution images. More viewpoints can increase quality, but also significantly raise the temporal cost.

In [CRC05] different hardware techniques were analyzed for geometric visualization using standard OpenGL running on current GPUs, the results showing that the *vertex buffer objects* technique is the best suited to dynamic geometry.

So this technique was used to render the images. For the off-screen rendering the OpenGL *framebuffer object* extension was employed.

At each iteration the edge cost has to be evaluated for the entire set of remaining edges. An edge collapse in our algorithm could, in principle, affect the cost of any remaining edge. But this does not always happen to each edge. At each step we only choose a small group of edges that are affected by an edge collapse and then the cost is recalculated for these edges. These edges are the ones that are adjacent to the vertices adjacent to the vertex v resulting from a half-edge collapse (see Figure 4.6). In our experiments, if we consider the whole set of edges of the model, the temporal cost is increased about 20 times, but we achieve results that are not significantly better.

In order to avoid performing unnecessary edge collapse calculations, after applying an edge collapse, each edge that should be recalculated is simply marked as dirty. Such edges are really recalculated only when they reach the top of the heap. If the edge extracted from the heap is dirty, it is simply discarded. Then, its cost is recomputed and inserted into the heap again. This heuristic was introduced in [CMO97]. In Figure 4.7 we show a summary of the simplification algorithm.

```
// Compute initial viewpoint selection measure for mesh $M$
Compute $I_v$, where $v = \{1, .., n\}$

// Build initial heap of edge collapses
for ($e \in M$)
    Choose the best orientation of $e$
    Perform collapse $e$
    Compute $I_v$, where $v = \{1, .., n\}$
    Compute collapse cost $C_e = \sum_{v=1}^{n} | I_v - I'_v |$
    Insert the duple $(e, C_e)$ in heap $h$
    Undo collapse $e$
end for

// Update mesh $M$
while (heap $h$ not empty)
    Delete from heap $h$ edge $e$ with lowest $C_e$
    Perform collapse $e$
    Recalculate cost for the neighborhood of $e$
    and update their location in heap $h$
end while
```

Figure 4.7: Pseudo-code of the viewpoint-driven simplification algorithm

4.5.2 Updating Projected Areas

As mentioned in Section 4.4, the projected area of every polygon from a viewpoint is needed in order to compute viewpoint entropy and mutual information. The bottleneck resides in the pixel-to-pixel analysis performed by the histogram [CSCF06] to obtain those areas, due to the memory transfer cost. Therefore, this overload can be reduced if, instead of analyzing the whole image, the area of reading is restricted to a small window that only includes the polygons surrounding the edge collapse (see Figure 4.8).

To obtain this window, first the bounding box of the polygons surrounding the edge collapse is determined and then projected onto the screen. This method allows the temporal cost of the algorithm to be reduced by about 10 times, but it may lead to some slight loss of quality. This is mainly due to the fact that after an edge collapse some hidden polygons might appear and it was not possible to measure their contribution to the formula.

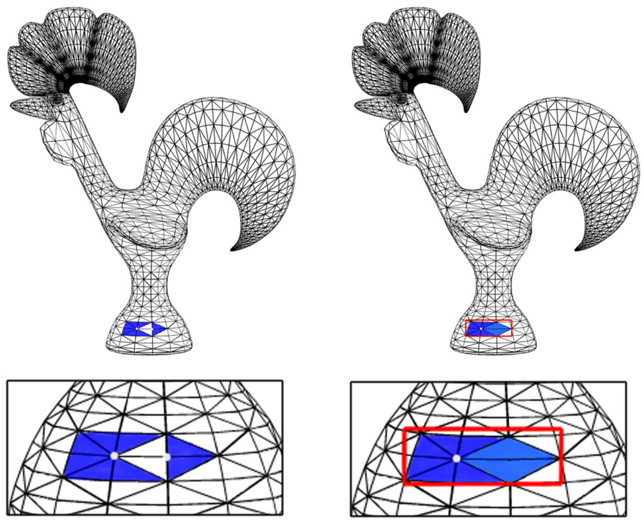


Figure 4.8: Image (a) shows the Galo model: the triangles surrounding the edge collapse are marked in blue. The triangles that are going to be removed are marked in white. Image (b) shows in red the window used to obtain the new projected areas for blue triangles. The modified triangles are in a lighter blue

4.5.3 Temporal Cost Analysis

The total temporal cost of our algorithm depends on the number of viewpoints V, the number of pixels (image resolution) P and the number of triangles in the mesh T. Calculating the viewpoint selection measure (VMI or H_v) implies a cost of $O(PV)$. This is due to the analysis performed in the framebuffer in order to obtain the projected areas, which form the probability distributions. The cost of inserting or updating an item in a priority queue is $O(\log T)$. Finally, If we consider the worst case, that is, all triangles are removed, the total cost belongs to $O(PVT \log T)$. Clearly, the temporal cost of the algorithm is determined by the cost of the viewpoint selection metric and the number of triangles.

4.6 Results

We carried out our tests with low complexity models from CAD programs. All models were simplified on a Pentium Xeon 2GHz with 1GB RAM and an NVIDIA 7800 GTX 512MB graphics card from 20 viewpoints. At the end of this section we show some experiments with a different number of viewpoints. The results obtained with the viewpoint-driven simplification method were compared to the results achieved with QSlim v2.0 [GH97], using the best half-edge collapse, at the same level of simplification. We chose QSlim because it is a well-known pure geometric-based algorithm, it is freely available, and it produces high quality simplifications. The images shown were obtained using different viewpoints from those used during the simplification process.

Model	**Triangles**		**RMSE**		**Time**	
	Original	**Final**	H_v	**VMI**	H_v	**VMI**
Shark	734	80	14.78	14.65	10.24	10.23
Galo	6592	500	9.34	8.55	141.75	142.24
Greekship	9510	600	13.37	12.85	241.78	246.72
Simpletree	11 136	600	16.98	16.27	321.06	332.49
Hammer	13 380	500	8.13	7.43	404.33	423.05
Elephant	31 548	900	13.18	11.14	1756.34	1795.74

Table 4.1: Results for H_v and VMI measuring visual error (RMSE) and simplification time in seconds

First, a comparison between viewpoint entropy and mutual information was performed for all models, the results of which appear in Table 4.1. As shown in this table the visual error is lower in VMI. The temporal cost is almost the same. This difference between the VMI and the H_v lies in the calculation performed to obtain the mean projected area of the polygons. In fact, this is not necessary in H_v. Therefore the temporal cost of H_v is just a bit lower.

Figure 4.9 shows the Galo Model simplified with H_v (Figure 4.9(c)) and with VMI (Figure 4.9(e)). As can be appreciated, the silhouette of this model is kept better with VMI than H_v, see for instance the crest. The wireframe images reveal how these measures work. H_v balances the size of the triangles, therefore all triangles have more or less the same area. However, VMI increases the simplification in flat regions such as the base and the tail. This behavior is desirable and allows VMI to reduce the simplification in other parts of the model, preserving better the visual similarity of the simplified model. In the rest of the section, we used VMI because of its high quality compared to H_v.

Model	Triangles		RMSE		Metro		Time	
	Original	Final	QSlim	VMI	QSlim	VMI	QSlim	VMI
Shark	734	80	33.41	14.65	0.20	0.04	0.02	10.20
Galo	6592	500	12.40	8.55	0.05	0.01	0.08	142.24
Greekship	9510	600	17.20	12.85	0.21	0.09	0.11	246.72
Simpletree	11 136	600	20.73	16.27	0.11	0.14	0.20	332.49
Hammer	13 380	500	8.99	7.43	0.03	0.04	0.20	423.05
Elephant	31 548	900	25.32	11.14	0.08	0.06	0.52	1795.74

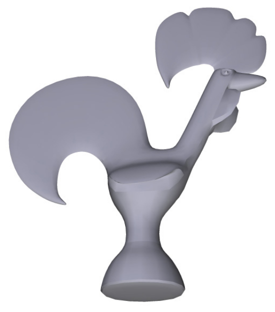
Table 4.2: Results for QSlim and VMI measuring visual error (RMSE), geometric error (Metro) and simplification time in seconds

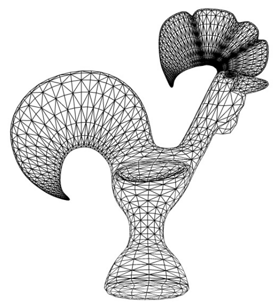
Table 4.2 depicts the visual and geometric error introduced in our experiments and the simplification time. We implemented the root mean square error (*RMSE*) of the pixel-to-pixel image difference defined in [LT00] to measure the mean visual error between the original and the simplified model. This error was taken using 24 viewpoints and 512 $\times$ 512 resolution images; both the number of viewpoints and the resolution were different from those used to simplify all models. We must emphasize that each viewpoint was different from the one used during the simplification process. Clearly, the visual error committed in our method is quite low compare to QSlim, and can even be 50% lower, as shown in case of the Shark, the Galo and the Elephant models.

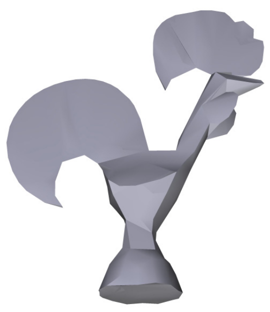
We measured the geometric error using the mesh comparison tool called *Metro* v4.06 [CRS98]. This tool measures the Hausdorff distance between two meshes. Our results are rather better than the geometric method used for comparison purposes. This makes us highly confident about our approach. For example, the geometric error committed in the Galo and Greekship models using VMI is 50% less than with QSlim and up to 75% in the Shark model. However, it is possible that in particular models (for instance, the Simpletree model) this error will be slightly higher than QSlim. The reason for this is because the model has lots of hidden interiors and these are completely removed, thus increasing the geometric error but not the visual one. In the Simpletree model, some branches are removed at the tree top when simplifying with VMI.

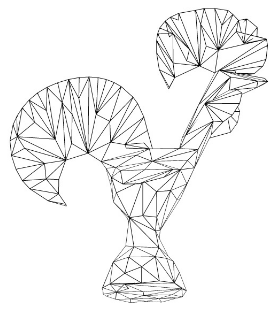
An analysis of the temporal cost is also shown in this table. This cost is proportional to the complexity of the model and the final number of triangles demanded. However, the QSlim algorithm is extremely fast. Its times for these models are less than a second. In any case, our method produces high quality simplifications as far as visual similarity is concerned. So it is valid, despite the high temporal cost compared to geometry-based methods.

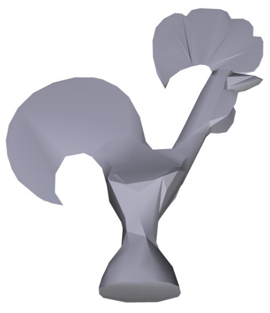
Figure 4.10 shows the results for the Shark model. VMI achieves much


(a) Original model. T=6592


(b) Wireframe


(c) H_v.T=500


(d) Wireframe


(e) VMI.T=500

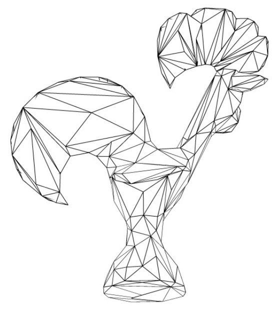

(f) Wireframe

Figure 4.9: Galo model. Image (a) and (b) show the original model. Images (c) and (d) show the model simplified with H_v and (e) and (f) with VMI

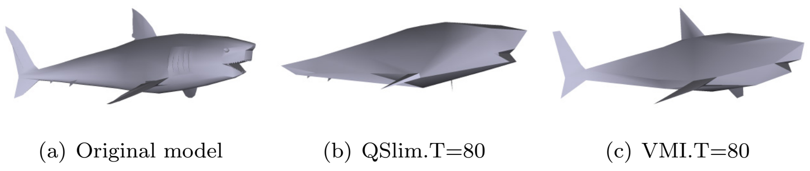


Figure 4.10: Results for the Shark model. Image (a) shows the original model (T=734), (b) the model simplified with QSlim and (c) with VMI

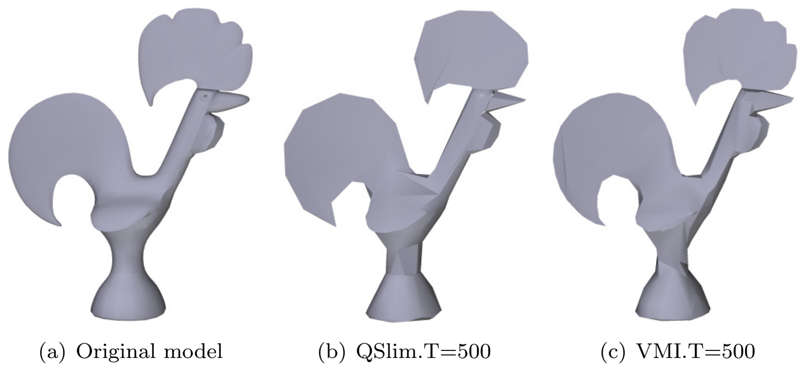


Figure 4.11: Results for the Galo model. Image (a) shows the original model (T=6592), (b) the model simplified with QSlim and (c) with VMI

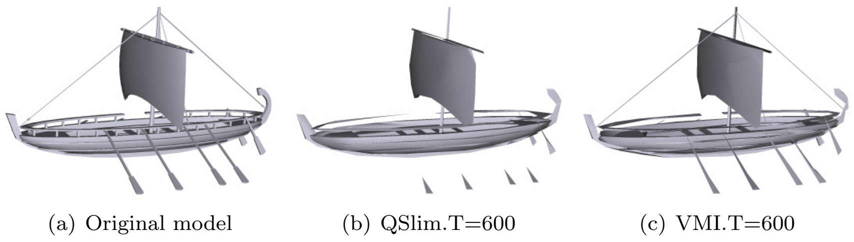
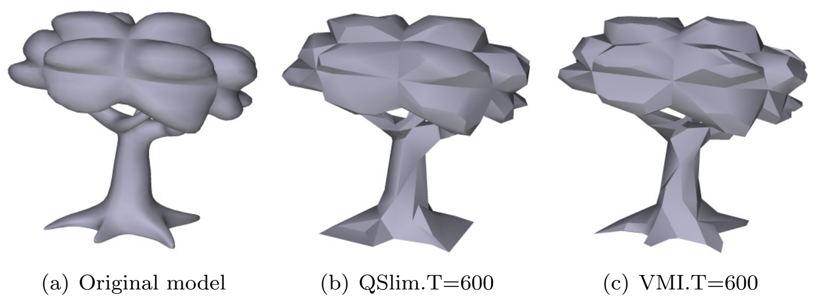
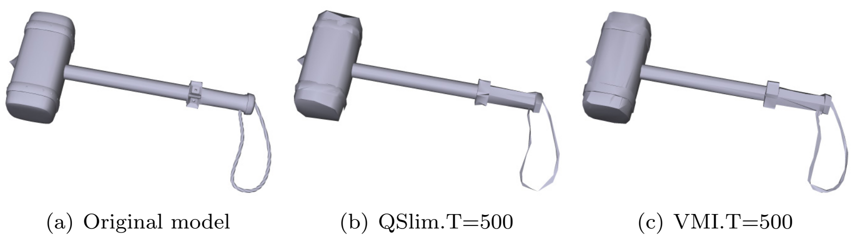


Figure 4.12: Results for the Greekship model. Image (a) shows the original model (T=9510), (b) the model simplified with QSlim and (c) with VMI



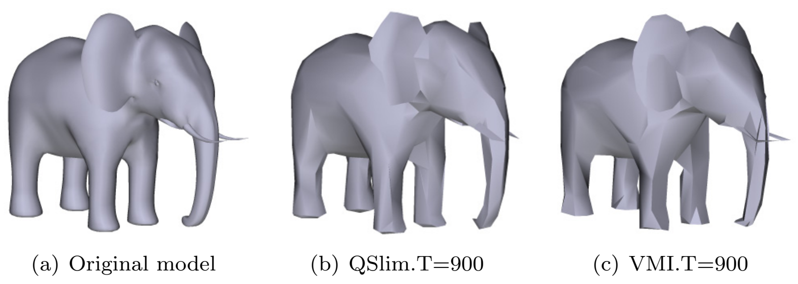
(a) Original model (b) QSlim.T=600 (c) VMI.T=600

Figure 4.13: Results for the Simpletree model. Image (a) shows the original model (T=11 136), (b) the model simplified with QSlim and (c) with VMI



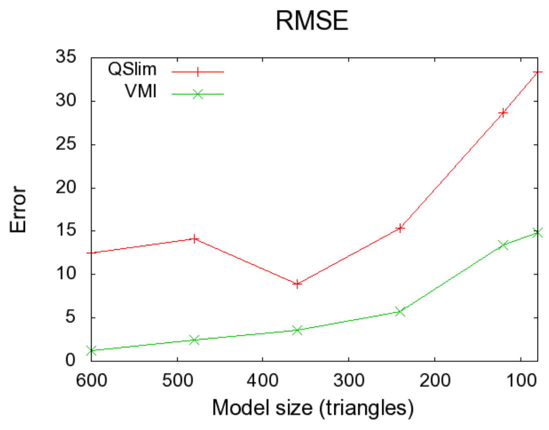
(a) Original model (b) QSlim.T=500 (c) VMI.T=500

Figure 4.14: Results for the Hammer model. Image (a) shows the original model (T=13 380), (b) the model simplified with QSlim and (c) with VMI

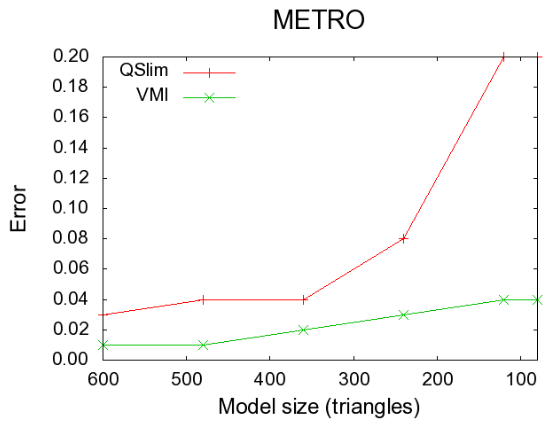


(a) Original model (b) QSlim.T=900 (c) VMI.T=900

Figure 4.15: Results for the Elephant model. Image (a) shows the original model (T=31 548), (b) the model simplified with QSlim and (c) with VMI



(a)



(b)

Figure 4.16: Errors measured for the Shark model at different levels of simplification. Chart (a) shows the visual error (RMSE) and (b) the geometric error (Metro)

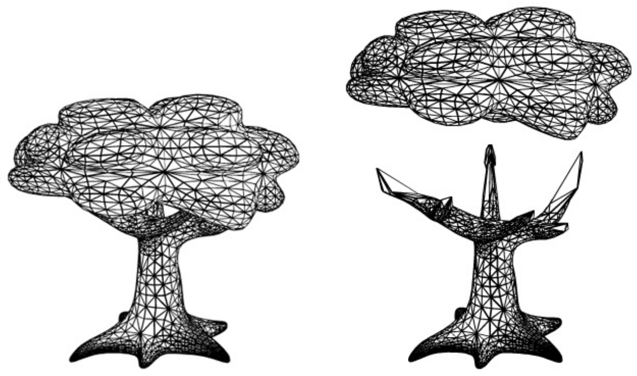


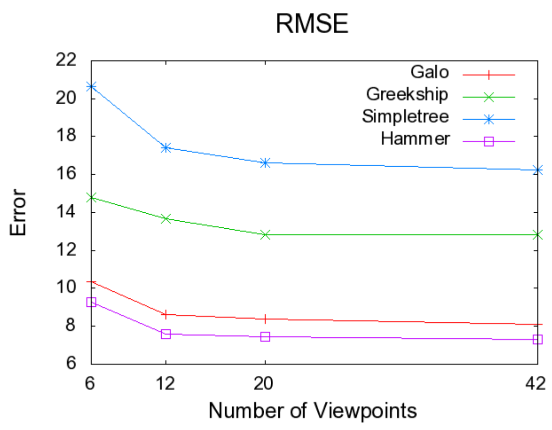
Figure 4.17: VMI operating in the first stages of simplification (T=7400) for the Simpletree model. VMI is able to remove all the hidden interiors

better simplification than QSlim. The fins, the head and the tail are kept better in VMI than in QSlim. Figure 4.11 shows the Galo model. The tail and the crest are maintained better in VMI. Figure 4.12 shows the Greekship model. The oars are kept much better in VMI, as well as the rope that comes from the mast. In QSlim it is removed completely. Figure 4.13 shows the Simpletree model. The roots at the base of the trunk are kept much better in VMI and, additionally, VMI is able to preserve the tree-top better than QSlim. Figure 4.14 shows the Hammer model. The handle is preserved better in our method than with QSlim. Furthermore, VMI allows the shape of the metal base to be kept better. Lastly, Figure 4.15 shows the results for the Elephant model. VMI preserves the shape of the ears and the tusks much better than QSlim. In summary, VMI achieves better simplification than the geometric method QSlim. The difference between VMI and QSlim is bigger if the model presents lots of hidden interiors, in which case VMI can accomplish much better simplifications.

