UNIVERSITAT POLITÈCNICA DE CATALUNYA

Programa de Doctorado:
Automatización Avanzada y Robótica

Tesis doctoral

Asistencia háptica basada en planificación de movimientos para la teleoperación cooperativa de sistemas multirobot

Alexander Pérez Ruiz

Director: Dr. Jan Rosell Gratacòs

Instituto de Organización y Control de Sistema Industriales
Barcelona, 2012

Tesis doctoral

Asistencia háptica basada en planificación de movimientos para la teleoperación cooperativa de sistemas multirobot

por

Alexander Pérez Ruiz

Director: Dr. Jan Rosell Gratacòs

Enviado al Instituto de Organización y Control de Sistema Industriales
como requisito parcial al título de

Doctor

por la

Universitat Politècnica de Catalunya

Barcelona, 2012

a esos bellos ojos,
que entre molinos y suecos
me dieron la alegría
de conocer a Sebastián...

a mis padres,
que con silencioso esfuerzo
construyeron las alas,
con las que alcé el vuelo...

Agradecimientos

No quiero dejar pasar esta oportunidad sin agradecer a todas y cada una de las personas que han contribuido en mayor o menor escala durante todo este proceso vivido en los últimos 6 años.

En primer lugar a mi esposa Gina Paola, por el apoyo y la compañia que me ha brindado, y sobre todo por haber querído tomar este riesgo conmingo, que nos ha permitido crecer como personas y como pareja, apostando por esta vida juntos que ahora compartimos con nuestro hijo Sebastián.

A mis padres Luis Enrique y Concepción, que con todo su cariño y dedicación me han mostrado un camino de superación que me ha inspirado a lo largo de mi vida, y a mi hermano William Daniel, quien siempre tiene una mano para ayudarme en los momentos que más lo he necesitado. Gracias por estár siempre allí. Al resto de mi familia por tenernos siempre tan presentes a pesar de la distancia.

A mi tutor y director de tesis Jan Rosell Gratacòs por su invaluable ayuda y apoyo en todo este proceso de aprendizaje de la labor investigadora y al Profesor Luis Basañez y al Director del Instituto Raúl Suarez por compartir su saber y qué hacer.

A la Agència de Gestió d'Ajuts Universitaris i de Recerca (AGAUR) por la beca que me ha permitido finalizar este proceso.

A todos los integrantes del Instituto de Organización y Control de Sistemas Industriales (IOC) y en especial al grupo de Robotica Industrial y de Servicio por su acogida. No hubiera sido igual sin las fructiferas conversaciones a la hora de la comida con Carlos Aldana, Leopold Palomo, Paolo

Cabras y demás integrantes de la *IOCBand*.

A todos los amigos que, a lo largo de estos años, hemos ido conociendo aquí y muy en especial a Margarida Julià-Sapé, por su interesante e inagotable punto de vista sobre la vida.

A mis incondicionales Henry Córdoba, Diego Ramos, Willington Ortíz y César Camargo, por mantener los lazos de una gran amistad que trasciende el tiempo y el espacio.

Mis más sinceros agradecimientos por cada grano de arena aportado, que ha hecho posible construir todo esto.

Resumen

En el presente trabajo se propone un marco para la teleoperación de robots industriales antropomórficos que cuenta con elementos de ayuda al operador en forma de guiado háptico basado en técnicas de planificación de caminos (*path planning*) para facilitarle su labor. Incorpora un novedoso modelo de correspondencia entre el espacio de trabajo del dispositivo háptico con el que va a ser comandado y el del robot. Esta correspondencia tiene en cuenta la posición y orientación de la cámara activa que proporciona el video desde el sitio remoto. Las principales características de este sistema son:

1. El usuario puede cambiar de cámara activa para incrementar la visibilidad o cambiar las escalas que relacionan los espacios de trabajo para ajustar la precisión.

2. El usuario experimenta ligeras fuerzas de guiado que lo atraen hacia el camino planificado y también lo empujan a lo largo del mismo hacia la configuración objetivo, en particular cuando contiene cambios de configuración.

3. Cuando se alcanzan los limites del espacio de trabajo del dispositivo háptico o cuando el usuario lo solicita, se calcula la nueva configuración de este dispositivo, desde donde pueda retomar y efectuar la teleoperación y lo guía hasta ella.

Las ayudas hápticas son generadas a partir de un camino que soluciona la tarea. Para lo cual se ha desarrollado un planificador que incorpora un

novedoso algoritmo que es capaz de encontrar caminos realizables tanto entre obstáculos estáticos como entre móviles o la mezcla de unos y otros. Las principales características del planificador propuesto en el presente trabajo son:

1. Está basado en un planificador de mapa de carreteras probabilístico (*Probabilistic Roadmap*) con evaluación tardía de las colisiones del camino encontrado.

2. Se reduce la cantidad de muestras que componen el grafo donde se busca la solución a partir de uno más grande que cubre el espacio de configuraciones.

3. Cada camino planificado es evaluado y mejorado progresivamente de forma iterativa, creando un nuevo conjunto de muestras cerca de los obstáculos cuando la validación de alguna arista falla.

4. Es útil para replanificar cuando están presentes obstáculos móviles o cuando la configuración inicial cambia, dando siempre como resultado un camino muy cercano al anteriormente válido.

Con esta ayuda el usuario se libera de prestar atención a las potenciales colisiones de cualquier parte del robot con el entorno y no solamente del efector final del robot (TCP) como se hace en otros enfoques. Esto es aplicado especialmente en entornos multirobot donde los otros robots presentes en la celda remota pueden interferir con los movimientos teleoperados y que en este trabajo son considerados como colecciones de obstáculos móviles.

El esquema de control del sistema de teleoperación bilateral, garantiza la estabilidad a pesar de los retardos de las variables. Adicionalmente, se influye sobre su factor de amortiguamiento en el nodo remoto como una forma de hacer frente a potenciales colisiones, ralentizando la teleoperación cuando se advierte una potencial colisión.

El sistema ha sido validado con varios usuarios, tanto en simulación como en experimentación con el robot real, mostrando que con esta asistencia se incrementa la velocidad y la seguridad en la ejecución de la tarea y se aligera la carga de la teleoperación, en especial cuando existen obstáculos móviles.

Índice general

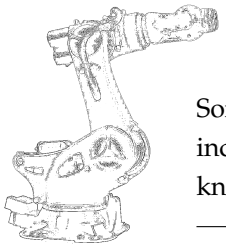
Índice de figuras

Índice de tablas

Índice de algoritmos

1

Introducción



Somewhere, something incredible is waiting to be known.

Carl Sagan

La teleoperación de robots mediante dispositivos hápticos permite, por un lado, reflejar al operador las fuerzas que siente el robot y, por el otro, el uso de fuerzas de guiado para ayudar al operador en la teleoperación. Este es actualmente un campo en continuo desarrollo, y más aún en el marco de Internet.

En esta tesis se explora la planificación de movimientos para la generación de un campo de fuerzas de guiado que brinda soporte en las tareas de teleoperación en las que el entorno puede ser dinámico. Esto requiere tanto una adaptabilidad en tiempo real del campo de fuerzas de guiado, cómo una respuesta reactiva del robot en el entorno remoto basada en información sensorial.

Esta asistencia a la teleoperación se extiende en el marco de sistemas multioperador-multirobot que permiten llevar a cabo tareas donde los robots comparten, al menos en parte, el espacio de trabajo. Los algoritmos y métodos propuestos se han desarrollado de forma que pueden ser utilizados también en el ámbito de la enseñanza de la planificación de movimientos.

1.1. Motivación

El uso de robots articulados fuera de los entornos industriales ha venido creciendo de forma paulatina en las últimas décadas y su uso dentro del marco de la teleoperación los hace idóneos para trabajos donde la pericia y experiencia humana en el desarrollo de las tareas resulta imprescindible.

En el marco de la teleoperación se pueden abstraer dos sistemas bien diferenciados que la hacen posible: el mando y el robot. Por un lado se encuentra el subsistema de mando equipado con un dispositivo que permite controlar el robot y que también puede realimentar al operador información proveniente de él. Y por otro lado se encuentra el subsistema robot que realizará los movimientos comandados desde el subsistema mando. Cada uno de ellos posee los módulos necesarios tanto de software como de hardware que hacen posible su correcto funcionamiento. De forma genérica se llamará nodo a cada uno de estos subsistemas:

- En el nodo de mando, el elemento maestro de control es un dispositivo háptico con realimentación de fuerza que le permitirá "sentir" al operador las particularidades del proceso de teleoperación como la facilidad o dificultad de realizar un movimiento y las interacciones con el entorno impuestas por la tarea.

- En el nodo de operación, se encuentra el/los robot y todos los módulos de control y adquisición de datos que son necesarios para cumplir de forma efectiva y segura los movimientos propios del mismo. Dentro de los sistemas de adquisición de información, ocupan un papel preponderante las cámaras que realimentan las imagenes de los movimientos del (de los) robot(s) presentes.

Para lograr la interacción conjunta de los dos nodos, existe un tercer componente o módulo de comunicaciones, que se encarga de intercambiar la información de forma correcta, fiable y oportuna sobre la red que se establezca como medio y diseñado para ser usado primordialmente sobre Internet.

En este marco de teleoperación enfocado a la teletarea, surge la necesidad de proveer de ayuda al teleoperador con un sistema que le permita evitar colisiones del brazo robot con el entorno sin que deba aumentar su

atención sobre las particularidades del mecanismo y a la vez permitir una mejora en su desempeño durante la ejecución de la tarea propuesta.

1.2. Descripción del problema

En los procesos de teleoperación, la selección de la información que se realimenta al operador sobre la ejecución de la tarea y la forma de hacerlo, es una parte que reviste gran importancia. Cuando se quiere hacer énfasis en los movimientos del objeto manipulado en lugar de las similitudes y/o diferencias tanto cinemáticas como dinámicas que existen entre el mando y el robot teleoperado, se esta definiendo la "teletarea", en el que se pretende hacer transparente para el operador todas las particularidades físicas del robot teleoperado y con ello centrar toda su atención en la culminación de la tarea.