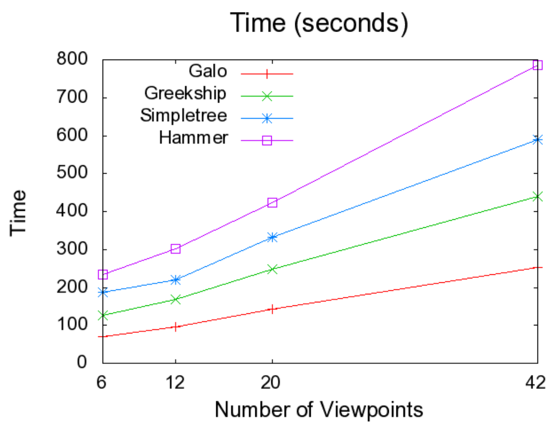
Figure 4.16 shows how our measure and QSlim work at several degrees of simplification for the Shark model. We measured the visual and geometric error. As shown in Figure 4.16(a), if we increase the level of simplification the difference between VMI and QSlim becomes larger and the visual quality of VMI is even much higher. Figure 4.16(b) shows that the geometric error of VMI is also lower than that of QSlim.

Figure 4.17 shows how VMI works at very early simplification levels. In this case we analyze the Simpletree model since it presents hidden interiors in the branches which are in contact with the tree-top. The Simpletree model was simplified to about 66% (T=7400). As shown in this figure, VMI accomplishes a great level of simplification in this region. At this level, most simplifications are focused on hidden interiors.

Lastly, we also conducted experiments with a different number of viewpoints



(a)



(b)

Figure 4.18: Errors measured for some models simplified with VMI using different numbers of viewpoints. Chart (a) shows the visual error (RMSE) and (b) the temporal cost

for some models, as shown in Figure 4.18. These viewpoints are distributed uniformly in direction and correspond to the vertices of the Platonic solids. The last configuration (42 viewpoints) was obtained by subdivision. In this figure the temporal cost and the visual error are analyzed. The visual error hardly improves when the number of cameras increases from 20 to 42. Nevertheless, the temporal cost is about twice as high (see Figure 4.18(b)). Therefore, we think that with 20 viewpoints we already obtain good results, and thus this number of viewpoints is a good compromise between quality and efficiency.

4.7 Conclusions

In this chapter we have presented a new simplification method for polygonal models based on viewpoint selection measures. This method quantifies the error introduced by a decimation operation as the variation in these measures for all the viewpoints. In order to conduct the simplification process, two information-theoretic viewpoint selection measures have been employed, namely, viewpoint entropy and mutual information. Both measures produce high quality approximations taking into account visual fidelity. Nevertheless, viewpoint mutual information decreases the visual error for the simplified models more than viewpoint entropy, since it is able to preserve the silhouette better. In addition, our approach also performs a better simplification than the quadric QSlim method, mainly because it can benefit from visibility information. As shown in our experiments, the resulting approximations have fewer visual and geometric errors. Moreover, our algorithm achieves very good results with CAD models which often present a lot of hidden interiors. Viewpoint entropy and mutual information achieve good results in visual similarity by being able to remove all these hidden interiors.

In general, the main drawback of image-based methods is the high temporal cost. Our approach, based on viewpoints, also has a high cost compared to geometric-based simplifications. In order to compute the error introduced at each simplification step, the model must be rendered and the framebuffer analyzed from every viewpoint, and therefore most of the time the bottleneck resides in the GPU and bus traffic. Nonetheless, in addition to being able to accomplish better simplified models while minimizing visual error, our proposal is even able to reduce the geometric error.

CHAPTER 5
Viewpoint-Driven Simplification Using *f*-divergences

5.1 Introduction

Currently, the majority of published simplification algorithms deal with the polygonal simplification framework from two different perspectives. A common point of view is the geometric fidelity while another is the visual quality. The solution up to now has been to develop algorithms which individually consider one of those approaches. Many geometric-based algorithms produce approximations with very low geometric error but with a poor visual quality. However, the contrary case also often happens, many visual-driven algorithms perform a high visual simplification but with a low geometric quality, especially when the model is simplified to a coarse level. In general, visual-driven approaches completely remove the hidden parts of the simplified model in order to obtain the best results in visual quality. This may sometimes be desirable depending on the type of applications, for instance, computer games. Nevertheless, it is not allowed in others such as medical applications.

Therefore, no algorithm today excels at simplifying all models for the different kinds of applications. This chapter is an attempt to tackle the problem from both points of view. The simplification scheme introduced in the previous chapter is extended with new viewpoint selection measures. These new measures now allow us to take into consideration those different perspectives at the same time. Moreover, it is not necessary to weight the effect of the geometric measure in relation with the visibility information as some other approaches have proposed.

In this chapter, some viewpoint selection measures based on f-divergences are introduced and applied to the viewpoint-driven simplification algorithm. These divergences were proposed by Csiszár [Csi63] and Ali and Silvey [AS66] as measures of discrimination between probability distributions. At the moment, these measures have been successfully applied to image processing, several areas of engineering and computer graphics. For instance, in computer graphics, they have been used to define refinement criteria for hierarchical radiosity and adaptive supersampling of a pixel in ray-tracing. The Kullback-Leibler distance has been recently employed as a measure of viewpoint quality in [SPFG05].

Experimental results show the better visual and geometric performance of our method compared to the high quality geometry-based algorithm [GH97]. The effect of the f-divergences in our viewpoint-driven simplification method was also compared to the viewpoint selection measures tested previously (viewpoint entropy and mutual information). In this case, we attain simplified models which present a similar visual quality but with an enhanced geometric error.

This chapter is organized as follows. Section 5.2 introduces some f-divergences. Section 5.3 defines new viewpoint selection measures based on these divergences. In Section 5.4, these measures are applied to the viewpoint-driven simplification algorithm. Section 5.5 shows the results of the experiments. Finally, in Section 5.6, conclusions are presented.

5.2 *f*-divergences

Many different measures quantifying the degree of discrimination between two probability distributions have been studied in the past. This section presents some of the most important f-divergences [Dra00], called *distances* in the literature.

The *Kullback-Leibler distance* [KL51] between two probability distributions p and q is defined by

$$KL(p \mid q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}, \tag{5.1}$$

where the convention that $0 \log 0 = 0$, $p(x) \log \frac{p(x)}{0} = \infty$ if $p(x) > 0$, and $0 \log \frac{0}{0} = 0$ is used.

The *Chi-Square distance* [Pea00] between two probability distributions p and q is defined by

$$\mathcal{X}^2(p \mid q) = \sum_{x \in \mathcal{X}} \frac{(p(x) - q(x))^2}{q(x)}. \tag{5.2}$$

The *Hellinger distance* [Hel09] between two probability distributions p and q is defined by

$$h^2(p \mid q) = \frac{1}{2} \sum_{x \in \mathcal{X}} (\sqrt{p(x)} - \sqrt{q(x)})^2. \tag{5.3}$$

Note that none of the above distance fulfills all the properties of a true metric. However, $h(p \mid q)$, the square root of the Hellinger distance, is a true metric.

5.3 *f*-divergences for Viewpoint Selection

5.3.1 Viewpoint Kullback-Leibler distance

In [SPFG05], a viewpoint quality measure based on the *Kullback-Leibler distance* (5.1) was proposed. This measure minimizes the distance between the projected and actual area distributions of the polygons in the object and is defined by

$$KL_v = \sum_{i=1}^{N_f} \frac{a_i}{a_t} \log \frac{\frac{a_i}{a_t}}{\frac{A_i}{A_T}}, \tag{5.4}$$

where a_i is the projected area of polygon i, $a_t = \sum_{i=1}^{N_f} a_i$, A_i is the actual area of polygon i and $A_T = \sum_{i=1}^{N_f} A_i$ is the total area of the scene or object.

5.3.2 Viewpoint Chi-Square distance

Using the same probability distributions as in [SPFG05], we define from 5.2 a new viewpoint quality measure, the *viewpoint Chi-Square distance* given by

$$CS_v = \sum_{i=1}^{N_f} \frac{\left(\frac{a_i}{a_t} - \frac{A_i}{A_T}\right)^2}{\frac{A_i}{A_T}}. \tag{5.5}$$

5.3.3 Viewpoint Hellinger distance

Again, another viewpoint quality measure can also be defined from 5.3, the *viewpoint Hellinger distance.* It is given by

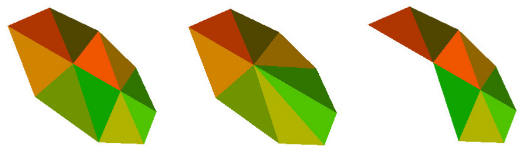
$$HE_v = \frac{1}{2} \sum_{i=1}^{N_f} \left(\sqrt{\frac{a_i}{a_t}} - \sqrt{\frac{A_i}{A_T}} \right)^2 . \tag{5.6}$$

These three measures (KL_v, CS_v and HE_v) represent the distance between the normalized distribution of projected areas and the 'ideal' projection, given by the normalized distribution of actual areas. Note that in this case the background can not be taken into account. The minimum value 0 is obtained when the normalized distribution of projected areas is equal to the normalized distribution of actual areas. Thus, selecting high quality views means minimizing these measures, where *quality* is interpreted as *representativeness.*

5.4 Extending viewpoint-driven simplification with *f*-divergences

The viewpoint-driven simplification algorithm can use any viewpoint selection measure to drive the simplification process. As seen in the previous chapter, the simplification error metric is defined as the sum of differences before simplifying and after simplifying. Therefore, there is no problem in extending the algorithm to incorporate the new f-divergences for viewpoint selection proposed in this chapter. However, some parts related to the computation of these new metrics have to be adapted.

For instance, in addition to the projected areas, the actual areas are also needed in order to compute the f-divergences. Obviously, this will increase the temporal cost of the algorithm compared to the previous version. Nevertheless, the most important fact is that the cost of an edge collapse can not be calculated iteratively as happened in case of viewpoint entropy and mutual information. This is because the background plays no role, that is, it is not considered as another polygon. The total projected area is always the image resolution. However, the total actual area is not a constant value. This means that after an edge collapse the total actual area will have changed because some polygons will have been removed. Therefore, the temporal cost of the algorithm will be increased significantly.



(a) $KL_1 = 0.094$ (b) $KL'_1 = 0.090$ (c) $KL'_1 = 0.053$

Figure 5.1: Image (a) shows the Original Test model (T=10), (b) the resulting model simplified with KL_v after applying the best edge collapse (T=8) and (c) after applying the worst (T=8). T indicates the number of triangles

Figure 5.1 illustrates how KL_v can be employed to conduct the simplification. This figure shows the original Test model after performing the best edge collapse e and the worst edge collapse e'. The best edge collapse belongs to the lowest simplification error C_e (4.4) and the worst to the highest. In addition, this figure shows KL_v from the particular viewpoint ($v = 1$) considered here for all the three analyzed cases. As can be observed, the worst edge collapse has decreased the visible area thus having a new lower KL'_v value. If we compute the error introduced by the best case $C_e = 0.094 - 0.090 = 0.004$, we will see that it is clearly lower than the worst case $C_{e'} = 0.094 - 0.053 = 0.041$.

An important property of KL_v is that it balances the size of polygons. This generates approximations in which polygons are distributed uniformly. But this property in conjunction with the capability of preserving the silhouette is also common in viewpoint entropy (H_v) and mutual information (VMI) when applied to the simplification framework as seen in the previous chapter. Both H_v and VMI only consider the projected area of polygons, and the background plays the same role as another polygon. Hence, the total projected area is always the image resolution. If a polygon is not visible, its contribution to the formula for H_v and VMI will be zero. There will not be a change in H_v and VMI before and after collapsing an edge that belongs to a hidden polygon. Consequently, the error committed will be zero. This means that at the beginning all the invisible polygons will be removed in both H_v and VMI. On the other hand, due to the fact that KL_v considers the actual area of polygons and because the background plays no role, after an edge collapse, normally one or two polygons will be removed, thus decreasing the total actual area. This will change the value for KL_v after an edge collapse. Therefore the error committed will be distinct from zero. The consequence is that even hidden polygons will have error when simplifying and will not be completely removed during the initial steps of the algorithm. Hidden polygons will be removed according to their actual area. Thus, the smallest polygons will be simplified before, preserving the main features of the object in its internal parts. Figure 5.2 illustrates a sequence of simplification for the test model that hides a very simple mesh of two triangles. As shown in this figure, the simple mesh is not simplified until it reaches one of the final steps (h). The other f-divergences proposed in this section share the properties described for KL_v.

5.5 Results

Several experiments with meshes of different complexity were performed. All models were simplified on a Pentium Xeon 2GHz with 1GB RAM and NVIDIA 7800 GTX 512MB graphics card using 20 viewpoints. The effect of using different number of viewpoints is also analyzed at the end of this section. The results obtained at the same simplification level were compared to the results with the geometric QSlim v2.0 [GH97] algorithm using the best half-edge collapse and the viewpoint-driven simplification algorithm using viewpoint entropy. The QSlim algorithm was chosen for the high quality of its approximations and because the code is freely available. The images shown were obtained using a different viewpoint from those used during the simplification.

A comparison among the different f-divergences is performed in Table 5.1. As shown, the best results are given using the Chi-Square distance. Consequently, this distance was used along the remainder of this section for all the experiments.

In Chapter 4 we demonstrated the superior visual quality of VMI with

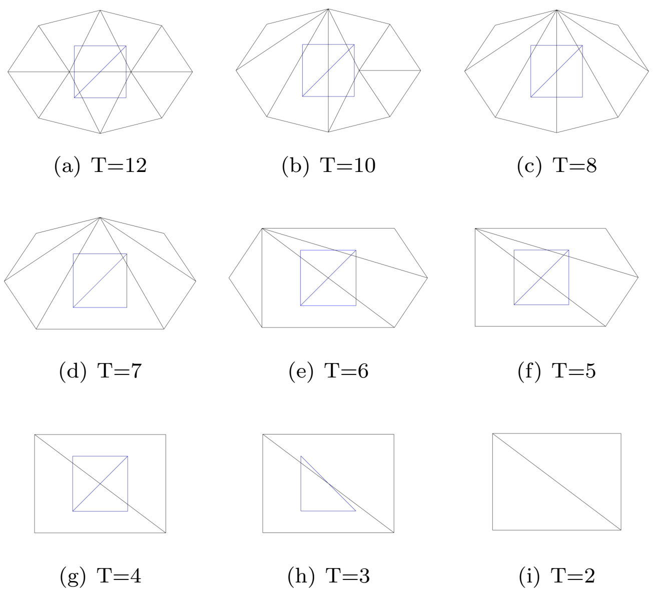


Figure 5.2: Simplification sequence using KL_v

Model	**Triangles**		**RMSE**			**Metro**		
	Original	**Final**	KL_v	HE_v	CS_v	KL_v	HE_v	CS_v
Fish	815	100	12.98	13.46	11.86	0.03	0.04	0.02
Galo	6592	500	10.48	11.27	9.10	0.01	0.01	0.01
Al	7124	1000	12.07	13.62	11.94	0.03	0.04	0.04
Simpletree	11 136	600	18.04	17.75	17.02	0.04	0.04	0.04
Big_atc	13 594	1000	15.44	16.27	15.28	0.03	0.04	0.03
Elephant	31 548	900	13.40	23.00	11.02	0.05	0.08	0.02

Table 5.1: Errors measured for all models using KL_v, HE_v and CS_v

Model	**Triangles**		**RMSE**			**Metro**		
	Original	**Final**	**QSlim**	H_v	CS_v	**QSlim**	H_v	CS_v
Fish	815	100	22.83	11.57	11.86	0.09	0.03	0.02
Galo	6592	500	12.40	9.34	9.10	0.05	0.03	0.01
Al	7124	1000	17.66	11.47	11.94	0.03	0.08	0.04
Simpletree	11 136	600	20.73	16.98	17.02	0.11	0.13	0.04
Big_atc	13 594	1000	16.50	15.97	15.28	0.08	0.05	0.03
Elephant	31 548	900	25.32	13.18	11.02	0.08	0.14	0.02

Table 5.2: Errors measured for all models using QSlim, H_v and CS_v

Model	Triangles		Time		
	Original	Final	QSlim	H_v	CS_v
Fish	815	100	0.03	10.01	11.31
Galo	6592	500	0.08	141.75	237.30
Al	7124	1000	0.08	150.90	273.18
Simpletree	11 136	600	0.20	332.49	605.49
Big_atc	13 594	1000	0.27	535.23	835.88
Elephant	31 548	900	0.52	1756.34	4016.78

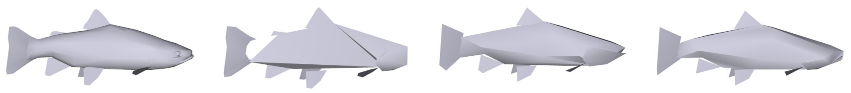
Table 5.3: Simplification time (seconds) for all models using QSlim, H_v and CS_v

respect to H_v. Now, we use H_v to show that the visual quality of the f-divergences is similar to H_v.

Table 5.2 depicts the visual and geometric error. As in the previous chapter, we used the root mean square error (*RMSE*) of the pixel-to-pixel image difference defined in [LT00] to measure the mean visual error between the original and the simplified model. This error was taken using 24 viewpoints and 512×512 resolution images. We must emphasize that each viewpoint was different from the one used during the simplification process. Clearly, the visual error committed with CS_v is quite low compare to QSlim, and can even be 50% lower, as shown in case of the Fish and the Elephant model. However, the visual error is slightly improved in H_v. This is due to the fact that H_v removes completely the hidden interiors and non-visible regions of the model increasing the visual quality.

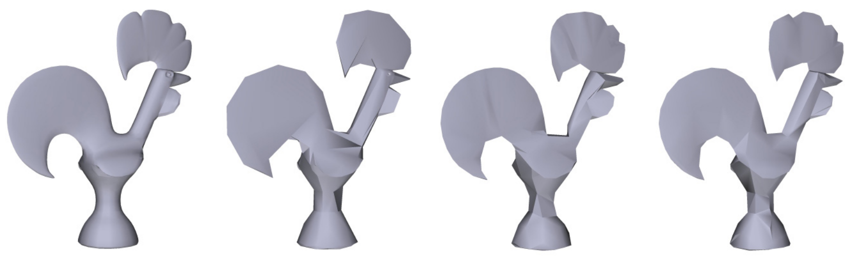
The geometric error was measured using *Metro* v4.06 [CRS98]. The results of CS_v are better in all the models than the geometric and visual simplification methods used for comparison purposes. For example, the geometric error committed in the Galo, the Simpletree and the Big_atc models using CS_v is at least 75% less than with QSlim. Figures 5.11 and 5.12, respectively, show the Al and the Simpletree models rendered with transparency. These models have lots of hidden interiors. For instance, in case of the Al model the hidden joints of the arms and the hip are preserved better in CS_v (see Figure 5.11(d)) whereas in H_v (see Figure 5.11(c)) these interpenetrating parts are partially removed. The hidden branches of the Simpletree model are retained better in CS_v (see Figure 5.12(d)) than in H_v (see Figure 5.12(c)). Therefore, in summary, the approximations produced with CS_v preserve the interior regions better than H_v and consequently present a lower geometric error.

As shown in Table 5.3 the temporal cost is proportional to the complexity of the model and the desired number of triangles. However, the QSlim algorithm is extremely fast. Its times for these models are less than a second. H_v is faster than CS_v. The reason is that H_v does not have to compute the actual areas for the polygons and its initial value can be updated at every simplification step because the total projected area is always the image resolution. CS_v considers the total actual area and this area changes at every simplification. Accordingly, CS_v must be recomputed. In any case, CS_v is valid, despite the high temporal cost because it improves the geometric error without decreasing significantly the visual quality. This can be useful in many applications as mentioned earlier.



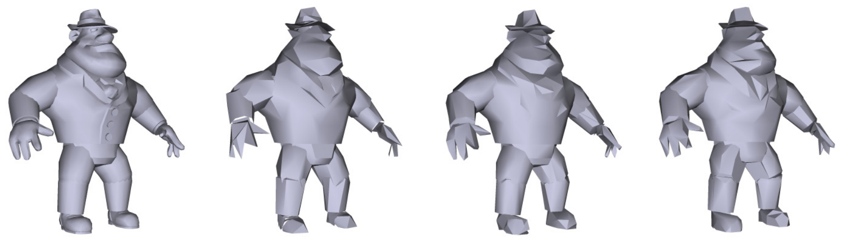
(a) Original model (b) QSlim.T=100 (c) H_v.T=100 (d) CS_v.T=100

Figure 5.3: Results for the Fish model. Image (a) shows the original model (T=815), (b) the model simplified with QSlim, (c) with H_v and d) with CS_v



(a) Original model (b) QSlim.T=500 (c) H_v.T=500 (d) CS_v.T=500

Figure 5.4: Results for the Galo model. Image(a) shows the original model (T=6592), (b) the model simplified with QSlim, (c) with H_v and (d) with CS_v



(a) Original model (b) QSlim.T=1000 (c) H_v.T=1000 (d) CS_v.T=1000

Figure 5.5: Results for the Al model. Image (a) shows the original model (T=7124),(b) the model simplified with QSlim, (c) with H_v and (d) with CS_v

Figure 5.3 shows the results for the Fish model. The head and the tail are kept better in CS_v and H_v than QSlim. Figure 5.4 shows the Galo model.

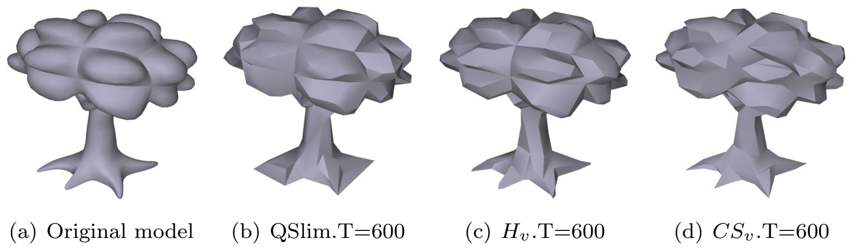

(a) Original model (b) QSlim.T=600 (c) H_v.T=600 (d) CS_v.T=600

Figure 5.6: Results for the Simpletree model. Image (a) shows the original model (T=11 136), (b) the model simplified with QSlim, (c) with H_v and (d) with CS_v

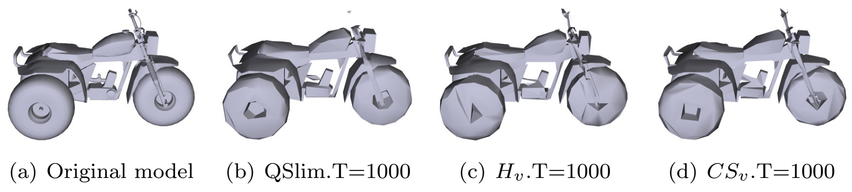

(a) Original model (b) QSlim.T=1000 (c) H_v.T=1000 (d) CS_v.T=1000

Figure 5.7: Results for the Big_atc model. Image (a) shows the original model (T=13 594), (b) the model simplified with QSlim, (c) with H_v and (d) with CS_v

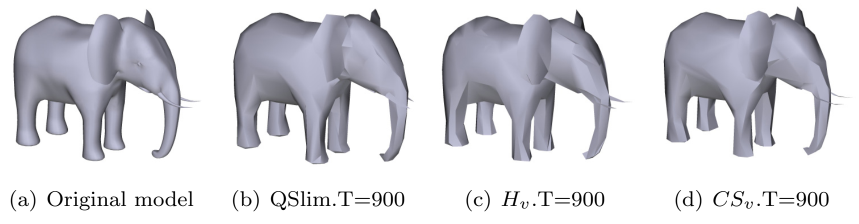

(a) Original model (b) QSlim.T=900 (c) H_v.T=900 (d) CS_v.T=900

Figure 5.8: Results for the Elephant model. Image (a) shows the original model (T=31 548), (b) the model simplified with QSlim, (c) with H_v and (d) with CS_v

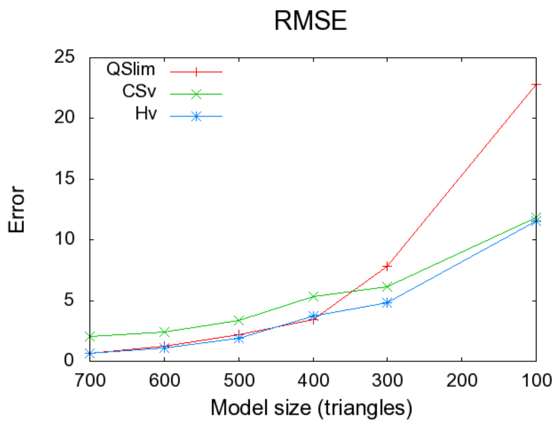
The tail and the crest are kept better in CS_v and H_v. Figure 5.5 shows the Al model. The hands and some parts of the body are preserved better in CS_v and H_v. Figure 5.6 shows the Simpletree model. The roots at the base of the trunk are kept much better in CS_v and H_v and even both are able to keep the tree top better than QSlim. Figure 5.6 shows the Big_atc model. The handlebar is kept much better in H_v and CS_v, as well as the back of the seat. QSlim completely removes it. Lastly, Figure 5.8 shows the results for the Elephant model. CS_v and H_v keep the shape of the ears and the tusks much better than QSlim. To sum up, CS_v and H_v attain better simplification than the QSlim quadric method. The difference between CS_v and H_v is noticeable if the model presents hidden interiors. In such case H_v could accomplish a slightly better visual simplification at the expense of increasing the geometric error, since it may remove completely those hidden interiors.

Figure 5.9 shows how CS_v and the simplification methods used for comparison work at several degrees of simplification for the Fish model. We measured the visual and geometric error. As shown in this figure, if we increase the level of simplification, both visual and geometric quality of QSlim approximations decrease more than CS_v and H_v. The visual difference between CS_v and H_v is practicable indistinguishable for this model, since it does not present hidden interiors. However, in very coarse approximations the geometric error is reduced more in CS_v than in H_v.

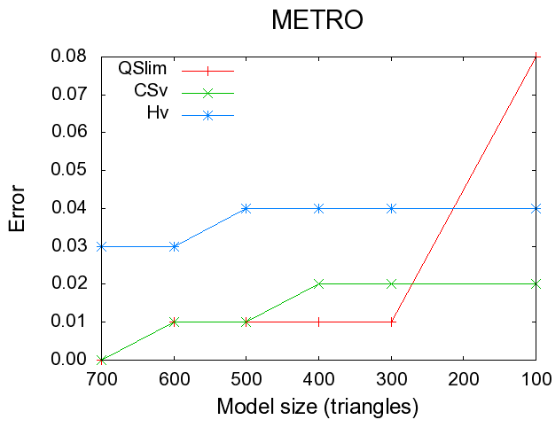
Some experiments with more viewpoints for some of our test models were conducted as shown in Figure 5.10. These different viewpoint configurations correspond to the vertices of Platonic solids. This guarantees that the viewpoints are distributed uniformly. The last configuration (42 viewpoints) was obtained by subdivision of the regular icosahedron. The visual and geometric error was analyzed in this figure together with the temporal cost. Both visual and geometric errors hardly improve when the number of cameras increases from 20 to 42. Nevertheless, the temporal cost is about twice as high (see Figure 5.10(c)). Therefore, we believe that the 20 viewpoints configuration is a good compromise between quality and efficiency.

5.6 Conclusions

In this chapter, the viewpoint-driven simplification algorithm was extended by using f-divergences, convex functions used to compare the distance between two consecutive simplifications. Among the different f-divergences tested, the Chi-Square distance yields the best results when applying to the simplification algorithm. This new scheme is able to produce approximations that accomplish a high visual and geometric fidelity. Indeed, this is because all the distances proposed here consider both visual and geometric information. Unlike many pure visual algorithms, the fact of including the f-divergences permits the viewpoint-driven simplification method not to remove the hidden interiors

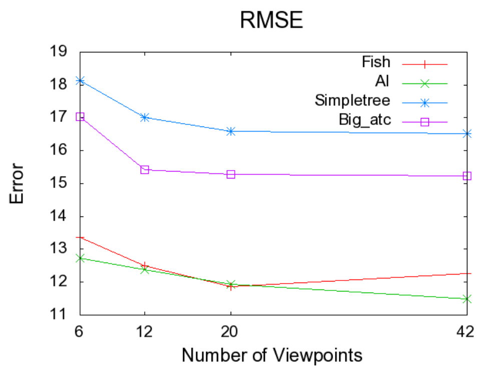


(a)

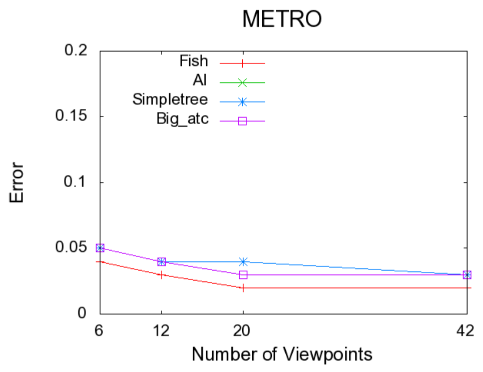


(b)

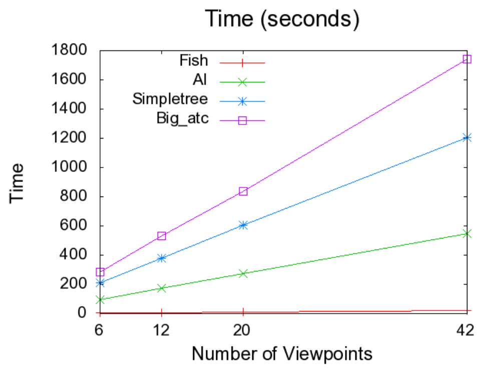
Figure 5.9: Errors measured for the Fish model at different levels of simplification



(a)



(b)



(c)

Figure 5.10: Errors and times measured for some models simplified with CS_v using different number of viewpoints

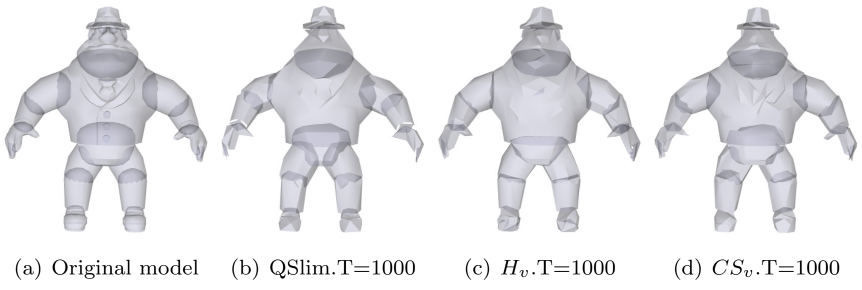


Figure 5.11: The Al model rendered with transparency. Image (a) shows the original model, (b) the model simplified with QSlim, (c) with H_v and (d) with CS_v

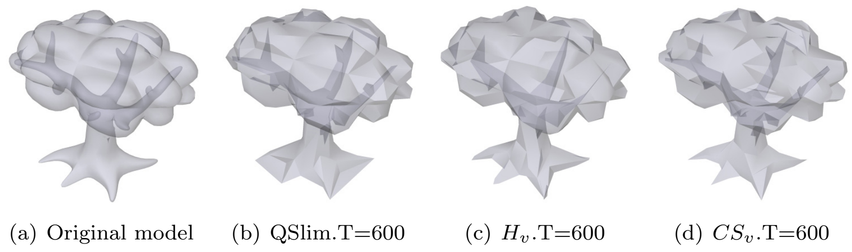


Figure 5.12: The Simpletree model rendered with transparency. Image (a) shows the original model, (b) the model simplified with QSlim, (c) with H_v and (d) with CS_v

completely. Therefore, the geometric quality is not damaged. The silhouette of the model is preserved because the f-divergences also consider the projected areas and this fact allows the achievement of good visual results.

In contrast, the main drawback of applying the f-divergences to the viewpoint-driven simplification scheme is that the temporal cost is increased. As mentioned in Section 5.4, in addition to the projected areas, it is also needed to compute the actual areas of the polygons in the model. However, what really increases the temporal cost is that the f-divergences proposed here can not be computed iteratively. As we have demonstrated, this new approach can be useful because of the high geometric and visual quality of the resulting approximations. We do not need to renounce to a high geometric fidelity if we wish to achieve good results in visual similarity.

CHAPTER 6 Viewpoint-Driven Simplification Using Mesh Saliency

6.1 Introduction

In general, small high-curvature details in the middle of largely flat regions are ignored by most mesh simplification methods, mainly because simplifying these details introduces very little error. However, these small surfaces are likely to be perceived as being important. A flat region in the middle of densely repeated high-curvature bumps is also perceptually important. Repeated patterns, even if high in curvature, are visually monotonous and usually not very interesting. Recently, the concept of saliency has been introduced to identify visually interesting regions that are different from their surrounding context.

Mesh saliency has been incorporated into graphics applications such as mesh simplification and viewpoint selection. For instance, Watanabe and Belyaev [WB01] proposed a method to identify regions in meshes where the main curvatures have locally maximal values along one of the principal directions (typically along ridges and ravines). This method was applied to the problem of simplification, the result being better preservation of the salient features. Hisada et al. [HBK02] proposed a method to detect salient ridges and ravines by computing the skeleton and finding non-manifold points on the skeletal edges and associated surface points. Lee et al. [LVJ05] proposed a saliency for meshes based on the Gaussian-weighted center-surround evaluation of surface curvatures. The center-surround mechanism allows the regions on a mesh that are different from their local context to be identified with greater precision. This notion of saliency was incorporated into mesh simplification and viewpoint selection. Kim and Varshney [KV06] presented a visual-saliency-based operator to enhance selected regions of a volume. Gal and Cohen-Or [GCO06] introduced a method for partial matching of surfaces by using the abstraction of salient geometric features and a method to construct them.

More recently, Feixas et al. [FSG06, FSG07] have proposed a new definition for mesh saliency based on the *Jensen-Shannon divergence*. This approach describes mesh saliency in terms of how polygons see a set of viewpoints.

In this chapter, we incorporate mesh saliency based on the Jensen-Shannon divergence into the viewpoint-driven simplification scheme proposed in this thesis. By including mesh saliency, our simplification method is able to preserve small, but visually significant, salient regions. However, as shown in our experiments, some other regions with low saliency values are simplified more drastically. Consequently, there is no guarantee that the global simplification error may be reduced.

The rest of the chapter is organized as follows. In Section 6.2, we introduce the Jensen-Shannon divergence. In Section 6.3, the concept of saliency of a polygon is introduced. Section 6.4 is devoted to the definition of the salient simplification error and its application to the viewpoint-driven simplification approach. Section 6.5 introduces viewpoint saliency. Section 6.6 shows the results of our experiments. Finally, in Section 6.7 we present our conclusions.