La definición de la tarea de manipulación consiste en mover de un punto a otro en el espacio el efector final del robot, siguiendo una trayectoria libre de colisión en el tiempo que dure la teleoperación. Además de los obstáculos reales, pueden definirse algunos obstáculos virtuales que actúen como restricciones de índole geométrico de forma que se definen zonas vedadas al planificador y que luego serán sugeridas al operador para ser eludidas. La tarea se realiza siempre con el soporte de un sistema de guiado háptico, que apoyándose en técnicas de planificación de trayectorias y movimientos, proporciona un campo de fuerzas virtuales idóneo para completarla satisfactoriamente. En la Figura 1.1 se puede apreciar el esquema general del proyecto en el cual se enmarca el presente trabajo, donde los bloques resaltados de color naranja son los objetos principales que aquí se analizan. En particular, los bloques de planificación y guiado revisten una mayor importancia y están fuertemente relacionados (simbolizado por la línea que los une). Dado que las tareas son de manipulación en el mundo real, se puede asociar un sistema de referencia al objeto manipulado y a través de él controlar la ejecución de toda la tarea, este es el sistema que se teleopera desde el nodo–mando a través del control o dispositivo háptico. De esta forma, las tareas se ejecutan siempre en el espacio *SE(3)* (*Special Euclidean Group*), pudiendo existir también obstáculos y otros tipos de restricciones reales o virtuales que limitarán los movimientos del robot que la ejecute, llegando a definir subespacios o subconjuntos en *SE(3)* que consti-

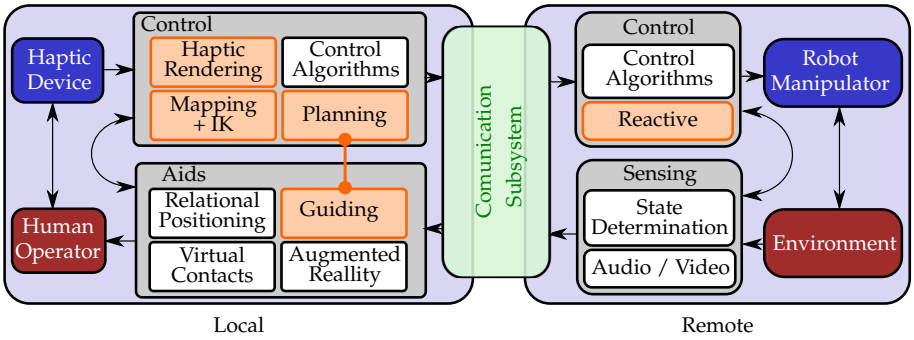


Figura 1.1: Esquema general del marco de teleoperación desarrollado en el Instituto de Organización y Control de Sistemas Industriales (IOC) donde se resaltan, en naranja, las partes involucradas en este trabajo.

tuyan la solución de la operación.

Una vez es definida la tarea y el robot que la llevará a cabo, se planifica la solución teniendo en cuenta las características propias del robot y su entorno. Esta forma de teleoperación asistida mediante el guiado háptico se puede dividir en dos momentos de tiempo diferentes, el primero en el que se define y planifica la solución de la tarea o su conjunto posible de soluciones y el segundo en el cual se ejecutan los movimientos, contando con la pericia del operador y la adecuada interacción con el sistema de guiado. De esta forma, durante el proceso de ejecución, el mismo operador forma parte del lazo de control de la tarea y de acuerdo a su interés particular, el sistema de guiado actualiza la solución tratando de incorporar los lineamientos que el operador va impartiendo con sus movimientos. La presencia de obstáculos dentro del volumen de trabajo del robot que pueden limitar los movimientos durante la ejecución de la tarea y la obtención de planes que sean factibles de ser realizados por el robot remoto, obligan a tratar de forma conjunta la resolución de la tarea y la planificación de los movimientos del robot.

Un escenario en el cuál existe particular interés es aquel conformado por dos o más teleoperadores interactuando al mismo tiempo sobre una celda de manufactura con igual número de robots, en el cual cada uno de ellos está trabajando en tareas independientes que pueden ser diferentes. Aquí, el nodo–operación realimenta a todos los nodos–mando involucrados en el desarrollo de las tareas, las configuraciones de cada uno de los robots que lo conforman y las posiciones y orientaciones de cada uno de

los obstáculos identificados dentro del espacio de trabajo, si existen. Esta actualización permite a los nodos–mando adaptar su planificación a las cambiantes condiciones de la ejecución de las tareas.

La planificación obtiene como resultado el camino dentro del espacio de configuraciones del robot remoto (*SE(3)* $\times \mathbb{R}^n$), que una vez trasladado al espacio de movimientos del efector final o pieza manipulada, da cumplimiento con los objetivos de la tarea y se convierte en la base de la generación de las fuerzas de guiado. Al mismo tiempo, en el nodo–mando se puede hacer uso de la guía visual del camino dentro del espacio $\mathbb{R}^3$ que ha de seguir el efector final del robot o el sistema coordenado asociado al objeto de la planificación.

Si durante la ejecución el operador no desea seguir el camino propuesto, lo que indica un deseo voluntario de incluir su "experticia" en el desarrollo de la tarea, se puede replanificar incluyendo esta información, de igual forma que se hace cuando un obstáculo móvil intercepta la ruta activa utilizada como guía para la ejecución. Esta característica está definida por la periódica y continua realimentación de las configuraciones de los robots del nodo–operación y las demás variables del entorno del robot que son adquiridas y que pueden ser utilizados para mejorar la telepresencia del operador.

El nodo–mando refleja en todo momento y para todo el espacio disponible, un campo de fuerzas que guían al operador hacia el camino encontrado y al cumplimiento satisfactorio de la tarea, aunque la magnitud de estas fuerzas puede ser fácilmente vencida por el operador en cualquier momento.

1.2.1. Planificación de movimientos

A lo largo de las dos últimas décadas, la planificación de movimientos se ha venido decantando por el uso de métodos de muestreo para el modelado del espacio de configuraciones por encima de los que ofrecen una representación completa de él. El concepto de espacio de configuraciones introducido por Lozano-Pérez y Wesley (1979) reduce el robot a un punto y por lo tanto el problema de planificación de movimientos se define como la búsqueda de un camino entre una configuración inicial y otra final dentro de este espacio.

Los métodos basados en muestreo más comúnmente utilizados son los mapas de caminos probabilísticos (*"Probabilistic Roadmaps – PRM"*) y los árboles aleatorios de crecimiento rápido (*"Rapidly Random Tree – RRT"*) con un sin número de variaciones y modificaciones que aumentan su efectividad en situaciones particulares. El sistema propuesto es del tipo de múltiple pregunta (*multiquery*) como vía de implementación de la capacidad de replanificación que se ofrece frente a las pautas de evolución del entorno y al deseo expreso del operador. Es de recalcar que en los entornos multirobot, los "otros" robots se tratan como obstáculos en movimiento.

Paralelamente se desarrolla un modelo virtual del laboratorio, que al igual que en (Safaric et al., 2001) es usado para validar el planificador y la interacción entre el (los) nodo(s)–mando y el nodo–operación. Se ofrece entonces, una herramienta de planificación de movimientos escalable y desarrollada teniendo en cuenta los lineamientos de la Programación Orientada a Objetos (Pérez y Rosell, 2010).

1.2.2. Realimentación al operador

La ejecución de la tarea consiste en comandar los movimientos del efector final del robot a través del dispositivo háptico basándose en el seguimiento del camino planificado, de forma que se asegura en todo momento la culminación satisfactoria de la tarea. Con base en este camino se genera el campo de fuerzas que guía al teleoperador en la ejecución de dicha tarea.

La realimentación de fuerza que ofrece el dispositivo háptico utilizado, refuerza el conocimiento que el operador puede tener sobre el sistema manipulado y aumenta la transparencia del sistema. Por otra parte, esta fuerza de realimentación puede ser adaptada con el fin de ofrecer al operador información proveniente de diferentes fuentes, sin embargo cuando existe una fuerza compuesta originada por más de una fuente a la vez, esta asociación puede diluirse enmascarando la relevancia que en un principio quería ofrecerse. Es por esta razón que se utilizan diferentes matices y momentos de aplicación para brindar al usuario una experiencia de "telepresencia" coherente.

1.3. Objetivos

1. Desarrollar un sistema de soporte a la teleoperación (definida de forma conjunta para un nodo–mando y un nodo–operación asociado), que brinde al operador una fuerza de guiado que le permita seguir el camino que da solución a la tarea, evitando las colisiones del robot con su entorno, el cual puede incluir tanto obstáculos estáticos como dinámicos.

   a) Definir un esquema de transformación entre los espacios de trabajo del dispositivo háptico y del robot que permitan una teleoperación de la tarea intuitiva y transparente.

   b) Desarrollar algoritmos de planificación de caminos basados en *PRM*, adaptados a la problemática de brindar soporte al usuario durante la ejecución de las tareas teleoperadas, adaptando los métodos de muestreo a las particularidades que tienen los problemas de teleoperación en entornos dinámicos.

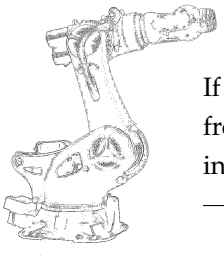
   c) Plantear ayudas hápticas simples e intuitivas basadas en la existencia de un camino solución que permitan la evitación de colisiones y los cambios de configuración, y que sean flexibles y reconfigurables.

   d) Desarrollar estrategias de interacción en entornos con múltiples robots y múltiples operadores, de forma que el sistema de planificación de la tarea de un robot y el sistema de teleoperación en general, consideren como obstáculos móviles los otros robots del nodo–operación.

   e) Desarrollar un *software* que simule el comportamiento del nodo–operación remoto donde sea posible validar los desarrollos aquí contenidos y el funcionamiento del sistema de guiado háptico sobre una plataforma virtual.

2

Marco de refencia



If you wish to make an apple pie from scratch, you must first invent the universe.

Carl Sagan, Cosmos

Con este capítulo se pretende contextualizar el presente trabajo dentro del extenso mundo de la robótica y en particular dentro de las áreas de conocimiento de la planificación de movimientos y la teleoperación.

2.1. Planificación de movimientos

Desde los orígenes de la planificación de movimientos y el muy recordado problema del *piano mover* se han definido un sin número de estrategias para generar automáticamente la secuencia de movimientos que deberán aplicarse a un objeto y conseguir con ello desplazarlo de un lugar a otro evadiendo posibles colisiones. Entre ellas, la más citada y recordada es la definición del espacio de configuraciones ($\mathcal{C}$) por parte de Lozano-Pérez (1983) que surgieron de sus trabajos previos con obstáculos poliédricos (Lozano-Pérez y Wesley, 1979) y que terminó de definir en un algoritmo sencillo y extensible (Lozano-Pérez, 1987) en el que reduce el robot a un punto en este espacio.

La problemática más grande que ofrece esta técnica del espacio de configuraciones consiste en determinar su forma y lograr dividir el espacio entre el que corresponde a los obstáculos ($\mathcal{C}_{obs}$) y el que se encuentra libre ($\mathcal{C}_{free}$).

Los métodos clásicos optan por el conocimiento exacto del espacio a través de formulaciones matemáticas que pueden llegar a ser computacionalmente muy costosas. Por otro lado surgieron enfoques en los que se pretende tener un conocimiento suficiente de este espacio a través de un conjunto de muestras obtenidas de él.

2.1.1. Métodos basados en muestreo

En contraposición al gran costo computacional que representa el conocimiento exacto de $\mathcal{C}$, aparecen y pronto toman fuerza, los métodos basados en muestreo como los mapas de carretera probabilísticos o *PRM* (*Probabilistic RoadMap*) planteados por Kavraki et al. (1996) o los árboles de exploración rápida o *RRT* (*Rapidly-exploring Random Tree*) planteados por LaValle (2006) y todas aquellas variantes y/o particularizaciones que han surgido a partir de ellas. Los fundamentos de la exploración probabilística de $\mathcal{C}$ se encuentran bien resumidos en (Svestka y Overmars, 1998).

Para obtener las muestras del espacio de configuraciones existen dos tendencias claramente diferenciadas entre aquellos métodos que las obtienen de forma aleatoria y otros a través de métodos determinísticos. Son conocidos los diferentes métodos de obtención de números aleatorios que son la base para conseguir las muestras aleatorias, y dentro de los determinísticos se pueden nombrar por un lado las secuencias como la propuesta por Halton (1960a) que lleva su mismo nombre, y las basadas en métodos de construcción digital de Lindemann et al. (2004) y Rosell et al. (2007a). Existe una basta teoría sobre las características que deben tener las muestras con el fin de obtener la mayor cantidad de información del espacio al que pertenecen, con el mínimo número de ellas. La dispersión y la discrepancia son las medidas más importantes sobre el cubrimiento y distribución que adoptan las muestras sobre el espacio explorado (LaValle, 2006, Cap. 5).

La parte más importante y también la más compleja es adquirir el conocimiento del espacio de configuraciones o, lo que es igual, poder determinar zonas libres conexas sobre las cuales trazar la solución a las preguntas de planificación. Para ello se hace uso de los algoritmos de detección de colisiones que son métodos de geometría computacional capaces de determinar si dos o más sólidos se encuentran o no en colisión (Jiménez et al., 1998). Estos algoritmos presentan normalmente dos opciones de prueba, una más rápida en la que utilizan las cajas englobantes (*Bounding Box*) que

encierran los sólidos y otro más preciso en el que usan propiamente los sólidos. Estas cajas englobantes de los sólidos pueden ser de dos tipos: alineadas a los ejes del sistema coordenado global (*Axis Aligned Bounding Box*) por van den Bergen (1997)) o alineadas a los ejes principales de los propios sólidos (*Oriented Bounding Box*) por Gottschalk et al. (1996)).

Los sólidos que representan tanto a los robots como a los elementos de su entorno, pueden ser cualquier geometría básica como: cilindros, esferas, conos o paralelepípedos que tienen representaciones simples; o también pueden ser geometrías más complejas que necesiten métodos más elaborados para representarlos. Uno de estos métodos de representación más conocido y ampliamente utilizado, es el de las mallas triangulares ya que ofrece ventajas frente a los algoritmos de detección de colisiones y para las cuales existen varios métodos descritos en la literatura como los utilizados en la implementación de la librería PQP de Larsen et al. (1999).

Este proceso de validación de colisiones es el más costosos computacionalmente, y es por esta razón que han aparecido enfoques donde esta validación se posterga lo más tarde posible, con la finalidad de minimizar el número de validaciones de muestras que no hacen parte del camino solución (Bohlin y Kavraki, 2000).

Una vez se determina si las muestras son o no libres, es necesario realizar la conexión entre ellas. Es aquí donde aparecen los algoritmos de búsqueda de vecinos como los planteados por Arya et al. (1998) o el basado en descomposiciones jerárquicas en árboles como el desarrollado por Yershova y LaValle (2007), que permiten determinar la cercanía de las muestras anteriormente obtenidas y establecer los criterios de conexión entre ellas. Aquí entra en funcionamiento el planificador local que determina el camino para unir la nueva muestra a la vecina elegida. En este proceso de búsqueda de vecinos es muy importante la elección del criterio que determina la vecindad y que en términos generales está dominado por la métrica (LaValle, 2006, Cap. 5).

Se puede ver cómo los *PRMs* y los *RRTs* transforman el problema de planificación en uno análogo de búsqueda del camino más corto en un grafo, donde usando el algoritmo de Dijkstra (1959) o el A* (Hart et al., 1968) se puede encontrar la ruta óptima. Otros métodos de búsqueda heredados de la inteligencia artificial, como los algoritmos genéticos o las redes neuronales (Rabin, 2003), también han sido utilizados con este fin.

El conocimiento del espacio de configuraciones también está determinado por la cantidad de preguntas de planificación que se ejecutarán en él y la estrategia propia de solución de las consultas. De esta forma los métodos se dividen en los de única pregunta (*Single–query*) como los *RRT* o los de multiple pregunta (*Multiple-query*) como los *PRM*. Para estos métodos se han propuesto diversas modificaciones que permitan agilizar este proceso de conocimiento de $\mathcal{C}$ como los descritos por Morales et al. (2007), donde los autores proponen el uso de métricas locales para analizar la complejidad de $\mathcal{C}$, o en Rosell et al. (2007b) donde se hace una descomposición jerárquica basada en la secuencia deterministica $S_d(k)$ y la métrica de $\mathcal{C}$, o como en Zhang et al. (2007) donde se combina una descomposición jerárquica con un *PRM*.

De los *PRM* se han elaborado estudios sobre la completez (Kavraki et al., 1998), sobre las fuentes de muestreo y los sesgos (Hsu et al., 2005), se han realizado comparaciones importantes entre algunas de las diferenes variantes que existen (Geraerts y Overmars, 2002) y se han comparado con los métodos de búsqueda clásicos sobre gradillas (LaValle et al., 2004). Un interesante informe sobre métodos de planificación basados en muestreo se puede encontrar en (Lindemann y LaValle, 2005).

Aunque la planificación de movimientos es madura dentro de la robótica, sigue siendo un área muy activa de investigación en temas tales como la optimización de los caminos (Akgun y Stilman, 2011; Karaman y Frazzoli, 2011), el manejo de la incertidumbre (Agha-mohammadi et al., 2011; Jaillet et al., 2011), aspectos de su implementación (Şucan y Kavraki, 2010; Svenstrup et al., 2011) y en temas más básicos como la conectividad (combinación entre *PRM* y *RRT*, Alterovitz et al. (2011); Kuffner y LaValle (2011)) y el sesgo en regiones de especial interés del espacio de configuraciones (Denny y Amato, 2011; Rantanen, 2011).

2.1.2. Entornos multirobot

Todos estos métodos han tenido su inicio en el ámbito monorobot pero en los últimos años se ha ido incrementando el número de trabajos relacionados con la planificación en entornos multirobot y que al utilizarlos en tareas teleoperadas pueden caer también en entornos multioperario como los tratados en (Li y Gupta, 2007) y (Sirouspour y Setoodeh, 2005).

En los entornos mutirobot se presentan dos situaciones que restringen

la libre planificación de los robots de forma independiente: compartir el espacio de trabajo y la manipulación conjunta de objetos lo que genera una cadena cinemática cerrada (Cortés et al., 2002; Khatib et al., 1996). En el primer caso, cuando dos o más robots comparten el espacio de trabajo, es necesario tener en cuenta que para cada uno de ellos, los otros robots se convertirán en objetos en movimiento.

Los métodos de planificación desarrollados para los entornos monorobot, pueden ser extendidos hacia los entornos multirobot usándolos bajo un esquema de planificación distribuida definiendo prioridades o jerarquías al momento de compatibilizar las soluciones particulares (Todt et al., 2000). Otra forma de extenderlos, es usar un esquema de programación (*scheduling*) con el cual se asignan tiempos de ejecución y se van desarrollando la planificación conjunta. Los métodos que son relativamente más fáciles de extender son los basados en muestreo dada la complejidad inherente que tienen los métodos exactos de descomposición del espacio de configuraciones.

También existen métodos específicamente desarrollados para estos entornos, en donde se involucran todos los robots presentes. La aplicación de estos métodos para el desarrollo de tareas conjuntas o cooperativas se pueden observar en (Li y Gupta, 2006). Estos métodos como los *super-graph* (Svestka y Overmars, 1998), pueden acarrear costos computacionales muy grandes que los pueden llegar a ser inviables en la práctica.

2.1.3. Restricción de la tarea

Dadas las características cinemáticas o dinámicas de los robots implicados, o de las características propias de la definición de las tareas, se pueden presentar restricciones a la solución que pueda ofrecer el planificador de movimientos. Estas restricciones pueden ser del tipo holonómicas (Svestka y Overmars, 1998) o geométricas como las presentes por ejemplo al transportar objetos que contienen líquidos o al tratar de abrir la gaveta de un mueble. En (Stilman, 2007) y en (Yao y Gupta, 2007) se pueden ver soluciones interesantes al problema general de restricciones de los posibles movimientos del efector final, como el traslado de estas restricciones al espacio de configuraciones del robot.