6.2 Jensen-Shannon divergence

A convex function f on the interval $[a, b]$ fulfills the Jensen inequality: $\sum_{i=1}^{n} \lambda_i f(x_i) - f\left(\sum_{i=1}^{n} \lambda_i x_i\right) \geq 0$, where $0 \leq \lambda \leq 1$, $\sum_{i=1}^{n} \lambda_i = 1$, and $x_i \in [a, b]$. For a concave function, the inequality is reversed. If f is substituted by the Shannon entropy, which is a concave function, we obtain the *Jensen-Shannon inequality* [BR82]:

$$\begin{aligned} & JS(\pi_1, \pi_2, \ldots, \pi_N; p_1, p_2, \ldots, p_N) \\ \equiv\ & H\left(\sum_{i=1}^{N} \pi_i p_i\right) - \sum_{i=1}^{N} \pi_i H(p_i) \geq 0, \end{aligned} \tag{6.1}$$

where $JS(\pi_1, \pi_2, \ldots, \pi_N; p_1, p_2, \ldots, p_N)$ is the *Jensen-Shannon divergence* of probability distributions $p_1, p_2, \ldots, p_N$ with prior probabilities or weights $\pi_1, \pi_2, \ldots, \pi_N$, fulfilling $\sum_{i=1}^{N} \pi_i = 1$. JS-divergence measures how ‘far’ the probabilities p_i are from their likely joint source $\sum_{i=1}^{N} \pi_i p_i$ and equals zero if and only if all the p_i are equal. It is important to note that the JS-divergence is identical to $I(X, Y)$ when $\pi_i = p(x_i)$ and $p_i = p(Y \mid x_i)$ for each $x_i \in \mathcal{X}$, where $p(X) = \{p(x_i)\}$ is the input distribution, $p(Y \mid x_i) = \{p(y_1 \mid x_i), p(y_2 \mid x_i), \ldots, p(y_M \mid x_i)\}$, $N =\mid \mathcal{X} \mid$, and $M =\mid \mathcal{Y} \mid$ [BR82, ST00].

6.3 View-based Mesh Saliency

In [FSG06, FSG07], a measure of mesh saliency based on JS-divergence was introduced. Unlike [LVJ05], this approach formulates mesh saliency in terms of how the polygons ‘see’ the set of viewpoints, i.e. according to visual perception. The saliency of a polygon is defined as the average dissimilarity between this polygon and its neighbors. Two polygons are ’similar’ when the JS-divergence between them is small. For each polygon, the average variation of JS between two neighbor polygons is evaluated. Thus, the *polygonal saliency* of o_i is defined by

$$S(o_i) = \frac{1}{N_o} \sum_{j=1}^{N_o} JS \left(\frac{p(o_i)}{p(o_i)+p(o_j)}, \frac{p(o_j)}{p(o_i)+p(o_j)}; p(V \mid o_i), p(V \mid o_j) \right) \geq 0, \quad (6.2)$$

where o_j is a neighbor polygon of o_i, N_o is the number of neighbor polygons of o_i, and the conditional probabilities $p(V \mid o_i)$ and $p(V \mid o_j)$ are weighted by $\frac{p(o_i)}{p(o_i)+p(o_j)}$ and $\frac{p(o_j)}{p(o_i)+p(o_j)}$, respectively. Hence, a polygon o will be salient if the average of the JS-divergences between o and its neighbors is high. On the other hand, a polygon at the center of a smooth region will probably have low saliency, since the polygons in this region will present small visibility differences with respect to the set of viewpoints.

At the initialization stage, a saliency map is built from the saliency of all the polygons. This map is computed just once and is never modified during the simplification process. We have verified that if the saliency map is updated at each step of simplification, the global simplification error is reduced but salient regions are poorly preserved. This can be especially noticeable when the model is simplified to coarse levels.

6.4 Salient Simplification Error

In this chapter, we apply mesh saliency based on JS-divergence to the viewpoint-driven simplification method (see Chapter 4). Nevertheless, it can be easily integrated with any other mesh simplification scheme. In order to consider the salient regions we must change the behavior of the viewpoint-driven simplification method. Specifically, it is necessary for salient regions to increase the simplification error. Thus, the *salient simplification error* for an edge collapse e is now given by

$$C_{total} = C_e + (C_e \times S_e), \quad (6.3)$$

where C_e represents the simplification error deviation from the viewpoint-driven algorithm and S_e is the edge saliency of e. S_e can be interpreted as the average of the saliency of its adjacent polygons. However, it can be calculated more accurately if we consider its value directly from the JS-divergence of its shared polygons. Thus, the *edge saliency* of e is defined as

$$S_e = JS\left(\frac{p(o_i)}{p(o_i)+p(o_j)}, \frac{p(o_j)}{p(o_i)+p(o_j)}; p(V \mid o_i), p(V \mid o_j)\right), \tag{6.4}$$

where o_i and o_j are the two polygons that share edge e. If an edge is a boundary edge, it only shares one polygon, and thus its saliency is considered to be the maximum value. In order to avoid mesh artifacts, the simplification algorithm only deals with 2-manifold edges and boundary edges. Nonetheless, calculation of the JS-divergence between more than two polygons can be performed without any problems, as can be inferred from (6.1).

With the current definition of the salient simplification error, even very small salient values may affect the final result. However, only high salient values are really important because they are perceived as interesting features. Accordingly, we define a threshold α such that we only apply the saliency values that are greater or equal to α. This α is the 30^{th} percentile saliency.

$$C_{total} = \begin{cases} C_e + (C_e \times S_e) & \text{if } S_e \geq \alpha \\ C_e & \text{if } S_e < \alpha \end{cases} \tag{6.5}$$

6.5 Viewpoint Saliency

Similarly to [LVJ05], where mesh saliency was used to select the best views, Feixas et al. [FSG07] proposed a method to calculate the saliency of a viewpoint using the conditional probabilities of the reverse channel. Consequently, the *viewpoint saliency* of v is defined as

$$S(v) = \sum_{o \in \mathcal{O}} S(o) p(v \mid o), \tag{6.6}$$

where the conditional probability $p(v \mid o)$ is obtained from the Bayes theorem $p(v,o) = p(v)p(o \mid v) = p(o)p(v \mid o)$.

In mesh simplification, viewpoint saliency can be used as a criterion to weight the importance of a viewpoint. With this approach, the regions that present the least salient parts of the model will be simplified further. For instance, in a bust, the back will be simplified more than the front, since there are a lot of salient regions on the face, such as the eyes, the nose and the mouth.

The simplification error deviation (4.4) for an edge collapse e in the viewpoint-driven simplification algorithm taking into account viewpoint saliency is now given by

$$C_e = \sum_{v \in \mathcal{V}} S(v) \mid I_v - I'_v \mid . \tag{6.7}$$

6.6 Results

In this section, we present several models that have been simplified using viewpoint-driven simplification with viewpoint mutual information and the mesh saliency introduced above. All models shown in this chapter were simplified on a Pentium Xeon 2GHz with 1GB of RAM and an NVIDIA 7800 GTX 512 MB graphics card using 20 regularly distributed viewpoints and 256×256 resolution images.

For the saliency map, we used a greater number of uniformly distributed viewpoints (42 viewpoints). Additionally, we increased the image resolution to 512×512 pixels. This significantly improves the accuracy of the saliency map. The viewpoints chosen for the map were also different from those used during the simplification process.

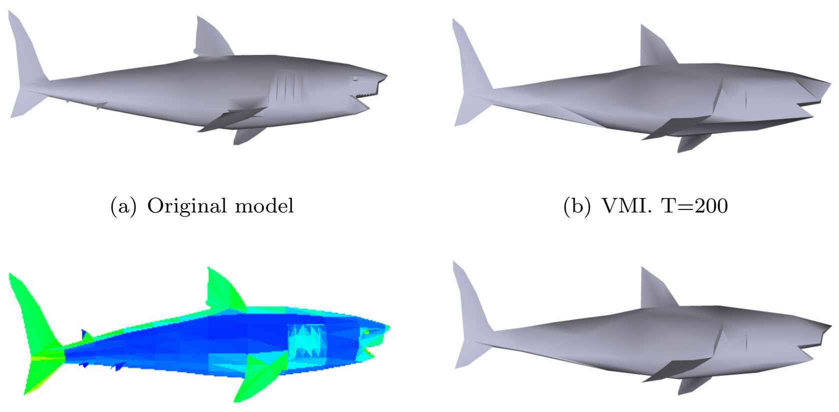
Model	**Triangles**		**VMI**		**VMI + Mesh Saliency**		
	Original	**Final**	**RMSE**	**Time**	**RMSE**	**Time**	
						Map	**Simp.**
Shark	734	200	6.61	8.23	6.46	0.71	8.07
Beethoven	5030	500	11.94	86.12	11.87	4.65	85.53
Galo	6592	500	9.76	127.09	9.78	6.90	123.40
Al	7124	1000	11.61	152.18	11.73	7.11	153.35
Hammer	13 380	750	6.65	369.62	6.72	13.53	363.78

Table 6.1: Visual error (RMSE) and simplification time (seconds) for all models simplified with VMI and mesh saliency

Table 6.1 shows the visual error measured with RMSE and the simplification time for all the models that were tested. As shown in this table, the global visual error is similar when applying mesh saliency, and it can sometimes be even better for some models, for instance, the Beethoven and Shark models. In fact, this depends on how many salient regions there are on the models and how important they are. This table also shows the simplification time. As can be seen, the simplification times are practically the same. Obviously, when using saliency first we must compute the saliency map. Nevertheless, this calculation hardly increases the total time. For instance, in case of the Hammer model if saliency is applied the total time is increased by about 1%. The calculation time for the saliency map is directly related to the number of viewpoints and the image resolution. As shown in our experiments, saliency computation is not a very time-consuming process.

Figure 6.1 shows the results for the Shark model and the saliency map is shown in Figure 6.1(c). The most salient surfaces are represented in warmer colors (reds and yellows) and the least salient are shown in cooler colors (greens and blues). As can be seen in this figure, the most salient parts correspond to the fins and the mouth. These parts are better preserved in the model that is simplified with saliency. See for instance the head and the tail fin.

Figure 6.2 shows the results for the Beethoven model and the saliency map is shown in Figure 6.2(c). As can be seen, the most salient parts, such as the


(a) Original model (b) VMI. T=200

(c) Saliency map (d) VMI + saliency. T=200

Figure 6.1: Shark model. T indicates the number of triangles

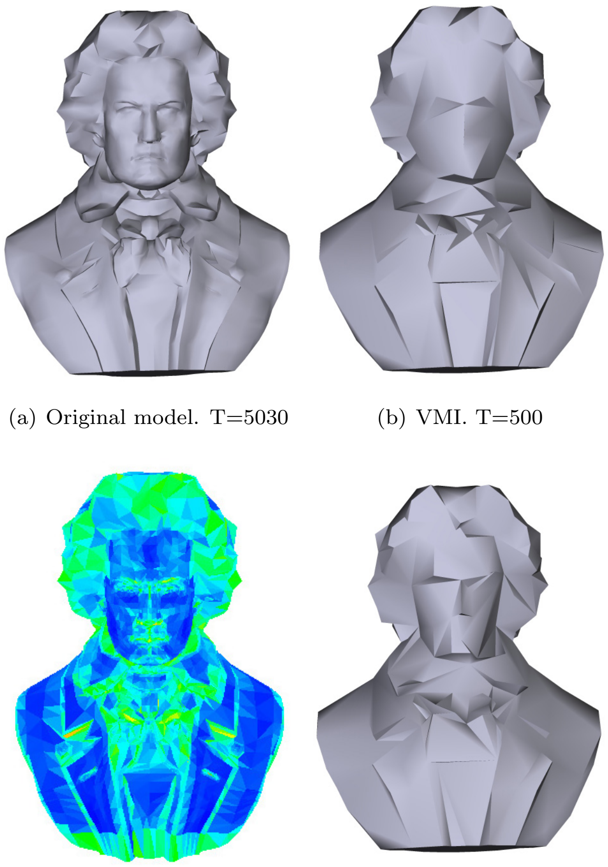
eyes, nose, mouth and the necktie, and some parts of the hair are shown in red, yellow and green. Some other low-saliency parts of the face, such as the forehead and the cheeks are shown in blue. If the model simplified without saliency is compared to that with saliency, it can be appreciated that highly salient regions are preserved better. See for instance the eyes, nose and the shape of the face (Figure 6.3).

Figure 6.4 shows the results for the Galo model. The saliency map reveals that highly salient regions are on the outline of the tail and the crest. As can be clearly seen, these regions are preserved better (see Figure 6.5). However, it can also be appreciated that some regions near the base that present very low saliency are smoothed. These regions are preserved slightly worse. As we mentioned at the beginning, preserving highly salient regions may sometimes decrease the quality of other parts in the model.

Figure 6.6 shows the results for the Al model. The most salient regions for this model are in the fingers, the nose, the ears and the hat. As can be seen, these regions are preserved better when mesh saliency is applied. See for instance the fingers of the right hand.

Figure 6.7 shows the results for the Hammer model. The most salient regions for this model are in the handle, the strips on the metal base and a peak in the middle of the metal base. As can be seen, these salient regions are preserved better when mesh saliency is applied. Figure 6.8 shows a close-up of these regions.

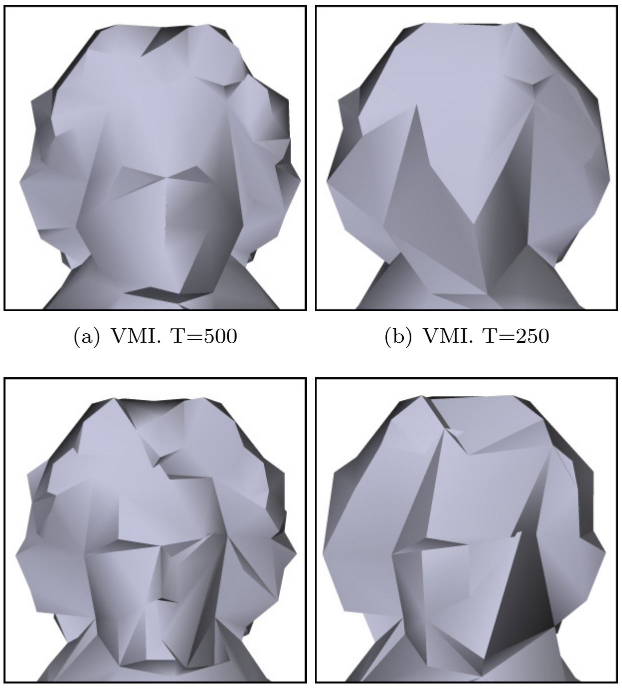
Lastly, Figure 6.9 shows the effects of applying viewpoint saliency in the Beethoven model. As shown in this figure, the most salient viewpoint corresponds to the one that sees the face (see Figure 6.9(a)), and the least is the


(a) Original model. T=5030 (b) VMI. T=500

(c) Saliency map (d) VMI + saliency. T=500

Figure 6.2: Beethoven model

one that sees the back (see Figure 6.9(b)). If we pay attention to the wireframe images, we will see that the model simplified with viewpoint saliency maintains more triangles in the most salient views. In this case, this corresponds to the front part of the bust. If we examine the back part, we will see that the model simplified with viewpoint saliency presents fewer triangles in this part. We also measured the visual error for this model and it was found to be slightly worse (RMSE using VMI + $S(v)$=12.66 and RMSE using VMI=11.94). But this is logical because the least salient views are simplified further and this means that the global error is probably increased. Figure 6.10 shows the results for the Al model and we measure the visual error for this model. In this case, the model that was simplified by applying viewpoint saliency has slightly re-


(a) VMI. T=500 (b) VMI. T=250

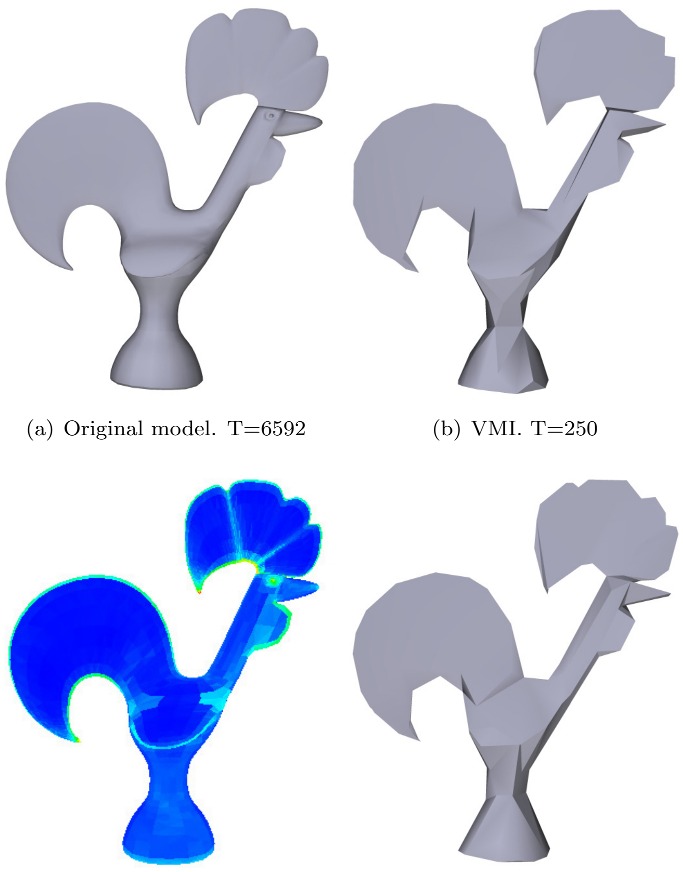
(c) VMI + saliency. T=500 (d) VMI + saliency. T=250

Figure 6.3: Close-ups of Beethoven model

duced the visual error (RMSE using VMI + $S(v)$ = 11.31 and RMSE using VMI=11.61). The hands maintain more triangles and are better preserved if viewpoint saliency is applied.

6.7 Conclusions

Recently, mesh saliency has been introduced as a measure of *regional importance* for graphics meshes. Its main characteristic is the ability to capture what most would classify as visually interesting regions on a mesh. However, note that not all such regions necessarily have high curvature. In this chapter, we have incorporated a view-based mesh saliency based on JS-divergence into viewpoint-driven simplification. Unlike other previous approaches, mesh saliency based on JS-divergence is defined according to visual perception. Its current definition only considers geometry, since it is formulated from the projected areas of polygons. Computing mesh saliency based on JS-divergence is not a very expensive process. In addition, any mesh simplification method can be modified in order to include this saliency.



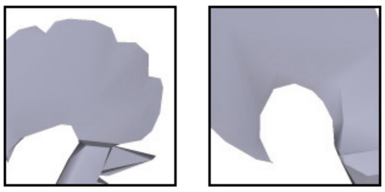
(a) Original model. T=6592 (b) VMI. T=250

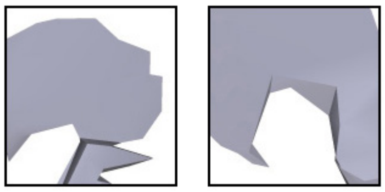
(c) Saliency map (d) VMI + saliency. T=250

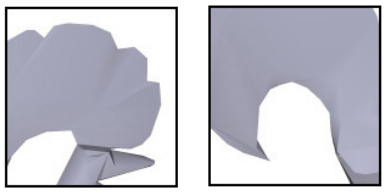
Figure 6.4: Galo model

Mesh saliency allows highly salient regions to be preserved better. Nevertheless, this clearly has a drawback. Low saliency regions are simplified to a coarse level. As a consequence, there is no guarantee that the overall quality of the resulting approximations will always be improved.