Otro tipo de restricciones que pueden existir, se basan en condiciones de temporalidad como el cambio de ubicación y/u orientación de los obs-

táculos presentes como sucede en los ambientes dinámicos. Frente a este tipo de restricciones se han plantado soluciones con algoritmos genéticos como en (Wang et al., 2007), la selección del camino más apropiado dentro del conjuntos que ofrecen la misma solución (Bernabeu, 2007) o en el caso de los *RRT* que pueden replanificar la solución como en (Zucker et al., 2007).

2.1.4. Entornos dinámicos

A lo largo del último lustro, la actividad investigadora relacionada con la solución del problema de planificación de trayectorias en entornos dinámicos ha crecido junto con la capacidad para adquirir el estado del entorno del robot. La cantidad de sensores con los que están equipados los robots ha aumentado, con el consecuente aumento del número de aplicaciones en entornos no estructurados, como aquellas relacionadas con los robots humanoides y los manipuladores móviles.

La característica importante que distingue los diferentes enfoques de la planificación de trayectorias en entornos dinámicos, es el tipo de movimiento de los obstáculos, particularmente si es predecible o no. El primer caso generalmente está centrado en encontrar el camino en el espacio de configuración ampliado con el tiempo, es decir, en el *CTspace*. Cuando el movimiento no es predecible, los planificadores incorporan un comportamiento reactivo con el medio ambiente circundante. Esta capacidad de reacción puede ser local, como en los problemas de exploración, o global, como en los problemas de navegación. Este comportamiento reactivo es también necesario en los sistemas de teleoperación en tiempo real, como las que se detallan por Lumelsky y Cheung (1991, 1993), y que forman parte de la motivación del presente trabajo.

La característica reactiva de la planificación, se puede lograr desde el punto de vista del procedimiento de planificación en sí, o desde el punto de vista del grafo asociado (estas dos formas no tienen por qué ser incompatibles). Los planificadores básicos pueden ser adaptados para ser incluidos, por ejemplo, en un bucle de tiempo real para replanificar la ruta deseada como se hace por Kunz et al. (2010) y por Geeks (2007), para tener en cuenta los estados de collisión inevitable (Bekris, 2010) o también abordar el problema desde un punto de vista más teórico como en el caso del estudio de completez (Hauser y Latombe, 2010; van den Berg et al., 2009). Desde el

contexto de los algoritmos de búsqueda en grafos, se han propuesto modificaciones del algoritmo de búsqueda A* como los algoritmos: *Differencial A** (Trovato, 1989) y (Trovato y Dorst, 2002), las variantes D* y D*-*Lite* (Stentz, 1994) y (Koenig y Likhachev, 2002), el *Anytime dynamic A** (Likhachev et al., 2005) y (Ferguson y Stentz, 2007), el *Anytime dynamic path-planning* (Belghith et al., 2006) o en la aplicación de estos algoritmos en nuevos espacios topológicos (Yershov y LaValle, 2011).

2.2. Teleoperación

2.2.1. Esquemas de teleoperación

Dentro de los trabajos desarrollados en el ámbito de la robótica teleoperada se ha hecho énfasis en los diversos aspectos que marcan la interacción y el comportamiento de los componentes del sistema y entre ellos se pueden destacar: los esquemas de control, la autonomía del robot remoto y los retardos. Estos sistemas teleoperados están compuestos principalmente por tres bloques funcionales: la estación local o nodo–mando donde estará situado el operador, la estación remota o nodo–operación donde esta ubicado el robot y el canal de comunicación que permite establecer el intercambio de información entre ellos (Basañez y Suárez, 2009). Estos tres bloques se pueden distinguir claramente en la Figura 1.1.

La estación local dispone de una interfaz que permite al operador enviar y recibir información del sistema. A través de esta interfaz se puede mostrar información de tipo gráfica, sonora, de fuerzas o cualquier otra que permita incrementar la sensación de telepresencia del operador (Nuño y Basañez, 2006).

Dentro de los esquemas de control que se han propuesto se encuentran aquellos clásicos basados en el modelo maestro–esclavo utilizados desde 1947 por Raymond Goertz, siendo este el pionero en la manipulación de sustancias químicas peligrosas a través de estos mecanismos. Este esquema consiste en que el esclavo se limita a seguir la evolución de los movimientos realizados por el maestro, lo que conlleva una dependencia desde el punto de vista cinemático. Es de resaltar que el control es de lazo cerrado e involucra a las dos estaciones en él. Aunque su planteamiento es el más antiguo, aún es vigente y puede ser utilizado en diversas aplicaciones co-

mo se puede ver en (Carignan y Olsson, 2004) donde se realiza un montaje destinado a técnicas de rehabilitación.

Existen otros esquemas un poco más elaborados de realizar, entre los cuales se pueden destacar dos grandes grupos de acuerdo a la autonomía que le brindan al robot remoto, el control coordinado y el control supervisado. En el control coordinado, el operador controla los actuadores pero además existe un lazo adicional de control en el robot remoto que le permite hacer un trabajo de más alto nivel (olvidándose de las particularidades mecánicas por ejemplo), y en el control supervisado se le permite al robot remoto realizar parte de las tareas de forma más o menos autónoma.

Estos esquemas inicialmente desarrollados para entornos de un único operador y un único robot teleoperados se han venido extendiendo a lo que se denomina sistemas MOMR (multioperador–multirobot). Los escenarios multirobot presentan diversas ventajas, entre las que se encuentran el aumento de la flexibilidad, la productividad, la capacidad de carga y la posibilidad de realizar tareas complejas entre otras, aunque también conlleva algunas desventajas entre las que se destacan el aumento en la complejidad del control, de la planificación de las trayectorias y de su consiguiente programación. La mayor cantidad de trabajos desarrollados en el área de la manipulación multirobot están enfocados al desarrollo de tareas colaborativas en donde pueden aplicarse diversas estrategias entre ellas las competitivas de Liu et al. (2005) o las que se basan en la interacción de fuerzas tanto externas como internas (Costa y Basañez, 2004).

Una de las premisas perseguidas con este trabajo es liberar al operador de la necesidad de conocer el robot y su entorno, y es por esta razón que el esquema de control implantado es del tipo de impedancia (Figura 2.1) en el que se impone al robot el comportamiento de un sistema físico compuesto por una masa, un muelle y un amortiguador (Caccavale et al., 2007). Este sistema se controla desde un dispositivo háptico que posee los seis grados de libertad (*gdl*), que podría tener cualquier objeto libre en el espacio y que se utiliza para realimentar fuerzas del entorno remoto al operador (Lo et al., 2004). Otros esquemas de teleoperación de robot así como trabajos relevantes desarrollados sobre teleoperación en entornos dinámicos y en ambientes multirobot pueden encontrarse ampliamente discutidos por Jihong et al. (2006), Kikuchi et al. (1998) y Nuño et al. (2010) donde se hace referencia más profunda a la capa de control.

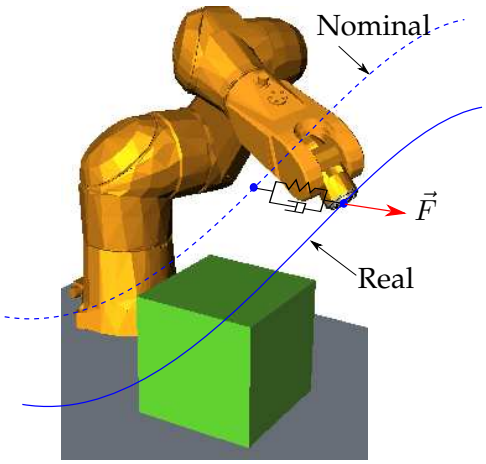


Figura 2.1: Trayectorias en comportamiento por impedancia.

El otro tópico importante dentro de la teleoperación es el manejo de los retardos causados por la trasmisión de la información entre los nodos o estaciones y las consecuencias que ésta tiene en el seguimiento de las consignas y las estabilidad de los controladores, efecto que se ve acentuado en los sistemas MOMR (Elhajj et al., 2001; Lee y Spong, 2005; Lee et al., 2005, 2006; Martinez-Palafox et al., 2006). Existen diferentes propuestas y estudios sobre este fenómeno inherente a las comunicaciones sobre redes de datos y especialmente a las públicas como Internet o Internet2 y una posible forma de abarcarlos, consiste en modelarlos como constantes de la forma en la que se tratan en (Lee y Spong, 2006).

2.2.2. Infraestructura de *software*

Con las líneas de investigación que cubre la división de robótica del IOC, se creó la necesidad de formar una infraestructura de *software* capaz de interconectar lógicamente todos los dispositivos de los que se dispone en sus laboratorios además de una herramienta de planificación de movimientos con capacidad para servir como ayuda para la teleoperación. Este es el origen del desarrollo de *The Kautham Project* (sir.upc.es/kautham).

El desarrollo de software en el campo de la robótica se ha incrementado durante los últimos años debido al enorme esfuerzo realizado para mejorar el rendimiento y las capacidades de interacción de los sistemas robóticos. En la actualidad, el objetivo de estas aplicaciones se centran en los conceptos más abstractos y valiosos de la ingeniería de software, co-

mo la reutilización, genericidad y la interoperabilidad. Algunos sistemas que siguen esta línea de desarrollo ya están disponibles en Internet como ROS (Quigley et al., 2009), OROCOS (Bruyninckx et al., 2003), OpenRave (Diankov, 2010), OOPS (Plaku et al., 2007) y OMPL (ompl.kavrakilab.org). Mientras OpenRave, OOPS y OMPL están enfocados en el área de la planificación de movimientos y de la manipulación, ROS y OROCOS lo están en proporcionar elementos de control a bajo nivel de dispositivos, de abstracción de hardware, implementación de funcionalidades genéricas, paso de mensajes entre procesos y en ofrecer herramientas y librerías de código abierto para el desarrollo de aplicaciones que puedan ser ejecutadas en múltiples equipos.

El diseño e implementación de *The Kautham Project* se ha llevado a cabo de forma paralela al desarrollo de los marcos antes citados, y comparten muchas de sus ideas y utiliza también muchas de las bibliotecas y herramientas en común como PQP para la detección de colisiones, Coin3D para la visualización 3D, Qt para el interfaz gráfica de usuario, el uso de XML para la descripción de los robots y la tarea, CMake como sistema de construcción del proyecto de *software* y Doxygen para la documentación del código.

El resto de infraestructura que permite conectar lógicamente los diferentes dispositivos dentro del laboratorio de robótica está utilizando ROS con el fin de incorporar toda la sinergia que está impulsando este sistema y la gran cantidad de personas e instituciones comprometidas con su desarrollo y mantenimiento.

Dada la dependencia a las instalaciones físicas del robot, se han desarrollado sistemas de manipulación de robots virtuales (Mellado et al., 2003) e implementado laboratorios virtuales de robótica como los propuestos en (Safaric et al., 2001, 2003). Se ha adoptado esta estrategia como mecanismo de validación de los desarrollos propios de este proyecto y es por esta razón que *The Kautham Project* funciona como un entorno virtual que permite teleoperar los robots que aparecen allí.

2.3. Ayudas a la teleoperación

La teleoperación ha evolucionado en muchos aspectos desde sus orígenes al final de los años 40 (Kumar, 2010), por ejemplo, mediante el desarrollo de algoritmos de control robusto que manejan adecuadamente los canales de comunicación con retrasos de tiempo variables (como Internet) que ha permitido ampliar la cobertura de teleoperación, o mediante el uso de dispositivos de realimentación de fuerza en sistemas de teleoperación bilateral, lo que permite ampliar el tipo de tareas que pueden ser teleoperadas.

La ejecución de las tareas de teleoperación se puede mejorar significativamente si se proporciona un adecuado soporte al operador. En esta línea ya han sido publicadas varias propuestas desde principios de los años noventa. Brooks y Ince (1992) presentan, por ejemplo, un análisis global de las ayudas visuales y los elementos claves que pueden mejorar la ejecución de las tareas teleoperadas, como:

- El seguimiento automático del efector final.
- Colocación de la cámara/iluminación asistida por computador
- El uso de las coordenadas de pantalla para resolver el problema de desajuste entre estímulo-respuesta.

También presenta mejoras en la visualización (más tarde conocida como realidad aumentada) para:

- Resaltar las zonas no accesibles o peligrosas.
- Resaltar los obstáculos que pueden producir una colisión inminente.
- Dar indicaciones visuales que puedan mejorar la orientación y la posición del efector final.
- Dar indicaciones de la profundidad con vistas virtuales o una rejillas con perspectiva.

Recientemente se han propuesto sistemas de realimentacion visual estereoscópica, que combinan imágenes provenientes de dos cámaras de vídeo accionadas de forma remota, y que mejoran la percepción de profundidad (Portilla y Basañez, 2007).

Con el fin de mejorar el rendimiento en escenarios complejos con obstáculos (aunque limitado a robots de tres *gdl*), algunos métodos también proponen utilizar el punto de vista del espacio de configuraciones, donde el usuario manipula el robot como un único punto, adicionalmente al del espacio de trabajo con el objetivo de superar algunas de las habituales deficiencias humanas de razonamiento espacial (Ivanisevic y Lumelsky, 2000; Morishige y Noborio, 2000).

2.3.1. Unión virtual y espacios de trabajo

En términos generales, la teleoperación debe ser transparente e intuitiva, lo que requiere que el robot, el dispositivo háptico y el brazo humano tengan estructuras cinemáticas similares. En este sentido, algunos enfoques han trabajado en el diseño de dispositivos hápticos que no restringen la movilidad del brazo humano conectándolo de una manera firme y natural (Folgheraiter et al., 2009), mientras que otros trabajan en el diseño de telemanipuladores capaces de replicar la habilidad de los movimientos humanos (Dominjon et al., 2004). Sin embargo, en muchos escenarios de teleoperación la cinemática del dispositivo maestro y la del robot son muy diferentes, como cuando un robot industrial es teleoperado con un dispositivo háptico de escritorio. Este hecho requiere que la teleoperación se realice en el espacio cartesiano, utilizando la cinemática inversa del robot.

Uno de los problemas asociados con este enfoque es la correspondencia entre el espacio de trabajo del dispositivo háptico y el del robot, porque por lo general el espacio de trabajo del dispositivo háptico es mucho menor que el del robot. Hay varios métodos para llevar a cabo esta correspondencia (Conti y Khatib, 2005):

Resincronización: Este método, también llamado indexación o embrague, es equivalente al clásico salto de un ratón (*mouse jump*) en 2D. Se basa en un modo de control de posición en la que los desplazamientos (posición y orientación) del efector final del dispositivo háptico se asignan directamente a los desplazamientos del efector final del robot, con un factor de escala que puede ser ajustado por el usuario. Al llegar a los límites del espacio de trabajo del dispositivo háptico, el usuario termina la sincronización haciendo uso de una entrada adicional, y reposiciona el dispositivo háptico sin mover el robot, donde

se sincroniza de nuevo el robot con el dispositivo háptico (Azorin et al., 2004).

Seguimiento balístico: En este método se ajusta la escala entre las áreas de trabajo como una función de la velocidad del dispositivo háptico, es decir, que cuando el dispositivo háptico se mueve rápidamente el factor de escala es grande y se hace un control de posición basto, mientras que cuando se mueve lentamente el factor de escala es pequeña lo que resulta en un control fino de la posición.

Control de velocidad: Este método no proporciona una correspondencia física directa entre los espacios de trabajo, sino que la velocidad del robot es proporcional a la posición del dispositivo háptico, como lo hacen las palancas de mando de los videojuegos o *joysticks* (Horan y Nahavandi, 2008).

Control de deriva: Este método consiste en un reposicionamiento continuo e imperceptible del espacio de trabajo, mientras que el usuario mueve el dispositivo háptico. Esta deriva es proporcional a la velocidad de la mano del usuario y a la distancia del efector final del dispositivo háptico al centro de su espacio de trabajo, lo que hace al usuario mantenerse lejos de los límites del área de trabajo (Conti y Khatib, 2005).

Cada uno de estos métodos tiene sus ventajas y desventajas. La resincronización hace necesario el uso de una entrada adicional (un interruptor, botón o pedal) para desacoplar el dispositivo háptico, y esta acción puede interferir en la tarea, que puede convertirse un problema si alguna parte de ella se debe hacer de manera continua. Por otro lado, el control se realiza a nivel de posición, lo que resulta útil cuando la tarea necesita de precisión. Se supone que la ventaja del seguimiento balístico es la escala automática, pero esto puede producir un corrimiento indeseado cuando el usuario se mueve rápidamente en una dirección y lentamente en la dirección opuesta, perdiendo así el control sobre la zona de exploración del área de trabajo. El control de velocidad es intuitivo y útil para los movimientos secundarios, pero no es adecuada para trabajos de precisión, ni para movimientos rápidos (acelerados). El control de deriva tiene las ventajas del control de posición, sin las desventajas del método de resincronización, pero el usua-

rio puede percibir algo de distorsión en función de la ganancia de la deriva, es decir, las fuerzas que produce la deriva pueden interferir con las otras fuerzas producidas por los contactos virtuales o las provenientes del lazo de control bilateral. Para beneficiarse de las ventajas y minimizar dichos inconvenientes, se han propuesto otros métodos como el *"buble"* que combina el control de posición y control de la frecuencia (Dominjon et al., 2005), o el método propuesto por Chotiprayanakul y Liu (2009), que combina el control de la deriva con el seguimiento balístico. Una característica que debe tenerse en cuenta es que la teleoperación de tareas de precisión puede ser difícil y agotadora para el usuario.

2.3.2. Asistencia en la ejecución

El esquema de teleoperacion que sobre el que se desarrolla este trabajo puede utilizar dispositivos hápticos como el Phantom (www.sensable.com), el Falcon (www.novint.com) o el Omega (www.forcedimension.com), que son dispositivos de realimentación de fuerza ampliamente utilizados como dispositivos maestro en los sistemas de teleoperación. El uso de este tipo de dispositivos, permite desarrollar la tarea de teleoperación a nivel del espacio Cartesiano, evitando las incompatibilidades de tipo cinemático con el robot esclavo (Conti y Khatib, 2005).

Cuando en el nodo–mando o local se utiliza un dispositivo háptico, se pueden ofrecer otros tipos de soportes, adicionales a las de tipo visual comentadas anteriormente, que exploten la capacidad de realimentación de fuerzas lo que tienden a mejorar en gran medida, el rendimiento de la teleoperación. Por ejemplo, las fijaciones virtuales o limitaciones de los grados de libertad se usan para restringir los movimientos de los usuarios dentro de una región determinada o dentro de un subespacio de menor dimensión, lo que le permiten concentrarse en el mando de los movimientos pertinentes a la tarea, dando como resultado ejecuciones más rápidas y precisas y reduciendo la carga de trabajo del operador. Se pueden predefinir algunas limitaciones virtuales para determinadas acciones como los movimientos de aproximación o para las operaciones de insertar/extraer presentadas por Payandeh y Stanisic (2002). Además estas limitaciones virtuales pueden ser especificadas por solucionadores de restricciones geométricas, que encuentran un subespacio parametrizado de menor dimensión que satisface un conjunto de restricciones de fácil definición por el usuario, a partir

del conocimiento de la tarea a ser ejecutada (Rodríguez et al., 2010).

La capacidad que tienen los dispotivos hápticos de este tipo, de realimentar fuerza al operador, también se puede utilizar para:

- Evitar los obstáculos, es decir, se puede genera un campo de fuerza proporcional a la proximidad entre los obstáculos y el efector final, que repela los movimientos de aproximación a los obstáculos, por ejemplo, mediante el uso de un mapa del entorno (Diolaiti y Melchiorri, 2003) o con el reconocimiento de los obstáculos por medio de marcadores predefinidos (Park et al., 2007).

- Evitar las singularidades, es decir, se puede generar un campo de fuerza proporcional a la proximidad a una singularidad y a la magnitud y la dirección de la consigna, para conducir al usuario a una dirección factible definida por el jacobiano del manipulador (Maneewarn y Hannaford, 1999);

- Evitar que en las tareas donde exista interacción física, al igual que las tareas de montaje, se generen fuerzas de contacto grandes, es decir, las fuerzas de contacto calculadas a partir de un modelo de la tarea se puede realimentar al usuario de forma que reaccione a tiempo y evite las enormes fuerzas reales de la colisión en el sitio remoto (Hirai et al., 2000; Rosell y Vázquez, 2005).