Viewpoint saliency can help us to preserve the parts of a model that present lots of salient regions and to increase the simplification in other parts with flat regions. This means that viewpoint saliency can be used as a weighting factor of the importance of a viewpoint. Similar to mesh saliency, viewpoint saliency cannot ensure that the global quality of the simplified models is always enhanced.


(a) VMI. T=500


(b) VMI. T=250


(c) VMI + saliency. T=500

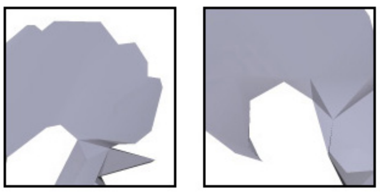
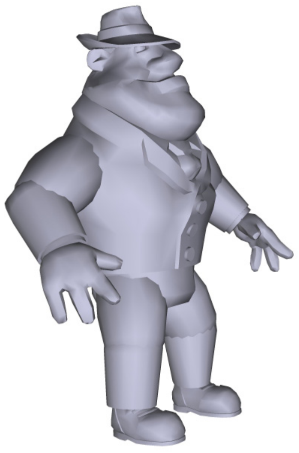
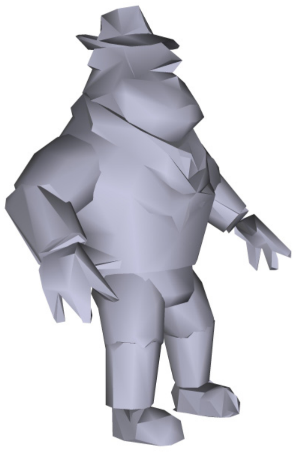
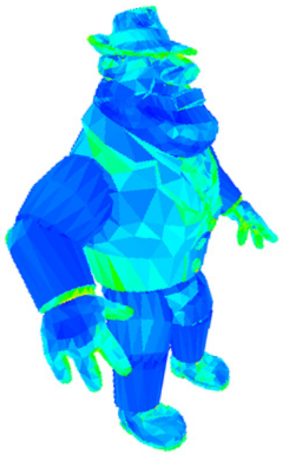

(d) VMI + saliency. T=250

Figure 6.5: Close-ups of Galo model


(a) Original model


(b) VMI. T=1000


(c) Saliency map

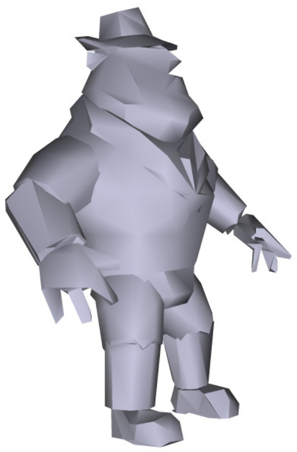
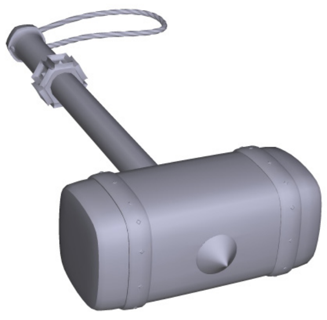
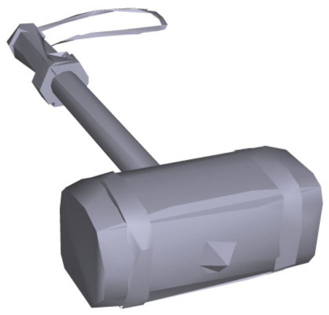
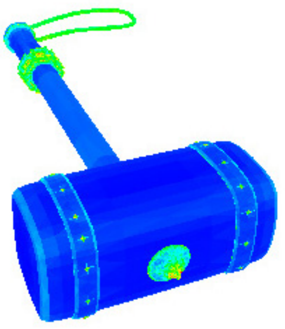

(d) VMI + saliency. T=1000

Figure 6.6: Al model


(a) Original model. T=13 380


(b) VMI. T=750


(c) Saliency map

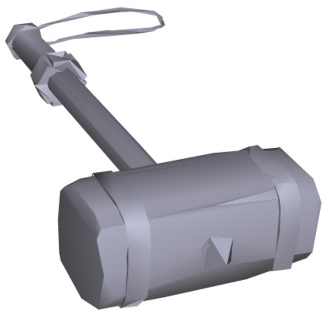
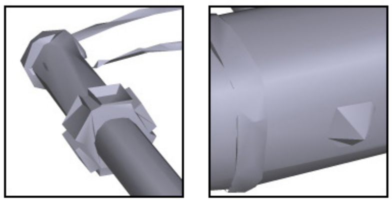
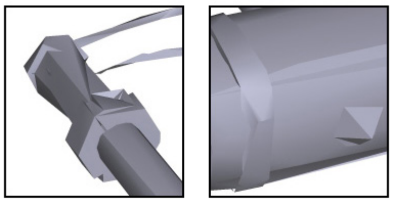

(d) VMI +saliency. T=750

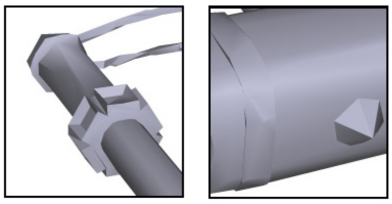
Figure 6.7: Hammer model



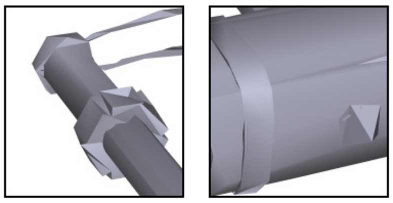
(a) VMI. T=750



(b) VMI. T=400



(c) VMI + saliency. T=750



(d) VMI + saliency. T=400

Figure 6.8: Close-ups of Hammer model

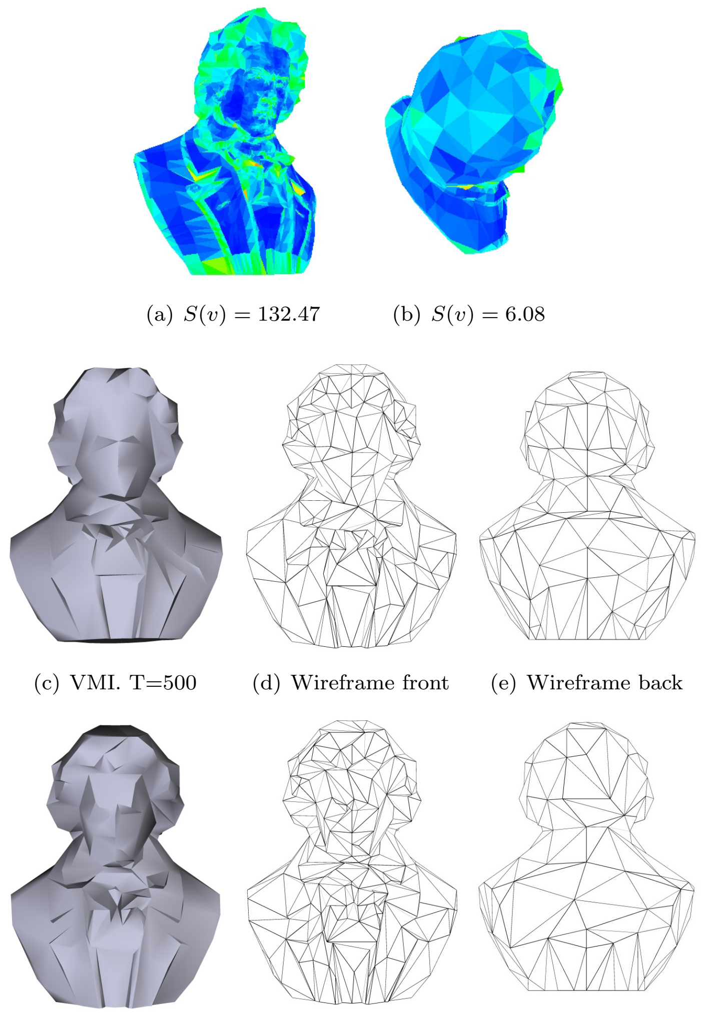


Figure 6.9: Beethoven model. The (a) most salient and (b) least salient views chosen from the 20 views obtained from the vertices of a regular dodecahedron. Images (c-e) show the model simplified with VMI. Images (f-h) show the model simplified with VMI and weighted by viewpoint saliency

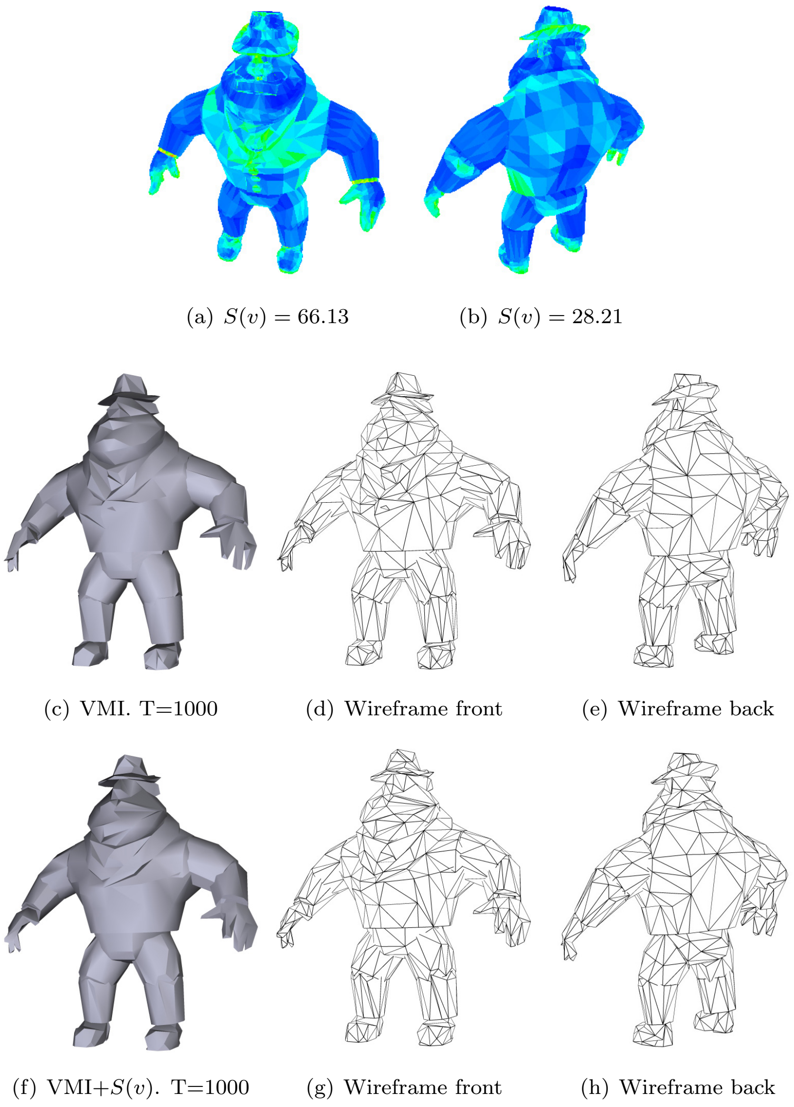


Figure 6.10: Al model. The (a) most salient and (b) least salient views chosen from the 20 views obtained from the vertices of a regular dodecahedron. Images (c-e) show the model simplified with VMI. Images (f-h) show the model simplified with VMI and weighted by viewpoint saliency

CHAPTER 7

Viewpoint-Driven Simplification Using the Best Viewpoints

7.1 Introduction

The number of viewpoints is a very important factor for the viewpoint-driven simplification algorithm. As shown in Chapter 4, the quality of the approximations and the temporal cost of the viewpoint-driven simplification method are directly dependent upon it. However, there is a threshold beyond which, although we increase the number of viewpoints, the quality is not improved. This limit is about 20 uniformly distributed viewpoints, as pointed out in previous chapters.

In this chapter, we analyze how the simplification algorithm behaves with a minimal set of good quality viewpoints and with the same number of viewpoints, but regularly distributed. This minimal set of viewpoints can be used to reduce the temporal cost of the algorithm, since the results that can be achieved with a minimal set of viewpoints are similar to those obtained with a greater number of uniformly distributed points.

Any of the viewpoint selection measures described in this thesis could obviously be used to measure the quality of a viewpoint and to obtain a minimal set of good viewpoints. We consider the Plemenos et al. algorithm [PSF04] for the computation of the minimal set of viewpoints. This algorithm uses a notion of viewpoint quality based on the heuristic measure (HM) [PB96]. In addition, we will experiment with the viewpoint mutual information measure in order to compute a different minimal set of best viewpoints by using the algorithm proposed in [FSG06, FSG07]. First, we will describe the algorithm used to compute the minimal set of viewpoints based on Plemenos et al.'s viewpoint quality measure. Next, we will apply this minimal set of viewpoints to the viewpoint-driven simplification scheme. Briefly, we will introduce the algorithm of the selection of N best views based on VMI and finally we will show the results obtained with the both different best viewpoint distributions.

7.2 Viewpoint Quality Heuristic Measure

Although a quality-of-view criterion is difficult to define, because the notion of 'good view' is partially subjective, a number of objective elements may be retained in order to achieve a close approximation to this notion. The chosen elements are: number of visible polygons, number of visible objects and area of the projected visible part of each polygon. The notion of object is used together with the notion of surface (or polygon) because very often a scene is composed of triangles with low semantic value. The notion of object, obtained by grouping polygons, is more intuitive and useful because it is not necessary to see all the polygons of an object to consider that the object is visible.

In [PB96, BPD00, Ple03, PSF04], the quality of a viewpoint v of a scene is computed using the *heuristic measure* (HM) given by

$$C(v) = \frac{\sum_{i=1}^{n} \left\lceil \frac{P_i(v)}{P_i(v)+1} \right\rceil}{n} + \frac{\sum_{i=1}^{n} P_i(v)}{r}, \tag{7.1}$$

where $P_i(v)$ is the number of pixels corresponding to polygon i in the image obtained from viewpoint v, r is the total number of pixels of the image (resolution of the image), and n is the total number of polygons in the scene. In this formula, $\lceil x \rceil$ denotes the smallest integer, greater than or equal to x. The first term in 7.1 gives the fraction of visible surfaces with respect to the total number of surfaces, while the second term is the ratio between the projected area of the scene (or object) and the screen area (thus, its value is 1 for a closed scene).

7.3 Computing a minimal set of viewpoints

The computation of a minimal set of viewpoints is performed in four steps [PGJT05]:

1. Compute an initial set of viewpoints.

   This is accomplished by applying a uniform sampling on the surface of a sphere surrounding the scene. The uniform sampling of the sphere is performed by increasing the values of two angles θ and ψ. Increments $\Delta\theta$ and $\Delta\psi$ determine the precision of the sampling.

2. Evaluating the initial set of viewpoints.

   Using the viewpoint quality measure (7.1), evaluate each viewpoint in the initial set and sort the set in decreasing order, starting from the best viewpoint and finishing with the worst viewpoint.

   To compute the main quantities required by the viewpoint quality measure, that is, the number of visible polygons, number of visible objects and area of projected visible part of a polygon, several techniques could be used. In [PGJT05], there is a full description of these techniques.

3. Computing a minimal set of viewpoints.

   Starting from the sorted initial viewpoint set, we create a new set by suppressing redundant viewpoints, that is, viewpoints that only allow us to see details that are visible from the viewpoints that have already been selected in the set.

   The current viewpoint in the initial set is compared to all the viewpoints in the reduced set of viewpoints. If all the visible details (surfaces and objects) from the current viewpoint are visible from the viewpoints already stored in the reduced set, the current viewpoint is ignored. Otherwise, the current viewpoint is stored in the reduced set of viewpoints. The reduced set is generally not minimal because every viewpoint in the initial set is only compared to the already existing viewpoints in the reduced set. So, the first viewpoints in the initial set have a higher probability of being retained for the reduced set than the last ones. However, some of these viewpoints may become redundant after a new viewpoint is added to the reduced set. To address this problem, an additional processing of the elements of the reduced set is required. Every viewpoint in the reduced set is compared to all the other viewpoints in the set and, if it does not allow us to see more details than the other viewpoints in the set, it is suppressed.

4. If the number of viewpoints in the final set seems to be too high, apply step 3 to this set once again in order to get a really minimal set.

The final set of non-redundant viewpoints can be sorted according to various criteria such as quality of view, distance from the previous viewpoint, etc.

7.4 Results

The minimal set of viewpoints is computed in a preprocess and hence the temporal cost of the viewpoint-driven simplification algorithm is not increased. We performed some tests with several models of different complexity, and compared the effect of using viewpoints that were uniformly distributed from the vertices of the Platonic solids to that observed using the same number of viewpoints obtained with the minimal set algorithm introduced in the previous section. The models were simplified on a Pentium Xeon 2GHz with 1GB RAM and an NVIDIA 7800 GTX 512MB graphics card.

Model	**Triangles**	**Best viewpoints**		
		6	**12**	**20**
Galleon	4698	20	40	65
Simpletree	11 136	22	44	74
Big_atc	13 594	24	44	73

Table 7.1: Time (seconds) for all models using the minimal set of viewpoints algorithm based on the heuristic measure (HM)

Table 7.1 shows the time spent on the computation of the minimal set of best viewpoints using the algorithm described in Section 7.3. These results were obtained using 256 × 256 resolution images. Note that this preprocess is computed once before the simplification step. As shown in this table, these times are proportional to the number of viewpoints and the complexity of the model.

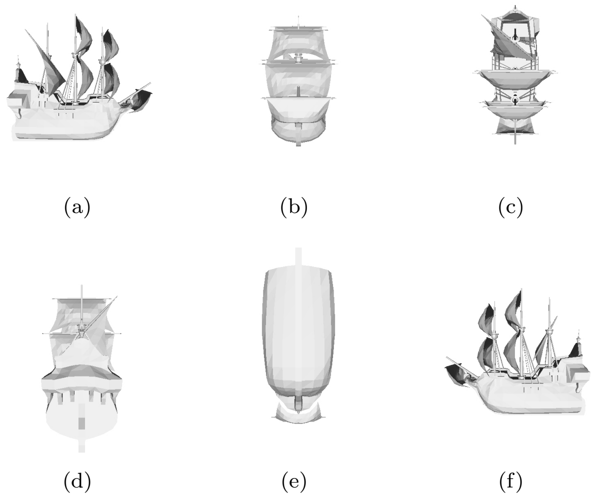


Figure 7.1: The six different views of the Galleon model. This viewpoint distribution corresponds to the vertices of a regular octahedron

Figures 7.1 and 7.2 show two different viewpoint configurations for the Galleon model. Figure 7.1 shows the uniformly distributed viewpoint configuration. This configuration was obtained from the vertices of the regular octahedron, a Platonic solid. Figure 7.2 shows the best viewpoint distribution. This distribution was obtained after applying the minimal set of viewpoints algorithm based on HM. Figures 7.3 and 7.4 show both configurations for the Simpletree model, and Figures 7.5 and 7.6 offer those for the Big_atc model.