Las fuerzas de guiado también se han sugerido con el fin de conducir al usuario a lo largo de un camino libre de colisiones hacia la configuración objetivo, dando como resultado un comandamiento más fácil y rápido de la tarea. Teniendo en cuenta que la planificación de movimientos ya es una disciplina madura dentro del área de la robótica, y que las técnicas basadas en muestreo son las más eficientes en desempeño, recursos y tiempo para programar los movimientos de un robot, se pueden emplear también para generar las fuerzas de guiado en la ejecución de tareas teleoperadas como se puede ver en (Rosell et al., 2008a). En particular, se han adelantado trabajos en planificación de movimientos y generación de fuerzas para el guiado a través de funciones armónicas durante la realización de las tareas como el realizado por Rosell et al. (2007b) y Vázquez y Rosell (2007).

Estos enfoques, sin embargo, están limitados a los robots móviles con dos *gdl* (Jarvis, 2010), o los movimientos de traslación en dos o tres *gdl*

del objeto que está siendo manipulado (Rosell et al., 2008b; Vazquez et al., 2010), es decir, sin tener en cuenta la cinemática y la geometría del robot que manipula el objeto. Hasta donde el autor conoce, dentro de la literatura no hay trabajos sobre sistemas de asistencia con ayudas de guiado para teleoperar robots manipuladores.

Sin embargo, el uso de los dispositivos hápticos como dispositivo maestro en los sistemas de teleoperación conlleva algunos problemas que necesitan ser manejados correctamente, y que a menudo se han pasado por alto. El primer problema está relacionado con el hecho de que los espacios de trabajo del robot y del dispositivo háptico tienen tamaños muy diferentes, siendo el espacio de trabajo del dispositivo háptico mucho menor que el del robot (generalmente, exceptuando los casos de nanorobótica). Esto requiere una adecuada correspondencia entre las áreas de trabajo, que ha sido tratado desde diferentes enfoques y que van desde simples factores de escala a procedimientos más complejos que controlan la velocidad (ver más adelante en el apartado 2.3.1).

El segundo problema se deriva de la necesidad de mover el robot a lo largo de un camino libre de colisión, tarea que resulta difícil y poco intuitiva ya que el robot se comanda a través de movimientos en el espacio Cartesiano. Esto incluso puede ser una tarea imposible, si no se presta asistencia cuando se trata de los robots redundantes o cuando los caminos libres de colisiones requieren algunos cambios en la configuración del robot, como en el ejemplo mostrado en la Figura 2.2, donde el robot debe cambiar de configuración de codo abajo a codo arriba para evitar chocar con el muro.

Asumiendo que se conoce la tarea que el usuario quiere teleoperar, se puede hacer uso de las técnicas de planificación de movimientos para guiarlo en el seguimiento del camino libre de colisión tanto de forma visual como háptica. La planificación normalmente se realiza en el espacio de configuraciones del robot y por consiguiente, con la finalidad de generar las fuerzas de guiado, se debe hacer uso de la cinemática directa y de la correspondencia entre los espacios de trabajo. Si el usuario sigue las instrucciones de movimiento sugerido, el robot puede seguir un camino libre, aunque el cruce de las configuraciones singulares tiene que ser manejado con cuidado. Si la ruta sugerida no es seguida de cerca, o si algún obstáculo se mueve en el espacio de trabajo (por ejemplo, otro robot), pueden ocurrir colisiones. Sin embargo estas colisiones pueden ser prevenidas ra-

lentizando la teleoperación con un aumento del factor de amortiguamiento en el controlador del nodo–operación cada vez que una posible colisión se detecta (computacionalmente, los movimientos del operador son bastante lentos y suaves, y por lo tanto se puede predecir con un cierto nivel de confianza).

El tercer problema nace de la necesidad de usar la cinemática inversa para relacionar los movimientos en el espacio cartesiano comandado con los movimientos ejecutados en el espacio articular del robot. Pueden existir múltiples soluciones de cinemática inversa, cuyo número aumenta con los grados de libertad de la cadena cinemática (Siciliano y Khatib, 2008), y por lo tanto se debe determinar la elección de la mejor alternativa. Algunos artículos en el área de la teleoperación pasan por alto este problema, y otros tratan de eliminar o minimizar su relevancia mediante el aprovechamiento de la redundancia, donde se aseguran grandes áreas de trabajo convexas de funcionamiento sin la singularidad (Peer et al., 2005), y otros tratan de encontrar soluciones aproximadas a la jacobiana inversa por medio de complejos esquemas de control robusto sobre la base de los mínimos cuadrados y la ponderación dinámica (Schinstock, 1998). Sin embargo, este problema puede ser resuelto fácilmente si se dispone de un camino que pueda ser sugerido kinestésicamente al usuario. En este caso, cerca de los cambios configuraciones, la sugerencia háptica puede reforzarse, haciendo que el robot cruce un punto crítico de la manera deseada, es decir, como se especifica por el camino.

El presente trabajo propone una combinación de varias de las ayudas antes mencionadas sobre el marco de teleoperación bilateral propuesto por Basañez et al. (2011). En particular un sistema de guiado basado en técnicas de planificación de movimientos para hacer frente a los problemas expuestos anteriormente y que tiene las siguientes características:

Asistencia de resincronización: La correspondencia de espacios de trabajo se basa en el método de resincronización y durante el accionamiento del embrague, ofrece una asistencia háptica al usuario que lo guía hacia la mejor configuración del dispositivo háptico para continuar la teleoperación.

Guiado libre de colisiones: Se utiliza un *PRM* en el espacio de configuración para planificar un camino libre de colisiones que, a través de

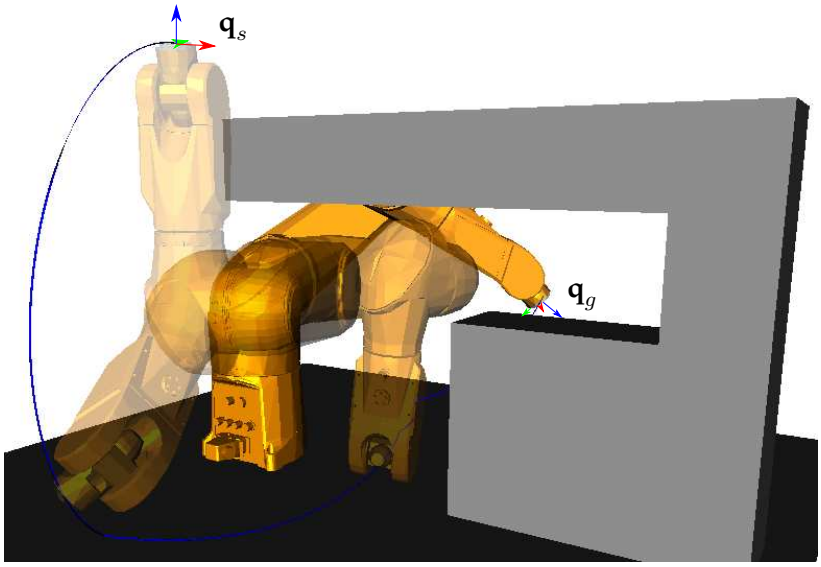


Figura 2.2: Ejemplo donde el brazo robot debe cambiar de configuración cinemática para llevar a cabo la tarea asignada y así evitar colisiones.

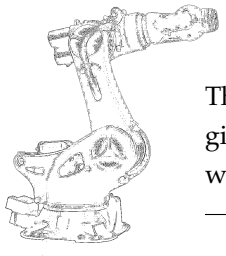
la cinemática directa y la correspondencia de los espacios de trabajo, sea utilizado para generar las fuerzas que sugieren al usuario los movimientos necesarios para teleoperar la tarea.

Guiado robusto: La robustez del módulo de guiado para conseguir que los cambios configuraciones que contiene el camino solución sea seguido exactamente, garantizando un cruce seguro de puntos críticos.

Comportamiento reactivo: Incrementa el valor del amortiguamiento del algoritmo de control en el nodo remoto cuando exista la posibilidad de colisiones con obstáculos, ya sean fijos o móviles, ralentizando la teleoperación con el fin de prevenirlas.

3

Teleoperación



The scientist is not a person who gives the right answers, he's one who asks the right questions.

Claude Lévi-Strauss

La teleoperación de robots manipuladores a través de dispositivos hápticos evita el problema de la diferencia cinemática entre el maestro (*master*) y el esclavo (*slave*) ya que la forma más común de teleoperación en estos casos es la de realizar una unión virtual rígida entre el punto de interface del dispositivo haptico HIP (*haptic interface point*) y el punto de control de la herramienta TCP (*tool center point*) del robot, que lleva el problema al espacio Cartesiano.

En la Figura 3.1 se pueden observar todos los elementos que intervienen en la teleoperación de robots antropomórficos industriales a través de dispositivos con realimentación háptica (en este caso realimentación de fuerzas). Forman parte del nodo–mando o local la imagen de video y el dispositivo háptico. Este video procede de una de las múltiples cámaras instaladas en el nodo–operación o remoto y que servirán para visualizar tanto el robot como los detalles de la tarea que se pretende realizar.

Este sistema de teleoperación cuenta con un sistema de control bilateral que permite asegurar tanto la calidad de los movimientos como la seguridad en la ejecución de los mismos y la reflexión de las fuerzas de teleoperación que se pueden presentar durante la actividad. Este sistema de control se describe en la Sección 3.1.

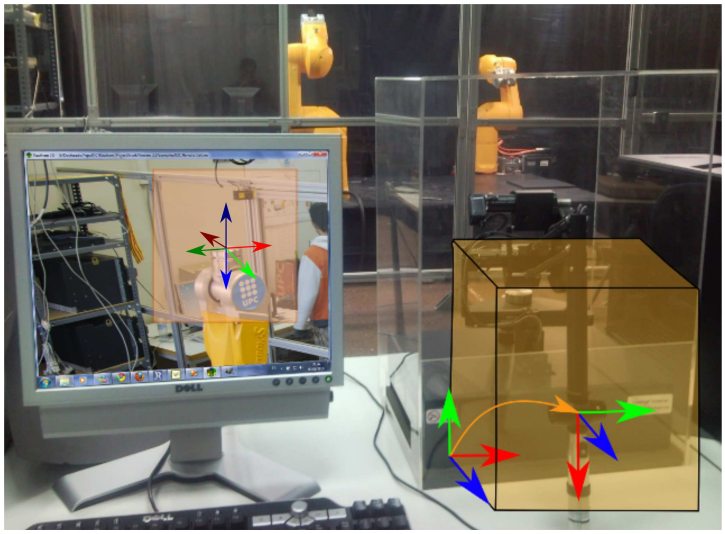


Figura 3.1: Estación local del usuario con las herramientas para la realización de tareas teleoperadas: (izq.) imagen realimentada por la cámara activa; (der.) el dispositivo háptico y su respectivo espacio de trabajo.

La primera y más evidente problemática que surge de este tratamiento, es la diferencia del tamaño de los espacios de trabajo, ya que el del dispositivo hápico es mucho más pequeño que el espacio de trabajo del robot que está teleoperando y por esta razón es necesario establecer una relación entre los movimientos realizados por el operador y los que realizará el robot teleoperado. El enfoque utilizado para resolver este problema se detalla en la Seccion 3.2.

3.1. Control

En la Figura 3.2 se puede apreciar el lazo de control básico utilizado en el sistema de teleoperación bilateral propuesto. Este marco de teleoperación bilateral está compuesto por el nodo–mando, el nodo–operación y el canal de comunicaciones. En el nodo–mando se encuentra el módulo del dispositivo háptico (HDM) que se encarga no sólo de obtener los movimientos del dispositivo (Dev) sino que realiza la correspondencia con la posición del robot, a través de los módulos Map de correspondencia entre espacios y el módulo IK para la cinemática inversa, y el sistema local de control o LCS (*Local Control System*). En el nodo–operación se encuentra el robot (Rob) y el sistema de control remoto o RCS (*Remote Control System*). Este sistema de

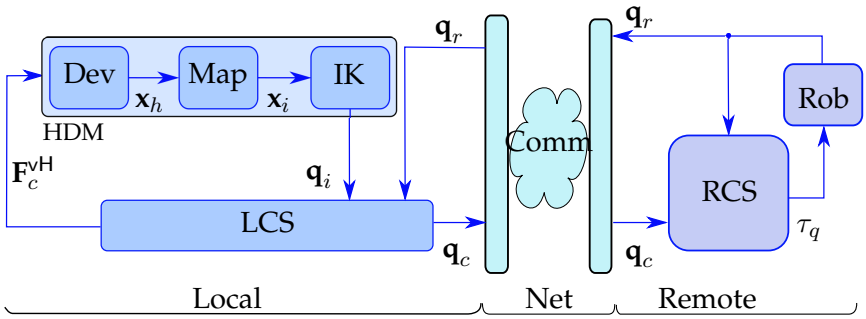


Figura 3.2: Sistema base de teleoperación bilateral.

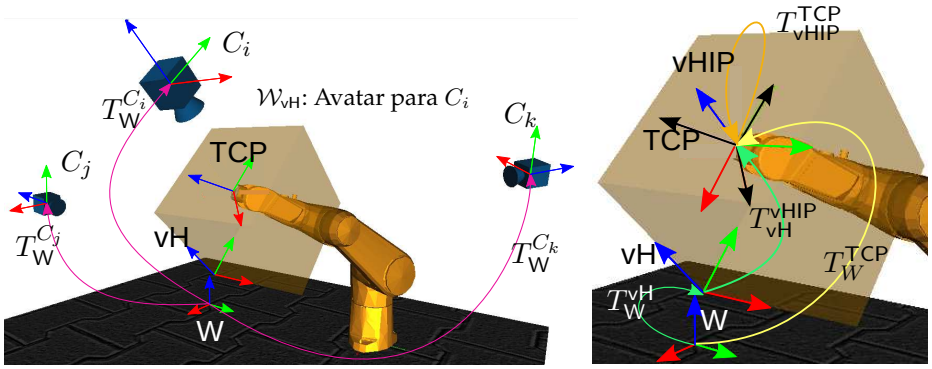
control está basado en la estrategia de control por impedancia en el que se utiliza como entrada la posición y se provee como salida una fuerza de realimentación. El controlador es un P+d (proporcional con amortiguamiento) que ha sido propuesto por Nuño et al. (2009), el cual ha demostrado ser estable y robusto frente a los retardos de la red de comunicaciones. Estos autores demostraron que los errores de posición y velocidad están acotados si las ganancias de control tanto en el nodo local ($K_l, B_l > 0$) como en el remoto ($K_r, B_r > 0$) se seleccionan cumpliendo la siguiente condición:

$$4B_lB_r > ({}^*T_l + {}^*T_r)^2 K_l K_r, \tag{3.1}$$

Se asume que tanto el operador humano como el entorno de trabajo son pasivos y que los retardos variables tienen límites superiores conocidos ($T_i(t) \leq {}^*T_i < \infty$) y con derivadas en el tiempo que no varían más rápido que el mismo tiempo ($|\dot{T}_i(t)| < 1$). Adicionalmente, estos autores demostraron que el control P+d puede reflejar de forma consistente las fuerzas del entorno remoto, y que para los pequeños retardos en el tiempo, proporciona una mayor transparencia que otras alternativas. Por estas ventajas se ha elegido este sistema de control para el desarrollo de este trabajo.

3.2. Correspondencia dispositivo háptico – robot

Dada la diferencia entre los espacios de trabajo de los dispositivos maestro (dispositivo háptico) y esclavo(robot) con los que se está realizando la teleoperación, es necesario establecer una correspondencia entre ellos. Para llevar a cabo esta correspondencia se debe tener en cuenta la forma en la que se realiza y en especial los elementos que juegan un papel importante



(a) Avatar: proyección en la escena.

(b) Unión virtual.

Figura 3.3: Modelo de la celda robótica remota para la ejecución de tareas teleoperadas equipado con un robot Stäubli TX90 y tres cámaras: a) La cámara i es la activa y la proyección del espacio de trabajo del dispositivo háptico está alineado con ella; b) La unión virtual se consigue haciendo coincidir los orígenes de los sistemas coordenados vHIP y TCP.

durante la teleoperación y en este caso, enfatizar el papel de la(s) cámara(s) que proveen la realimentación de video del sistema. Antes de considerar todos los procesos que tienen lugar durante la preparación y ejecución de una tarea robótica teleoperada, se definirán los sistemas coordenados de referencia y las transformaciones que permiten describir la ubicación de todos los elementos del entorno de trabajo, basado en el mostrado en la Figura 3.1 y modelado en la Figura 3.3, compuesto por un robot, un dispositivo háptico y varias cámaras, así:

- $\mathcal{W}_H$: Espacio de trabajo del dispositivo háptico.
  - H: Sistema coordenado de referencia $\mathcal{W}_H$.
  - HIP: Sistema coordenado de referencia asociado al efector final del dispositivo háptico.
  - $T_{\mathsf{H}}^{\mathsf{HIP}}$: Transformación entre H y HIP. Esta transformación es adquirida desde el dispositivo háptico.
- $\mathcal{W}_W$: Espacio de trabajo de la celda donde está localizado el robot.
  - W: Sistema coordenado de referencia $\mathcal{W}_W$.
  - TCP: Sistema coordenado de referencia asociado al efector final del robot.

  - T_W^{TCP}: Transformación entre W y TCP. Esta transformación se recalcula periódicamente durante la teleoperación y puede ser adquirida desde la controladora del robot.

- $\mathcal{W}_{\mathsf{vH}}$: Espacio de trabajo virtual del dispositivo háptico dentro de la escena, es decir que se proyecta $\mathcal{W}_H$ dentro de $\mathcal{W}_W$. Es también llamado **Avatar** del espacio de trabajo.

  - vH: Sistema coordenado de referencia $\mathcal{W}_{\mathsf{vH}}$.
  - vHIP: Sistema coordenado de referencia asociado a la proyección del efector final del dispositivo háptico en la escena.
  - $T_{\mathsf{vH}}^{\mathsf{vHIP}}$: Transformación entre vH y vHIP. Se calcula a partir de $T_{\mathsf{H}}^{\mathsf{HIP}}$ y usando los factores de escala.
  - $T_{\mathsf{W}}^{\mathsf{vH}}$: Transformación entre W y vH. Se define durante el proceso de sincronización.
  - $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$: Transformación entre vHIP y TCP. Se define durante el proceso de sincronización y es una rotación pura.

- $\mathcal{W}_{C_i}$: Espacio de trabajo de la cámara i (pueden haber varias cámaras disponibles para la teleoperación pero sólo una es elegida para realimentar las imágenes desde el sistema remoto y es llamada la cámara activa.)

  - C_i: Sistema coordenado de referencia de la cámara i.
  - $T_{\mathsf{W}}^{C_i}$: Transformación entre W y C_i. Esta transformación localiza la cámara i con relación al sistema coordenado de referencia global de la escena (W).

El principal objetivo de todo el marco de trabajo de teleoperación, es brindar al usuario la capacidad de mover el efector final TCP del robot mediante los movimientos realizados con el dispositivo háptico HIP en concordancia con las imágenes provenientes de la celda remota y proveídas por la cámara activa.

Para la correcta ejecución de una tarea teleoperada, es necesario dar solución a los siguientes problemas:

- **Correspondecia H–R**: Encontrar la relación existente entre el espacio de trabajo del dispositivo háptico y el del robot. Esto incluye:

- *Escalado*: Definir la relación entre el espacio físico y el virtual (espacio proyectado en la escena) o avatar del dispositivo háptico, es decir encontrar la expresión que define $T_{\mathsf{vH}}^{\mathsf{vHIP}}$ a partir de $T_{\mathsf{H}}^{\mathsf{HIP}}$ aplicando factores de escala tanto de rotación como traslación (Apartado 3.2.1).

- *Sincronización*: Encontrar la relación entre el espacio de trabajo avatar y el del robot, definido por la transformación $T_{\mathsf{W}}^{\mathsf{vH}}$ que relaciona los correspondientes sistemas coordenados de referencia, y la transformación $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$ que relaciona la proyección del vHIP y el TCP en el efector final. Esta sincronización hace uso del sistema coordenado de la cámara activa C_i como punto en común entre el dispositivo háptico y la escena (Apartado 3.2.2).