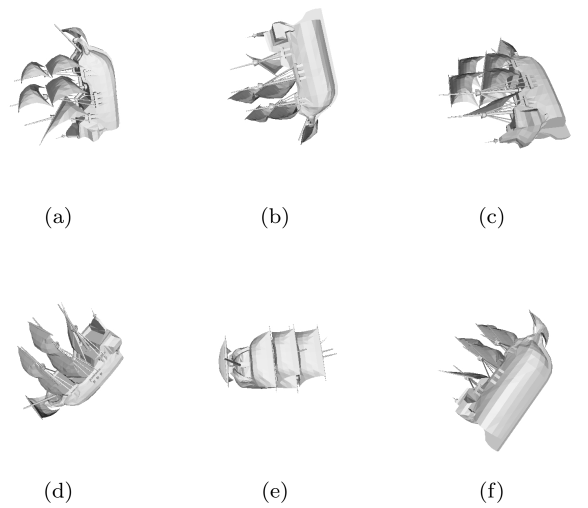


Figure 7.2: The six different views of the Galleon model. This viewpoint distribution was obtained from the minimal set of viewpoints algorithm based on the heuristic measure (HM)

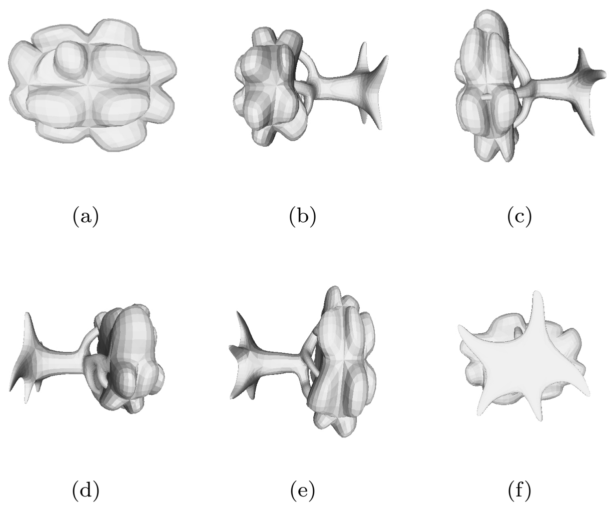


Figure 7.3: The six different views of the Simpletree model. This viewpoint distribution corresponds to the vertices of a regular octahedron

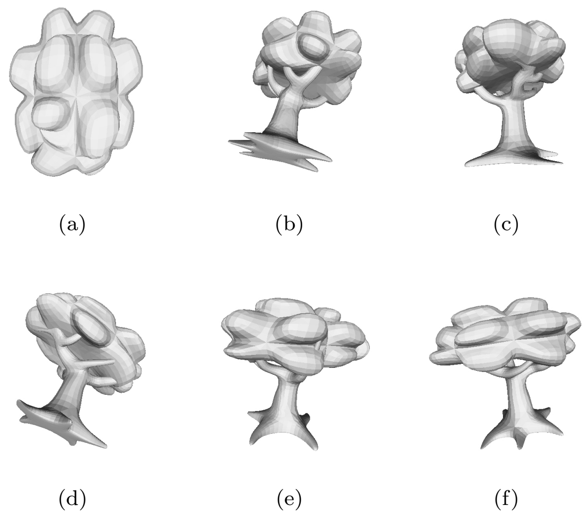


Figure 7.4: The six different views of the Simpletree model. This viewpoint distribution was obtained from the minimal set of viewpoints algorithm based on the heuristic measure (HM)

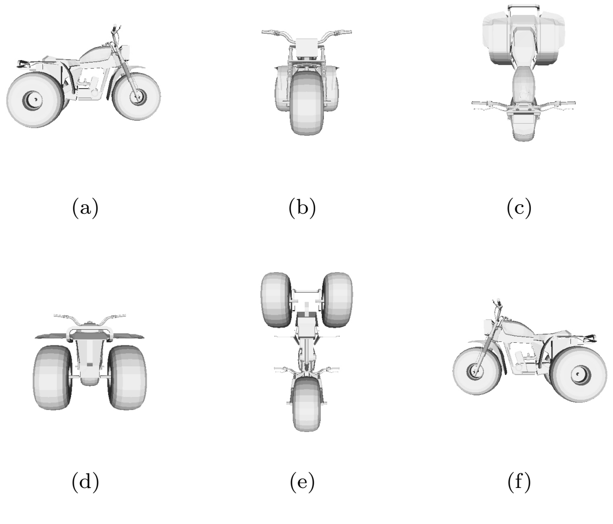


Figure 7.5: The six different views of the Big_atc model. This viewpoint distribution corresponds to the vertices of a regular octahedron

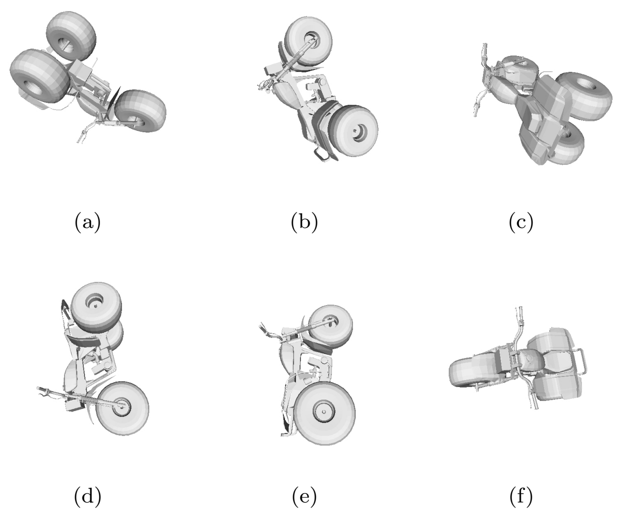


Figure 7.6: The six different views of the Big_atc model. This viewpoint distribution was obtained from the minimal set of viewpoints algorithm based on the heuristic measure (HM)

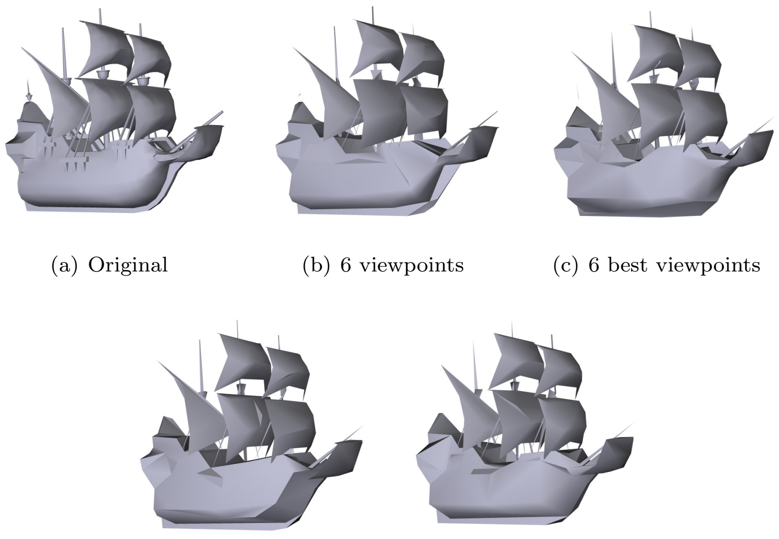
Model	**Triangles**		**Viewpoints**					
	Original	**Final**	**6**	**6 best**	**12**	**12 best**	**20**	**20 best**
Galleon	4698	500	27.27	19.17	18.16	17.77	17.06	17.98
Simpletree	11 136	600	18.93	18.28	17.12	17.73	16.27	16.69
Big_atc	13 594	1000	19.55	18.09	15.90	16.79	15.31	16.31

Table 7.2: Visual error (RMSE) for all models simplified with VMI, using the viewpoint distribution from the minimal set of viewpoints algorithm based on the heuristic measure (HM)

Model	**Triangles**		**Viewpoints**					
	Original	**Final**	**6**	**6 best**	**12**	**12 best**	**20**	**20 best**
Galleon	4698	500	28	30	52	54	83	83
Simpletree	11 136	600	113	110	196	196	331	332
Big_atc	13 594	1000	148	153	271	272	412	419

Table 7.3: Simplification time (seconds) for all models simplified with VMI, using the viewpoint distribution from the minimal set of viewpoints algorithm based on the heuristic measure (HM)

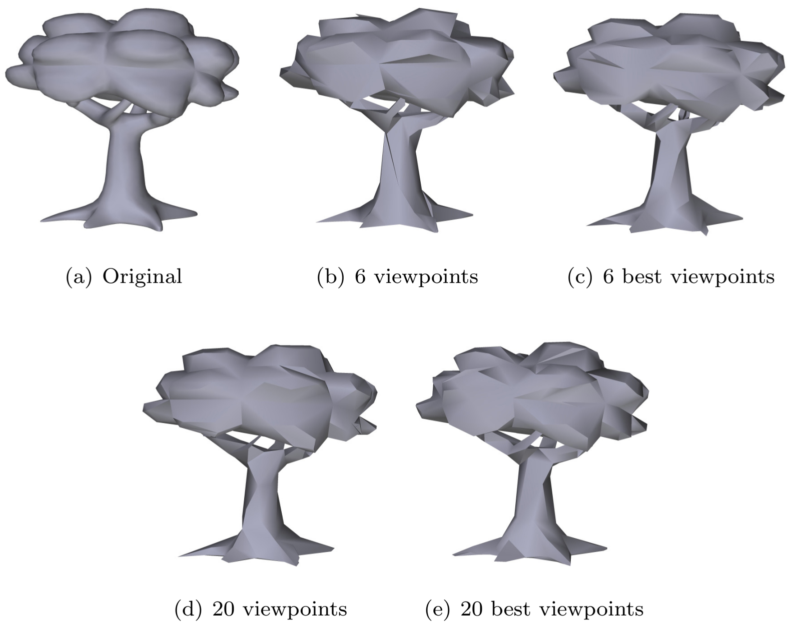
Table 7.2 depicts the visual error committed by the viewpoint-driven simplification algorithm using viewpoint mutual information in the simplified models and Table 7.3 shows the simplification time. As can be seen, the RMSE error is lower when using a minimal set of 6 best viewpoints than with 6 equidistant viewpoints. With 6 best viewpoints, a similar quality of 12 regularly distributed viewpoints can be accomplished. The temporal cost of the simplification algorithm using 6 best viewpoints is almost half. As mentioned earlier, the minimal set of best viewpoints is calculated once before applying the simplification algorithm. However, as we have demonstrated in previous chapters, having more than 20 regularly distributed viewpoints hardly improves the quality of the simplified models. Thus, if we use the same number of viewpoints but in this case best viewpoints, the quality of the resulting approximations will not be increased either.


(a) Original (b) 6 viewpoints (c) 6 best viewpoints

(d) 20 viewpoints (e) 20 best viewpoints

Figure 7.7: Different approximations of the Galleon model obtained with VMI, all at 500 triangles. The models shown in (b) and (d) were simplified using the regular viewpoint distribution and models (c) and (e) using the viewpoint distribution from the minimal set of viewpoints algorithm based on the heuristic measure (HM)

Figure 7.7 shows the Galleon model simplified at the same level but with different viewpoint distributions. As can be seen, there are not many visual differences among the approximations although the RMSE error is quite distinct. If we look closer perhaps we will see some small differences in the galleon’s hull. With this model, the best distribution of 6 viewpoints gives quite good



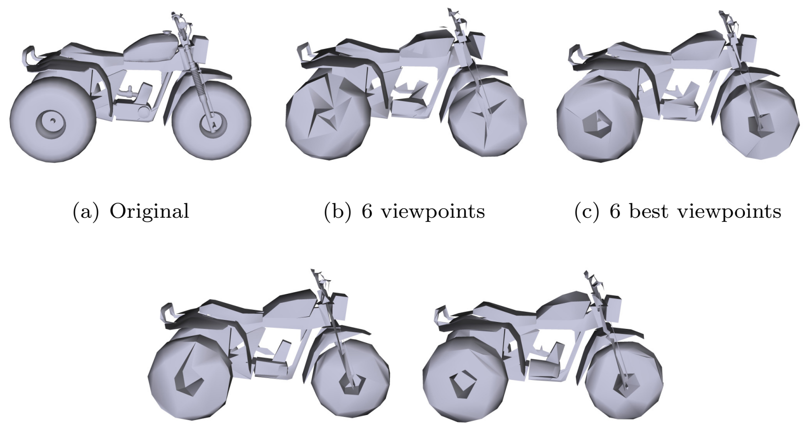
(a) Original (b) 6 viewpoints (c) 6 best viewpoints

(d) 20 viewpoints (e) 20 best viewpoints

Figure 7.8: Different approximations of the Simpletree model obtained with VMI, all at 600 triangles. The models shown in (b) and (d) were simplified using the regular viewpoint distribution and models (c) and (e) using the viewpoint distribution from the minimal set of viewpoints algorithm based on the heuristic measure (HM)

results. Figure 7.8 shows the Simpletree model. In this case, the trunk and the shape of the tree top are preserved slightly better with this distribution. The best distribution of 20 viewpoints does not seem to clearly improve the results compared to the regular one. Lastly, Figure 7.9 shows the Big_atc model. The best distribution of 6 viewpoints maintains the wheels much better than the regular one. However, in case of the configuration of 20 viewpoints, no clear benefit is gained when using the best distribution instead of the regular one.

Finally, as pointed out at the beginning of the chapter, we experimented with another algorithm for the selection of N best views proposed in [FSG06, FSG07]. This algorithm is based on *viewpoint mutual information.* It provided us with a different optimal distribution of viewpoints. Basically, the aim of this algorithm is to select the N viewpoints so that their merging $\hat{v}$ minimizes the viewpoint mutual information $I(\hat{v}, O)$. At each step, it attempts to maximize the JS-divergence between the set of previously merged viewpoints and the new viewpoint to be selected. Briefly, the algorithm proceeds as follows. First, the best viewpoint v_1 with probability distribution $p(O \mid v_1)$ corresponding to the


(a) Original (b) 6 viewpoints (c) 6 best viewpoints

(d) 20 viewpoints (e) 20 best viewpoints

Figure 7.9: Different approximations of the Big_atc model obtained with VMI, all at 1000 triangles. The models shown in (b) and (d) were simplified using the regular viewpoint distribution and models (c) and (e) using the viewpoint distribution from the minimal set of viewpoints algorithm based on the heuristic measure (HM)

minimum viewpoint mutual information $I(v, O)$ is selected. Next, v_2 is selected such that the mixed probability distribution $p(v_1)p(O \mid v_1) + p(v_2)p(O \mid v_2)$ will minimize $I(\hat{v}, O)$, where $\hat{v}$ represents the clustering of v_1 and v_2. At each step, a new mixed probability distribution $p(v_1)p(O \mid v_1) + p(v_2)p(O \mid v_2) + \cdots + p(v_n)p(O \mid v_n)$ is produced until the VMI-ratio given by $\frac{I(\hat{v},O)}{I(v,O)}$ is lower than a given threshold or a fixed number of views is achieved.

Table 7.4 shows the results using the selection algorithm based on VMI. We have tested the same viewpoint configurations, the regular distribution of 6, 12 and 20 viewpoints and the best distribution of 6, 12 and 20 viewpoints. As shown in this table, the best distribution improves the visual error more than the regular one only when a small number of viewpoints (6 viewpoints) is selected. These results are in agreement with the minimal set of viewpoints algorithm based on the heuristic measure (HM).

7.5 Conclusions

The quality of the viewpoints is an important factor in the viewpoint-driven simplification algorithm. As seen in this chapter, when a small number of viewpoints are used (for example, 6 viewpoints), the best viewpoint distribution can improve the resulting approximations as compared to the same number of

Model	Triangles		Viewpoints					
	Original	Final	6	6 best	12	12 best	20	20 best
Galleon	4698	500	27.27	23.19	18.16	23.21	17.06	17.47
Simpletree	11 136	600	18.93	17.96	17.12	17.91	16.27	18.38
Big_atc	13 594	1000	19.55	16.81	15.90	16.74	15.31	16.18

Table 7.4: Visual error (RMSE) for all models simplified with VMI, using the viewpoint distribution from the best view selection algorithm based on VMI

viewpoints with a regular distribution. This is because the best distribution receives more information from the object. However, if a regular viewpoint configuration with a large number of viewpoints (for instance, 20 viewpoints) is employed, the quality of the simplified meshes may not be enhanced when using the same number of good viewpoints. This is due to the fact that a dense uniformly distributed configuration completely covers the object and obtains enough information about its entire surface. Therefore, although the best distribution is chosen in this case, the results will probably not be easily improved.

By using a minimal set of good viewpoints, the temporal cost of the algorithm can be reduced and reasonable results in terms of quality can be achieved. This can be useful for low-end computers, which do not have expensive graphics cards, but also to accomplish affordable simplification times for very complex models.

CHAPTER 8

Conclusions and Future Work

8.1 Conclusions

In this thesis, we have studied the problem of automatic simplification of polygonal models. We have presented a new simplification method driven by viewpoint selection measures. These measures allow us to define a new error metric for polygonal simplification which is used to carry out the simplification process. Below, we review the concepts and the results achieved during the development of this dissertation:

- In Chapter 3, several techniques for computing the projected areas of polygons were analyzed. These areas are taken as probability distributions in the different viewpoint selection measures such as viewpoint entropy, mutual information and the Kullback-Leibler distance, used to quantify the error introduced by a simplification operation. The viewpoint-driven simplification method presented in this thesis is a greedy algorithm in which only one single simplification operation is performed at each step. Therefore, depending on the model and the level of simplification that is demanded, a large number of decimation operations are carried out. Thus, it is very important to find a technique that allows the simplification error metric to be computed quickly.

  We studied various techniques: the *OpenGL histogram*, the *hybrid software and hardware histogram* and the *occlusion query*. We found that the fastest technique in current hardware is the hybrid software and hardware histogram. This histogram analyzes the framebuffer pixel-by-pixel to retrieve information about its color. Thus, it depends on the image resolution and the performance of the bus. Nowadays, the symmetric PCI express bus allows a fast access, but we must take into consideration that the effectiveness of this technique is related to the size of the frame-buffer. The occlusion query does not depend on the image resolution and can be easily paralyzed. However, it has not been designed to deal with the thousands of queries needed to obtain the projected area of each individual polygon. The OpenGL histogram can be directly supported by the hardware in some graphics cards. Unfortunately, it has a clear disadvantage, i.e. the need to make several rendering passes due to the color assignment for the polygons. This makes the OpenGL histogram the worst technique of those analyzed here.

- In Chapter 4 the viewpoint-driven simplification method was presented. This new polygonal simplification scheme uses viewpoint selection measures to guide the simplification process and was designed to improve the visual fidelity of the approximations. The simplification error metric is defined by the variation in viewpoint selection measures for every viewpoint. Various uniformly distributed viewpoint configurations were employed to cover the whole object and consequently to achieve a regular simplification. This method can use any decimation operation. However, we apply the best half-edge collapse because this greedy operation makes it possible to reuse the simplification process in order to generate continuous multiresolution models. These models adjust the level of detail in real time depending on many factors, such as distance to the viewer.

  In this chapter, we studied the behavior of two Information-Theoretic viewpoint selection measures, namely *viewpoint entropy* and *viewpoint mutual information*. Both measures give excellent visual results when they are applied to the viewpoint-driven simplification algorithm. In many cases both of them can even enhance the geometric quality of the simplified models. Viewpoint mutual information represents an improvement over viewpoint entropy. In fact, it is able to preserve the silhouette of the models better than viewpoint entropy. The reason for this is that mutual information prefers to maintain the total visible area of the object rather than only balance the size of polygons as happens with viewpoint entropy. An important feature of both viewpoint entropy and mutual information is that both remove all hidden geometry in the earliest stages of simplification.

- In Chapter 5, the viewpoint-driven simplification algorithm is extended with several viewpoint f-divergences, such as the *viewpoint Kullback-Leibler distance*, the *viewpoint Chi-Square distance* and the *viewpoint Hellinger distance*. The f-divergences are convex functions that were introduced as measures of discrimination between probability distributions. The viewpoint Kullback-Leibler distance was proposed as a measure for viewpoint selection, and the other two defined here can also be used for this purpose. These divergences calculate the distance between the projected and the actual area distributions of the polygons in an object.

The application of these measures to the viewpoint-driven simplification algorithm considerably changes its results. By using the viewpoint f-divergences, the simplification method proposed in this thesis is able to produce simplified models which preserve hidden interiors and at the same time present a high degree of visual fidelity. Indeed, preserving hidden regions increases the geometric quality of the approximations. This characteristic is due to the fact that the actual areas of the polygons are considered. Among the different f-divergences tested, the Chi-Square distance gives the best results.

- In Chapter 6 we included a definition of mesh saliency based on JS-divergence in the viewpoint-driven simplification approach. This definition formulates *mesh saliency* in terms of how polygons see a set of viewpoints. Mesh saliency was introduced as a measure of regional importance for graphics meshes in order to capture the visually most interesting regions on a mesh.

  By incorporating mesh saliency into the simplification method, the most salient regions on a mesh are better preserved. However, mesh saliency may increase the global error because regions with low saliency are further simplified.

- In Chapter 7 we studied how viewpoint distribution affects the quality of the results in the viewpoint-driven simplification algorithm. We tested some uniformly distributed viewpoint configurations obtained from the vertices of some Platonic solids and the minimal set of best viewpoints using the same number of viewpoints. These minimal sets were calculated from *viewpoint mutual information* and the *heuristic measure*, although any other viewpoint selection measure presented in this thesis could also be used. The experiments show that with a small number of viewpoints, the minimal set of best viewpoints produces simplified meshes with a better quality than by using the same number of viewpoints distributed in a regular manner. This can be very useful, since the temporal cost of the algorithm is directly dependent on the number of viewpoints, so both distributions (regular and best) have the same cost. Nonetheless, there is a threshold beyond which using the best distribution does not enhance the quality of the simplified meshes. This threshold is reached at about 20 viewpoints.

8.2 Main contributions

The main contributions of this thesis are described below. In addition, the publications related to each contribution are also indicated.

- A study of several techniques to compute projected areas for the polygons in a scene or object. Most of the viewpoint selection measures used in this thesis are based on projected area distributions. This analysis therefore allowed these metrics to be computed quickly. [CRC04, CRC05, CCSF05, CSCF06]

- A new simplification method for polygonal meshes based on the viewpoint. This method uses the variation in viewpoint selection measures to quantify the error introduced by a decimation operation. Two viewpoint selection measures based on Information Theory, namely viewpoint entropy and mutual information, were tested. The simplification algorithm with these measures produces approximations with a high visual quality. Furthermore, it is able to remove all hidden geometry. [CSCF07c, CCSF07]

- The viewpoint-driven simplification algorithm was extended with several viewpoint f-divergences. In addition to the projected area distribution, these divergences also consider the actual area distribution and this fact completely changes the results of the simplification algorithm. With these divergences, the simplification algorithm is able to preserve hidden interiors and at the same time achieve a good visual quality. But perhaps the most relevant consequence is that the geometric error is substantially improved with respect to the previous selection measures (viewpoint entropy and mutual information). [CCSF07]

- A definition of mesh saliency based on JS-divergence was added to the viewpoint-driven simplification method. This notion of saliency was formulated according to visual perception. Mesh saliency allows salient features on the mesh to be preserved better.

- The effect of applying different viewpoint distributions in the simplification algorithm was also tested. We performed some trials with the regular distribution and the minimal set of best viewpoints distribution. The results of these experiments showed that when a reduced number of viewpoints are used, the best distribution can improve the quality of the simplified models. However, if an adequate number of viewpoints is used, that is, a number that is enough to entirely cover the object, the best distribution does not seem to enhance the results.

8.3 Publications

- Publications that support the contents of this thesis:

  "Explotación del hardware gráfico para acelerar la visualización de geometría"

Pascual Castelló, J. Francisco Ramos and Miguel Chover.
In *CEIG '04: XIV Congreso Español de Informática Gráfica, Seville, Spain*, pages 363–366, 2004.

"A comparative study of acceleration techniques for geometric visualization"
Pascual Castelló, J. Francisco Ramos and Miguel Chover.
In *ICCS '05: Proceedings of the International Conference on Computational Science (2), Atlanta, USA*, LNCS 3515, pages 240–247, 2005.

"Técnicas para calcular la entropía dependiente de la vista de una escena"
Pascual Castelló, Miguel Chover, Mateu Sbert and Miquel Feixas.
In *CEIG '05: XV Congreso Español de Informática Gráfica, Granada, Spain*, pages 277–280, 2005.

"Techniques for computing viewpoint entropy of a 3D scene"
Pascual Castelló, Mateu Sbert, Miguel Chover and Miquel Feixas.
In *ICCS '06: Proceedings of the International Conference on Computational Science (2), Reading, UK*, LNCS 3992, pages 263–270, 2006.

"Viewpoint entropy-driven simplification"
Pascual Castelló, Mateu Sbert, Miguel Chover and Miquel Feixas.
In *WSCG '07: Proc. of 15th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Plzen, Czech Republic*, volume 2, pages 249–256, 2007.

"Applications of Information Theory to Computer Graphics Part VII: Viewpoint-driven Simplification"
Pascual Castelló, Miguel Chover, Mateu Sbert and Miquel Feixas.
Eurographics Tutorial notes, 2007.

- Additional publications:

  "Simplificación de mallas para juegos"
  Carlos González, Jesús Gumbau, Miguel Chover and Pascual

Castelló.
In *CEIG '07: XVII Congreso Español de Informática Gráfica, Saragossa, Spain*, 2007.

"A texture-based metric extension for simplification methods"
Carlos González, Pascual Castelló and Miguel Chover.
In *Proc. of GRAPP 2007, Barcelona, Spain*, pages 69–77, 2007.

"Tiras de triángulos con niveles de detalle"
J. Francisco Ramos, Pascual Castelló and Miguel Chover.
In *CEIG '04: XIV Congreso Español de Informática Gráfica, Seville, Spain*, pages 339–342, 2004.

- This research was supported in part by the project

  "Advanced Tools for Developing Highly Realistic Computer Games"
  European Union, (project IST-2-004363), 2005- 2007.

- A list of submitted papers:

  "Viewpoint-driven simplification using Mutual Information"
  Pascual Castelló, Mateu Sbert, Miguel Chover and Miquel Feixas.
  Computers & Graphics, 2007. This paper offers a detailed explanation of the viewpoint-driven simplification algorithm. In addition, it analyzes the effect of applying viewpoint mutual information and compares the results obtained with mutual information to viewpoint entropy (Chapter 4). It is in second phase of revision.

  "Viewpoint-based simplification using f*-divergences"*
  Pascual Castelló, Mateu Sbert, Miguel Chover and Miquel Feixas.
  Information Sciences, 2007. This paper extends the viewpoint-driven simplification algorithm by including some viewpoint f-divergences (Chapter 5).

  "Polygonal simplification using View-based Saliency"
  Pascual Castelló, Mateu Sbert, Miguel Chover and Miquel Feixas.

In *WSCG '08: Proc. of 16th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Plzen, Czech Republic*, 2008. This paper applies mesh saliency based on Jensen-Shannon divergence in the viewpoint-driven simplification algorithm (Chapter 6).

8.4 Future Research

There are various potential areas for future research and applications of the simplification method proposed in this thesis. Below, we suggest and discuss some of them.

Better techniques for computing projected areas. As seen in previous chapters, the projected areas are used to define probability distributions. These distributions are the basis of most of the viewpoint selection measures introduced in this thesis. But by far the most significant fact is that the error metric of the simplification algorithm is directly dependent on these measures. Therefore, performing a fast computation of the projected areas would considerably reduce the temporal cost. In Chapter 3 we studied several techniques to obtain these areas. However, there is much room for improvement here. On the one hand, maybe the first idea that comes to mind is to paralyze the computation. Viewpoint selection measures are calculated from a set of viewpoints. By using a multi-graphics processor platform, each viewpoint could be rendered and analyzed independently. This solution is likely to be very expensive and complex. On the other hand, current graphics processing units (GPUs) offer a very powerful programming feature: *shaders*. These programs are written to apply transformations to a large set of elements at the same time, for example, to each pixel in an area of the screen, or for every vertex of a model. This is well suited to parallel processing and most modern GPUs have a multi-core design to facilitate this, which vastly improves the efficiency of processing. For instance, a possible solution for a fast calculation of projected areas using shaders might be as follows. First, we create a vertex for every polygon of the object, each one with a different color. Then, we render the object from the particular point of view to a texture. After that, we analyze the bounding box of each polygon in the texture with a pixel shader, counting the pixels that have the same vertex color associated to the polygon. In the end, we will obtain the projected area in each vertex.

Vertex placement heuristics. The viewpoint-driven simplification method measures the variation in the viewpoint selection measures before and after a decimation operation. This implies that any decimation operation can be performed. We applied the half-edge collapse as the decimation operation in the simplification algorithm. This may offer some advantages for multiresolution modeling and progressive transmission, since we do not introduce a new vertex after a simplification. As a consequence, it is possible to reuse the simplification sequence of half-edge collapses to develop multiresolution models that efficiently exploit current graphics hardware. In contrast, higher quality approximations can be accomplished with the most general edge collapse operation. In this case, the edge is collapsed into a new vertex that minimizes the local error. However, we need a vertex placement heuristic. We could use the quadric error metric to determine the position of the new vertex or some other different vertex placement strategy. A study of several strategies would be very useful.

Automatic selection of parameters. The quality of the results obtained with the simplification algorithm depends on various input parameters such as image resolution, number of viewpoints and viewpoint distribution. According to our experience, the default values normally work well. However, it would be very worthwhile studying different values and contrasting their effects on the results. In Chapter 7, we analyzed what repercussions the viewpoint distribution has on the simplification algorithm. In this case, we compared the minimal set of best viewpoints obtained with the heuristic measure to the regular distribution from the vertices of the Platonic solids. Here we suggest other algorithms based on different viewpoint selection measures to obtain this minimal set of good quality viewpoints. Finally, other aspects such as the image resolution should be explored. This fact is directly related to the temporal cost of the algorithm and might affect the quality of the simplified models.

Incorporating mesh attributes. Nowadays, in many real applications, such as computer games, models are represented not only by means of their geometric information. In order to appear more realistic, other information such as color, normals and texture is also used. Therefore, it would very useful to extend the viewpoint-driven simplification method in order to consider these mesh attributes. One possible way of doing this could be to extend the viewpoint selection measures with such attributes. Another might be, for instance in case of texture information, to add an error associated with the texture deformation to the cost of the decimation operation.

Other viewpoint selection measures. As pointed out previously, the simplification algorithm introduced in this thesis can use any viewpoint selection measure. So far, we have tested some of the most successful viewpoint selection metrics. However, we would like to conduct a study with other metrics based on Information Theory that can also be applied to the simplification framework. For example, we would like to analyze the extended mutual information of Tsallis [Tsa88], which depends on a parameter and contains, as a particular case, Shannon's mutual information. In Chapter 5 we extended the viewpoint-driven simplification algorithm with some f-divergences. Another family of divergences could also be used.

Including appearance attributes in mesh saliency. Mesh saliency based on JS-divergence only takes into consideration the geometric information, because it is computed from the projected areas of polygons. Generalizing mesh saliency to include other appearance attributes such as color, texture and reflectance would be an important area for future research.

Applications. In real environments, models are not isolated. Normally, objects are close and sometimes surrounded by others. For instance, imagine a tree just at the corner of two buildings. Most of its detail will be hidden. Only a part will be visible from many viewpoints. This case illustrates how static objects could be simplified depending on their position in the environment. We could put some viewpoints only in the visible part of the object and then perform a viewpoint-driven simplification. Accordingly, the viewpoint-driven simplification algorithm would produce approximations which will preserve the visible parts, the non-visible ones being simplified to a coarse level.

Bibliography

[Ale30] James W. Alexander. The combinatorial theory of complexes. *The Annals of Mathematics*, 31(2):292–320, may 1930.

[AS66] S. M. Ali and S. D. Silvey. A general class of coefficients of divergence of one distribution from another. *Royal Statistical Society Series B*, 28:131–142, 1966.

[BDP99] Pierre Barral, Guillaume Dorme, and Dimitri Plemenos. Visual understanding of a scene by automatic movement of a camera. In *GraphiCon*, 1999.

[Bla87] Richard E. Blahut. *Principles and practice of information theory*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1987.

[BPD00] Pierre Barral, Dimitri Plemenos, and Guillaume Dorme. Scene understanding techniques using a virtual camera. In *Eurographics*, 2000. Short paper.

[BR82] Jacob Burbea and C. Radhakrishna Rao. On the convexity of some divergence measures based on entropy functions. *IEEE Transactions on Information Theory*, 28(3):489–495, 1982.

[BS05] Udeepta Bordoloi and Han-Wei Shen. View selection for volume rendering. In *IEEE Visualization*, page 62, 2005.

[CCSF05] Pascual Castelló, Miguel Chover, Mateu Sbert, and Miquel Feixas. Técnicas para calcular la entropía dependiente de la vista de una escena. In *CEIG '05: XV Congreso Español de Informática Gráfica*, pages 277–280, 2005.

[CCSF07] Pascual Castelló, Miguel Chover, Mateu Sbert, and Miquel Feixas. Applications of information theory to computer graphics Part VII: Viewpoint-driven simplification. In *Eurographics Tutorial Notes*, Praque, Czech Republic, September 2007.

[Cla76] James H. Clark. Hierarchical geometric models for visible surface algorithms. *Commun. ACM*, 19(10):547–554, 1976.

[CMO97] Jonathan Cohen, Dinesh Manocha, and Marc Olano. Simplifying polygonal models using successive mappings. In *VIS '97: Proceedings of the 8th conference on Visualization*, pages 395–402. IEEE Computer Society Press, 1997.

[CMS98] P. Cignoni, C. Montani, and R. Scopigno. A comparison of mesh simplification algorithms. *Computers and Graphics*, 22(1):37–54, 1998.

[COM98] Jonathan Cohen, Marc Olano, and Dinesh Manocha. Appearance-preserving simplification. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 115–122, New York, NY, USA, 1998. ACM Press.

[CRC04] Pascual Castelló, J. Francisco Ramos, and Miguel Chover. Explotación del hardware gráfico para acelerar la visualización de geometría. In *CEIG '04: XIV Congreso Español de Informática Gráfica*, pages 363–366, 2004.

[CRC05] Pascual Castelló, J. Francisco Ramos, and Miguel Chover. A comparative study of acceleration techniques for geometric visualization. In *LNCS 3515, ICCS '05: Proceedings of International Conference on Computational Science (2)*, pages 240–247, 2005.

[CRS98] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring error on simplified surfaces. *Computer Graphics Forum*, 17(2):167–174, 1998.

[CSCF06] Pascual Castelló, Mateu Sbert, Miguel Chover, and Miquel Feixas. Techniques for computing viewpoint entropy of a 3d scene. In *LNCS 3992, ICCS '06: Proceedings of International Conference on Computational Science (2)*, pages 263–270, 2006.

[CSCF07a] Pascual Castelló, Mateu Sbert, Miguel Chover, and Miquel Feixas. Viewpoint-based simplification using f-divergences. *Information Sciences. Submitted*, 2007.

[CSCF07b] Pascual Castelló, Mateu Sbert, Miguel Chover, and Miquel Feixas. Viewpoint-driven simplification using mutual information. *Computers & Graphics. Submitted (Second phase of revision)*, 2007.

[CSCF07c] Pascual Castelló, Mateu Sbert, Miguel Chover, and Miquel Feixas. Viewpoint entropy-driven simplification. In *WSCG '07:*

Proc. of 15th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, volume 2, pages 249–256, 2007.

[CSCF08] Pascual Castelló, Mateu Sbert, Miguel Chover, and Miquel Feixas. Polygonal simplification using view-based saliency. In *WSCG '08: Proc. of 16th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision. Submitted*, 2008.

[Csi63] Imre Csiszár. Eine Informationsheoretische Ungleichung und ihre Anwendungen auf den Beweis der ergodizität von Markoffschen Ketten. *Magyar Tudományos Akadémia Közleményei*, 8:85–108, 1963.

[CT91] Thomas M. Cover and Joy A. Thomas. *Elements of information theory.* Wiley-Interscience, New York, NY, USA, 1991.

[CVM+96] Jonathan Cohen, Amitabh Varshney, Dinesh Manocha, Greg Turk, Hans Weber, Pankaj Agarwal, Frederick Brooks, and William Wright. Simplification envelopes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 119–128. ACM Press, 1996.

[Dra00] Sever S. Dragomir. Some inequalities for the Csiszár f-divergence. In *Inequalities for Csiszár f-Divergence in Information Theory. Research Group in Mathematical Inequalities and Applications, Victoria University of Technology, Melbourne, Australia*, 2000.

[DV00] Minh N. Do and Martin Vetterli. Texture similarity measurement using Kullback-Leibler distance on wavelet subbands. In *Proc. IEEE Int. Conf. on Image Proc. (ICIP), (Vancouver, Canada)*, September 2000.

[EDD+95] Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Michael Lounsbery, and Werner Stuetzle. Multiresolution analysis of arbitrary meshes. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 173–182. ACM Press, 1995.

[EM99] Carl Erikson and Dinesh Manocha. Gaps: general and automatic polygonal simplification. In *SI3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics*, pages 79–88, New York, NY, USA, 1999. ACM Press.

[FdABS99a] Miquel Feixas, Esteve del Acebo, Philippe Bekaert, and Mateu Sbert. An information theory framework for the analysis of scene complexity. In P. Brunet and R. Scopigno, editors, *Computer Graphics Forum (Eurographics '99)*, volume 18(3), pages 95–106. The Eurographics Association and Blackwell Publishers, 1999.

[FdABS99b] Miquel Feixas, Esteve del Acebo, Philippe Bekaert, and Mateu Sbert. Information theory tools for scene discretization. In *Rendering Techniques '99*, pages 95–106, New York, NY, 1999. Springer Wien.

[Fei02] Miquel Feixas. *An Information-Theory Framework for the Study of the Complexity of Visibility and Radiosity in a Scene*. PhD thesis, Universitat Politècnica de Catalunya, October 2002.

[FMP97] Leila De Floriani, Paola Magillo, and Enrico Puppo. Building and traversing a surface at variable resolution. In *VIS '97: Proceedings of the 8th conference on Visualization*, pages 103–ff., Los Alamitos, CA, USA, 1997. IEEE Computer Society Press.

[FMP98] Leila De Floriani, Paola Magillo, and Enrico Puppo. Efficient implementation of multi-triangulations. In *VIS '98: Proceedings of the conference on Visualization*, pages 43–50. IEEE Computer Society Press, 1998.

[FSG06] Miquel Feixas, Mateu Sbert, and Francisco González. A unified information-theoretic framework for viewpoint selection and mesh saliency. Technical Report IIiA-06-06-RR, Institut d'Informàtica i Aplicacions, Universitat de Girona (Girona, Spain), 2006.

[FSG07] Miquel Feixas, Mateu Sbert, and Francisco González. A unified information-theoretic framework for viewpoint selection and mesh saliency. Technical Report IIiA-07-03-RR, Institut d'Informàtica i Aplicacions, Universitat de Girona (Girona, Spain) Extended version of IIiA-06-06-RR., 2007.

[GAG04] Carlos Andújar Gran, Pere P. Vázquez Alcocer, and Marta Fairén González. Way-finder: Guided tours through complex walkthrough models. *Computer Graphics Forum*, 23(3):499–508, 2004.

[Gar99] Michael Garland. Multiresolution modeling: Survey & future opportunities. *State of the Art Reports of EUROGRAPHICS'99*, 14(4):111–131, 1999.

[GCC07] Carlos González, Pascual Castelló, and Miguel Chover. A texture-based metric extension for simplification methods. In *Proc. of GRAPP 2007, Barcelona, Spain*, pages 69–77, 2007.

[GCO06] Ran Gal and Daniel Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Trans. Graph.*, 25(1):130–150, 2006.

[GDA98] Ahmed Ghali, M. Farhang Daemi, and K. A. Al-Khateeb. Information-based image dissimilarity measure. *Optical Engineering*, 37(3):808–812, March 1998.

[GDBA94] A. Ghali, M. F. Daemi, R. L. Beurle, and K. A. Al-Khateeb. Pattern recognition based on information theory principles. In A. G. Tescher, editor, *Proc. SPIE Vol. 2298, Applications of Digital Image Processing XVII*, pages 562–572, September 1994.

[GH97] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216. ACM Press/Addison-Wesley Publishing Co., 1997.

[GH98] Michael Garland and Paul S. Heckbert. Simplifying surfaces with color and texture using quadric error metrics. In *VIS '98: Proceedings of the conference on Visualization 1998*, pages 263–269, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[GZ05] Michael Garland and Yuan Zhou. Quadric-based simplification in any dimension. *ACM Trans. Graph.*, 24(2):209–239, 2005.

[HBK02] Masauki Hisada, Alexander Belyaev, and Tosiyasu L. Kunii. A skeleton-based approach for detection of perceptually salient features on polygonal surfaces. *Computer Graphics Forum*, 21(4):1–12, 2002.

[HDD$^+$93] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 19–26. ACM Press, 1993.

[Hel09] Ernst D. Hellinger. Neue Begründung der Theorie quadratischen Formen von unendlichen vielen Veränderlichen. *Journal für Reine und Angewandte Mathematik*, 136:210–271, 1909.

[HG97] Paul S. Heckbert and Michael Garland. Survey of polygonal surface simplification algorithms. Technical report, Multiresolution Surface Modeling Course Notes of SIGGRAPH'97, 1997.

[HHK$^+$95] Taosong He, Lichan Hong, Arie Kaufman, Amitabh Varshney, and Sidney Wang. Voxel based object simplification. In *VIS '95: Proceedings of the 6th conference on Visualization*, page 296, Washington, DC, USA, 1995. IEEE Computer Society.

[Hop96] Hugues Hoppe. Progressive meshes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and* *interactive techniques*, pages 99–108, New York, NY, USA, 1996. ACM.

[Hop97] Hugues Hoppe. View-dependent refinement of progressive meshes. *Computer Graphics*, 31(Annual Conference Series):189–198, 1997.

[Hop98] Hugues Hoppe. Efficient implementation of progressive meshes. *Computers & Graphics*, 22(1):27–36, 1998.

[Hop99] Hugues Hoppe. New quadric metric for simplifying meshes with appearance attributes. In *VIS '99: Proceedings of the 10th IEEE Visualization 1999 Conference*, Washington, DC, USA, 1999. IEEE Computer Society.

[ILGS03] Martin Isenburg, Peter Lindstrom, Stefan Gumhold, and Jack Snoeyink. Large mesh simplification using processing sequences. In *VIS '03: Proceedings of the 14th IEEE Visualization*, page 61, Washington, DC, USA, 2003. IEEE Computer Society.

[IVA$^+$96] William M. Wells III, Paul Viola, Hideki Atsumi, Shin Nakajima, and Ron Kikinis. Multi-modal volume registration by maximization of mutual information. *Medical Image Analysis*, 1(1):35–51, 1996.

[JPS99] Vincent Jolivet, Dimitri Plemenos, and Mateu Sbert. A pyramidal hemisphere subdivision method for monte carlo radiosity. In *Computer Graphics Forum (Proc. Eurographics '99)*, pages 63–66, September 1999.

[JS06] Guangfeng Ji and Han-Wei Shen. Dynamic view selection for time-varying volumes. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1109–1116, 2006.

[JTY06] Bin-Shyan Jong, Juin-Ling Tseng, and Wen-Hao Yang. An efficient and low-error mesh simplification method based on torsion detection. *The Visual Computer*, 22:56–67, 2006.

[JWRS06] Justin Jang, Peter Wonka, William Ribarsky, and Christopher D. Shaw. Punctuated simplification of man-made objects. *Vis. Comput.*, 22(2):136–145, 2006.

[KB99] Vera Kettnaker and Matthew Brand. Minimum-entropy models of scene activity. In *CVPR*, pages 1281–1286, 1999.

[KCS98] Leif Kobbelt, Swen Campagna, and Hans-Peter Seidel. A general framework for mesh decimation. In *Graphics Interface*, pages 43–50, 1998.

[KG00] Zachi Karni and Craig Gotsman. Spectral compression of mesh geometry. In *SIGGRAPH ’00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 279–286, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[KG03] Youngihn Kho and Michael Garland. User-guided simplification. In *SI3D ’03: Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 123–126, New York, NY, USA, 2003. ACM Press.

[KL51] Solomon Kullback and Richard A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:76–86, 1951.

[KTiKN05] Takayuki Kanaya, Yuji Teshima, Ken ichi Kobori, and Koji Nishio. A topology-preserving polygonal simplification using vertex clustering. In *GRAPHITE ’05: Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 117–120, New York, NY, USA, 2005. ACM Press.

[KV06] Youngmin Kim and Amitabh Varshney. Saliency-guided enhancement for volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):925–932, 2006.

[LC87] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH ’87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169, New York, NY, USA, 1987. ACM Press.

[LE97] David Luebke and Carl Erikson. View-dependent simplification of arbitrary polygonal environments. In *SIGGRAPH ’97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 199–208. ACM Press/Addison-Wesley Publishing Co., 1997.

[LH01] David Luebke and Benjamin Hallen. Perceptually-driven simplification for interactive rendering. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 223–234, London, UK, 2001. Springer-Verlag.

[Lin00] Peter Lindstrom. Out-of-core simplification of large polygonal models. In *SIGGRAPH ’00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 259–262. ACM Press/Addison-Wesley Publishing Co., 2000.

[LP01] Peter Lindstrom and Valerio Pascucci. Visualization of large terrains made easy. In *IEEE Visualization*, 2001.

[LRC$^+$04] David Luebke, Martin Reddy, Jonathan D. Cohen, Amitabh Varshney, Benjamin Watson, and Robert Huebner. Level of detail for 3d graphics. In Brian A. Barsky, editor, *Robust Computer Vision: Quality of Vision Algorithms*, pages 1–26. Elsevier Science, USA, 2004.

[LT97] Kok-Lim Low and Tiow-Seng Tan. Model simplification using vertex-clustering. In *SI3D ’97: Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 75–82, New York, NY, USA, 1997. ACM Press.

[LT00] Peter Lindstrom and Greg Turk. Image-driven simplification. *ACM Transaction Graphics*, 19(3):204–241, 2000.

[Lue01] David Luebke. A developer’s survey of polygonal simplification algorithms. *IEEE Computer Graphics Application*, 21(3):24–35, 2001.

[LVJ05] Chang Ha Lee, Amitabh Varshney, and David W. Jacobs. Mesh saliency. In *SIGGRAPH ’05: ACM SIGGRAPH 2005 Papers*, pages 659–666, New York, NY, USA, 2005. ACM Press.

[Mel98] Stan Melax. A simple, fast, and effective polygon reduction algorithm. *Game Developer*, pages 44–48, November 1998.

[PB96] Dimitri Plemenos and Madjid Benayada. Intelligent display in scene modeling. new techniques to automatically compute good views. In *International Conference GraphiCon*, July 1996.

[Pea00] Karl Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Psychology Magazine*, 1:157–175, 1900.

[PGJT05] Dimitri Plemenos, Jérôme Grasset, Benoît Jaubert, and Karim Tamine. Intelligent visibility-based 3d scene processing techniques for computer games. In *GraphiCon*, 2005. STAR Report.

[Ple03] Dimitri Plemenos. Exploring virtual worlds: current techniques and future issues. In *GraphiCon*, 2003. STAR Report.

[Plu01] Josien P. W. Pluim. *Mutual Information based registration of medical images*. PhD thesis, University Medical Center Utrecht (The Netherlands), 2001.

[PSF04] Dimitri Plemenos, Mateu Sbert, and Miquel Feixas. On viewpoint complexity of 3d scenes. In *GraphiCon*, 2004. STAR Report.

[RB93] Jarek Rossignac and Paul Borrel. Multi-resolution 3d approximations for rendering complex scenes. In B. Falcidieno and T. Kunii, editors, *Modeling in Computer Graphics: Methods and Applications*, pages 455–465. Springer-Verlag, 1993.

[RFS00] Jaume Rigau, Miquel Feixas, and Mateu Sbert. Information theory point measures in a scene. Technical Report IIiA-00-08-RR, Institut d'Informàtica i Aplicacions, Universitat de Girona, 2000.

[RFS03] Jaume Rigau, Miquel Feixas, and Mateu Sbert. Entropy-based adaptive sampling. In *Graphics Interface*, pages 149–158, 2003.

[RFS05] Jaume Rigau, Miquel Feixas, and Mateu Sbert. Shape complexity based on mutual information. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications*, pages 357–362, Washington, DC, USA, 2005. IEEE Computer Society.

[RLB$^+$02] José Ribelles, Angeles López, Oscar Belmonte, Inmaculada Remolar, and Miguel Chover. Multiresolution modeling of arbitrary polygonal surfaces: a characterization. *Computers & Graphics*, 26(3):449–462, 2002.

[Ros01] Jarek Rossignac. 3d compression made simple: Edgebreaker with zip&wrap on a corner-table. In *SMI '01: Proceedings of the International Conference on Shape Modeling & Applications*, page 278, Washington, DC, USA, 2001. IEEE Computer Society.

[SA06] Mark Segal and Kurt Akeley. *The OpenGL Graphics System: A Specification (Version 2.1 - December 1, 2006).* Silicon Graphics, Inc., 2006.

[Sch97] William J. Schroeder. A topology modifying progressive decimation algorithm. In *VIS '97: Proceedings of the 8th conference on Visualization*, pages 205–ff., Los Alamitos, CA, USA, 1997. IEEE Computer Society Press.

[SFR$^+$02] Mateu Sbert, Miquel Feixas, Jaume Rigau, Francesc Castro, and Pere P. Vázquez. Applications of information theory to computer graphics. In *Proceedings of 5th International Conference on Computer Graphics and Artificial Intelligence, 3IA'02*, pages 21–36, 2002.

[SG01] Eric Shaffer and Michael Garland. Efficient adaptive simplification of massive meshes. In *VIS '01: Proceedings of the conference* *on Visualization*, pages 127–134, Washington, DC, USA, 2001. IEEE Computer Society.

[Sha48] Claude E. Shannon. *A Mathematical Theory of Communication.* CSLI Publications, 1948.

[SPFG05] Mateu Sbert, Dimitri Plemenos, Miquel Feixas, and Francisco González. Viewpoint quality: Measures and applications. In *Computational Aesthetics in Graphics, Visualization and Imaging*, 2005.

[ST00] Noam Slonim and Naftali Tishby. Document clustering using word clusters via the information bottleneck method. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 208–215, New York, NY, USA, 2000. ACM Press.

[Stu97] Colin Studholme. *Measures of 3D Medical Image Alignment.* PhD thesis, University of London, August 1997.

[SZL92] William J. Schroeder, Jonathan A. Zarge, and William E. Lorensen. Decimation of triangle meshes. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 65–70, New York, NY, USA, 1992. ACM Press.

[TFTN05] Shigeo Takahashi, Issei Fujishiro, Yuriko Takeshima, and Tomoyuki Nishita. A feature-driven approach to locating optimal viewpoints for volume visualization. In *IEEE Visualization*, page 63, 2005.

[TOS98] Yoshinori Takeuchi, Noboru Ohnishi, and Noboru Sugie. Active vision system based on information theory. *Systems and Computers in Japan*, 29(11):31–39, 1998.

[Tsa88] Constantino Tsallis. Possible generalization of Boltzmann-Gibbs statistics. *J. Stat. Phys.*, 52:479–487, 1988.

[Váz03] Pere P. Vázquez. *On the Selection of Good Views and its Application to Computer Graphics.* PhD thesis, Universitat Politècnica de Catalunya, April 2003.

[VC07] Huy T. Vo and Steven P. Callahan. Streaming simplification of tetrahedral meshes. *IEEE Transactions on Visualization and Computer Graphics*, 13(1):145–155, 2007. Member-Peter Lindstrom and Member-Valerio Pascucci and Member-Claudio T. Silva.

[vdL97] Jan C. A. van der Lubbe. *Information theory.* Cambridge University Press, New York, NY, USA, 1997. Translator-Hendrik Jan Hoeue and Translator-Steve Gee.

[Vel01] Luiz Velho. Mesh simplification using four-face clusters. In *SMI '01: Proceedings of the International Conference on Shape Modeling & Applications*, page 200, Washington, DC, USA, 2001. IEEE Computer Society.

[VFSG06] Ivan Viola, Miquel Feixas, Mateu Sbert, and Meister Eduard Gröller. Importance-driven focus of attention. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):933–940, October 2006.

[VFSH01] Pere P. Vázquez, Miquel Feixas, Mateu Sbert, and Wolfgang Heidrich. Viewpoint selection using viewpoint entropy. In *VMV '01: Proceedings of the Vision Modeling and Visualization Conference*, pages 273–280. Aka GmbH, 2001.

[VFSH03] Pere P. Vázquez, Miquel Feixas, Mateu Sbert, and Wolfgang Heidrich. Automatic view selection using viewpoint entropy and its application to image-based modelling. *Computer Graphics Forum*, 22(4):689–700, 2003.

[VFSL02] Pere P. Vázquez, Miquel Feixas, Mateu Sbert, and Antoni Llobet. Viewpoint entropy: A new tool for obtaining good views for molecules. In *Data Visualisation 2002 (Eurographics/IEEE TCVG Symposium Proceedings)*, pages 183–188, 2002.

[VFSL04] Pere P. Vázquez, Miquel Feixas, Mateu Sbert, and Antoni Llobet. On the fly best view detection using graphics hardware. In *VIIP 2004: Proceedings of 4th IASTED International Conference on Visualization, Imaging, and Image Processing*, August 2004.

[VFSL06] Pere P. Vázquez, Miquel Feixas, Mateu Sbert, and Antoni Llobet. Realtime automatic selection of good molecular views. *Computers & Graphics*, 30(1):98–110, February 2006.

[vKP06] Oliver Matias van Kaick and Helio Pedrini. A comparative evaluation of metrics for fast mesh simplification. *Computer Graphics Forum*, 25:197–210(14), June 2006.

[VLV$^+$04] Antonio W. Vieira, Thomas Lewinera, Luiz Velho, Helio Lopes, and Geovan Tavares. Stellar mesh simplification using probabilistic optimization. *Computer Graphics Forum*, 23(4):825838, 2004.

[VS02] Pere P. Vázquez and Mateu Sbert. Automatic keyframe selection techniques for high-quality image-based walkthrough animation using viewpoint entropy. *Journal of WSCG*, 10(1):461–468, 2002.

[VS03] Pere P. Vázquez and Mateu Sbert. Automatic indoor scene exploration. In *Proc. of 6th International Conference on Computer Graphics and Artificial Intelligence*, pages 13–24, 2003.

[VWMW97] Paul Viola and III William M. Wells. Alignment by maximization of mutual information. *Int. J. Comput. Vision*, 24(2):137–154, 1997.

[WB01] Kouki Watanabe and Alexander G. Belyaev. Detection of salient curvature features on polygonal surfaces. *Comput. Graph. Forum*, 20(3), 2001.

[WHC04] Yong Wu, Yuanjun He, and Hongming Cai. Qem-based mesh simplification with global geometry features preserved. In *GRAPHITE ’04: Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 50–57, New York, NY, USA, 2004. ACM Press.

[WK02] Jianhua Wu and Leif Kobbelt. Fast mesh decimation by multiple-choice techniques. In *VMV 2002: Proceedings of the Vision, Modeling, and Visualization Conference*, pages 241–248, 2002.

[WK03] Jianhua Wu and Leif Kobbelt. A stream algorithm for the decimation of massive meshes. In *Graphics Interface*, pages 185–192, 2003.

[WLC$^+$03] Nathaniel Williams, David Luebke, Jonathan D. Cohen, Michael Kelley, and Brenden Schubert. Perceptually guided simplification of lit, textured meshes. In *SI3D ’03: Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 113–121, New York, NY, USA, 2003. ACM Press.

[WST03] Adam Wilen, Justin Schade, and Ron Thornburg. *Introduction to PCI Express. A Hardware and Software Developer’s Guide.* Intel Press, 2003.

[YSZ04] Jingqi Yan, Pengfei Shi, and David Zhang. Mesh simplification with hierarchical shape analysis and iterative edge contraction. *IEEE Transactions on Visualization and Computer Graphics*, 10(2):142–151, 2004.

[ZT02] Eugene Zhang and Greg Turk. Visibility-guided simplification. In *VIS ’02: Proceedings of the conference on Visualization 2002*, pages 267–274, Washington, DC, USA, 2002. IEEE Computer Society.