- **Teleoperar**: Encontrar la expresión que define T_{W}^{TCP} respecto a $T_{\mathsf{H}}^{\mathsf{HIP}}$. El usuario manipula el dispositivo háptico y por consiguiente obtiene cambios en $T_{\mathsf{H}}^{\mathsf{HIP}}$, es decir que cambia la posición y la orientación del HIP respecto a su propia base H. Estos movimientos deben ser trasladados al robot de forma que se produzcan cambios en el TCP respecto a su propio sistema coordenado de referencia W.

Efectuar la unión virtual significa que la posición del vHIP y la del TCP del robot sean coincidentes, al mismo tiempo que el espacio de trabajo del dispositivo háptico mantiene la alineación con el sistema coordenado de la cámara activa. Este requerimiento se hace para ofrecer al operador humano una correlación natural entre los movimientos que va a ejecutar con su mano en el dispositivo háptico y los movimientos resultantes en el efector final del robot y que van a ser observados a través de las imágenes provenientes de la cámara activa. En este enfoque la cámara activa reviste vital importancia ya que es el elemento que provee el punto de conexión entre la escena observada y el espacio de trabajo del dispositivo háptico.

Para llevar a cabo esta unión virtual satisfactoriamente, se han diseñado estas dos directivas:

- *Directiva de posición:* Hacer que los orígenes de los sistemas coordenados TCP y vHIP sean coincidentes y considerar una transformación puramente rotacional entre ellas dos $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$ (Figura 3.3b).

- *Directiva de orientación:* Hacer que la orientación del espacio de trabajo proyectado en la escena vH y la de la cámara activa sean coincidentes, es decir que[1] $ori(T_{\mathsf{W}}^{\mathsf{vH}}) = ori(T_{\mathsf{W}}^{C_i})$ (Figura 3.3a).

En este enfoque prima la relación existente entre los movimientos realizados por el operador y los observados a través de la cámara activa. En enfoques anteriormente utilizados, la orientación de espacio de trabajo del dispositivo háptico se hace coincidir con el de la base del robot de forma que los movimientos comandados tienen relación directa con lo comandados a través de la interface de control del propio robot. Aquí se busca un enfoque más global de la interacción del robot con su entorno y a su vez, hacer más flexible la relación entre el robot remoto y su teleoperador, haciendo que se pueda conectar y/o desconectar la teleoperación en cualquier instante donde el robot esté detenido.

Como se aprecia en la Figura 3.3b, la transformación T_W^{TCP} que se utilizará durante la fase de teleoperación activa, se calcula de la siguiente manera:

$$T_W^{\mathsf{TCP}} = T_{\mathsf{W}}^{\mathsf{vH}} \cdot T_{\mathsf{vH}}^{\mathsf{vHIP}} \cdot T_{\mathsf{vHIP}}^{\mathsf{TCP}} \tag{3.2}$$

Donde:

- $T_{\mathsf{vH}}^{\mathsf{vHIP}}$ se recalcula a partir de $T_{\mathsf{H}}^{\mathsf{HIP}}$, como se detalla en el Apartado 3.2.1, cada vez que $T_{\mathsf{H}}^{\mathsf{HIP}}$ es actualizada desde el dispositivo háptico (el período de actualización de este dispositivo es aproximadamente 1 ms).

- $T_{\mathsf{W}}^{\mathsf{vH}}$ y $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$ son calculadas, como se detalla en el Apartado 3.2.2, cada vez que se requiera una nueva sincronización.

3.2.1. Escalado

$T_{\mathsf{vH}}^{\mathsf{vHIP}}$ es calculada a partir de $T_{\mathsf{H}}^{\mathsf{HIP}}$. Una vez que la transformación $T_{\mathsf{H}}^{\mathsf{HIP}}$ es leída del dispositivo háptico, las partes de traslación y de rotación son escaladas independientemente de forma que permitan a $\mathcal{W}_{\mathsf{vH}}$ cubrir un cierto volumen de $\mathcal{W}_W$. El factor de escalado rotacional es aplicado directamente al ángulo, manteniendo sin cambios el eje de la rotación.

[1] Si T es una transformación homogenea, $pos(T)$ y $ori(T)$ son utilizadas para representar respectivamente la posición y la orientación.

Sea $AA(\mathbf{v}, \theta)$ la representación eje-ángulo de una rotación de un ángulo θ alrededor de un eje $\mathbf{v} = (v_x, v_y, v_z)$, y $\mathcal{S}_t$ y $\mathcal{S}_r$ son los factores de escala traslacional y rotacional respectivamente. Entonces:

$$\begin{aligned} AA(\mathbf{v}, \theta) &= ori(T_{\mathsf{H}}^{\mathsf{HIP}}) \\ ori(T_{\mathsf{vH}}^{\mathsf{vHIP}}) &= AA(v_x, v_y, v_z, \mathcal{S}_r \cdot \theta) \\ pos(T_{\mathsf{vH}}^{\mathsf{vHIP}}) &= pos(T_{\mathsf{H}}^{\mathsf{HIP}}) \cdot \mathcal{S}_t \end{aligned} \tag{3.3}$$

El ángulo escalado que será barrido por el avatar se limita dentro del rango $[0; \pm 2\pi)$, teniendo en cuenta el sentido del giro, de forma que el usuario sienta y ejerza control de los movimientos angulares más intuitivamente. Esto quiere decir que aunque el usuario elija un valor no adecuado para el factor de escala rotacional, por ejemplo muy grande, y realice amplios movimientos de rotación en el dispositivo háptico, el ángulo de rotación nunca excederá el valor de 2π en el sentido que haya elegido, con un único movimiento.

3.2.2. Sincronización

Al momento de inicializar la teleoperación o cuando el usuario quiera cambiar la cámara activa, el escalado de la correspondencia entre espacios de trabajo o la posición de la projección del espacio de trabajo del dispositivo háptico dentro de la escena, el sistema necesita ser sincronizado para crear una nueva conexión virtual entre el robot y el dispositivo háptico. Además de permitir los movimientos del robot a través de los movimientos del dispositivo háptico, esta conexión permite una correlación simple y directa entre los movimientos realizados en el dispositivo háptico con aquellos que realiza el robot y que son vistos a través de las imágenes que realimenta la cámara activa desde el nodo remoto. Esta sincronización se lleva a cabo siguiendo las dos directivas de la Sección 3.2 y realizando los siguientes dos pasos:

1. Determinar $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$: Como se ha descrito en la **directiva de posición**, esta transformación es rotacional pura. Aplicando ahora la **directiva**

de orientación ($ori(T_{\mathsf{W}}^{\mathsf{vH}}) = ori(T_{\mathsf{W}}^{C_i})$), $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$ se calcula como sigue[2]:

$$\begin{aligned} T_{\mathsf{vHIP}}^{\mathsf{TCP}} &= \left(\begin{array}{c|c} \mathcal{R}_{\mathsf{vHIP}}^{\mathsf{TCP}} & \mathbf{0} \\ \hline 0 & 1 \end{array}\right) \\ \mathcal{R}_{\mathsf{vHIP}}^{\mathsf{TCP}} &= (\mathcal{R}_{\mathsf{vH}}^{\mathsf{vHIP}})^{-1} \cdot \mathcal{R}_{\mathsf{vH}}^{\mathsf{TCP}} \\ &= (\mathcal{R}_{\mathsf{vH}}^{\mathsf{vHIP}})^{-1} \cdot (\mathcal{R}_{\mathsf{W}}^{\mathsf{vH}})^{-1} \cdot \mathcal{R}_{\mathsf{W}}^{\mathsf{TCP}} \\ &= (\mathcal{R}_{\mathsf{vH}}^{\mathsf{vHIP}})^{-1} \cdot (\mathcal{R}_{\mathsf{W}}^{\mathsf{Ci}})^{-1} \cdot \mathcal{R}_{\mathsf{W}}^{\mathsf{TCP}} \end{aligned} \tag{3.4}$$

Donde $\mathcal{R}_{\mathsf{vH}}^{\mathsf{vHIP}}$ es el valor de la rotación de vHIP leída después de reposicionar el dispositivo háptico (cuando el robot y el dispositivo háptico se vuelven a vincular); $\mathcal{R}_{\mathsf{W}}^{\mathsf{Ci}}$ es la rotación de la cámara seleccionada como activa; y $\mathcal{R}_{\mathsf{W}}^{\mathsf{TCP}}$ es el valor de la rotación del efector final del robot TCP en el instante de la sincronización (cuando el robot fue desvinculado del dispositivo háptico).

2. Determinar $T_{\mathsf{W}}^{\mathsf{vH}}$: Una vez se ha calculado $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$ según la Ec. (3.4), se utiliza para actualizar la transformación $T_{\mathsf{W}}^{\mathsf{vH}}$ como sigue:

$$T_{\mathsf{W}}^{\mathsf{vH}} = T_{W}^{\mathsf{TCP}} \cdot (T_{\mathsf{vHIP}}^{\mathsf{TCP}})^{-1} \cdot (T_{\mathsf{vH}}^{\mathsf{vHIP}})^{-1} \tag{3.5}$$

3.3. Ejecución de movimientos

Durante la teleoperación activa, o sea cuando se están comandando movimientos, el sistema está gobernado por la Ec. (3.2), donde se puede apreciar que los cambios en $T_{\mathsf{vH}}^{\mathsf{vHIP}}$ son convertidos en movimientos de T_{W}^{TCP}. Cada vez que se lleva a cabo una nueva sincronización, las transformaciones $T_{\mathsf{W}}^{\mathsf{vH}}$ como $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$ son también actualizadas. La aplicación de este modelo de correspondencia para la teleoperación brinda la posibilidad de mejorar la ejecución de las tareas teleoperadas en estas tres direcciones:

- Una mejora de la visibilidad, que se puede conseguir al cambiar de cámara activa.
- Una mejora de la alcanzabilidad, que se consigue al mover la proyección del espacio de trabajo del dispositivo háptico (**avatar**) dentro de

[2] $\mathcal{R}$ representa la matriz de rotación de una transformación homogénea T, es decir $\mathcal{R} = ori(T)$.

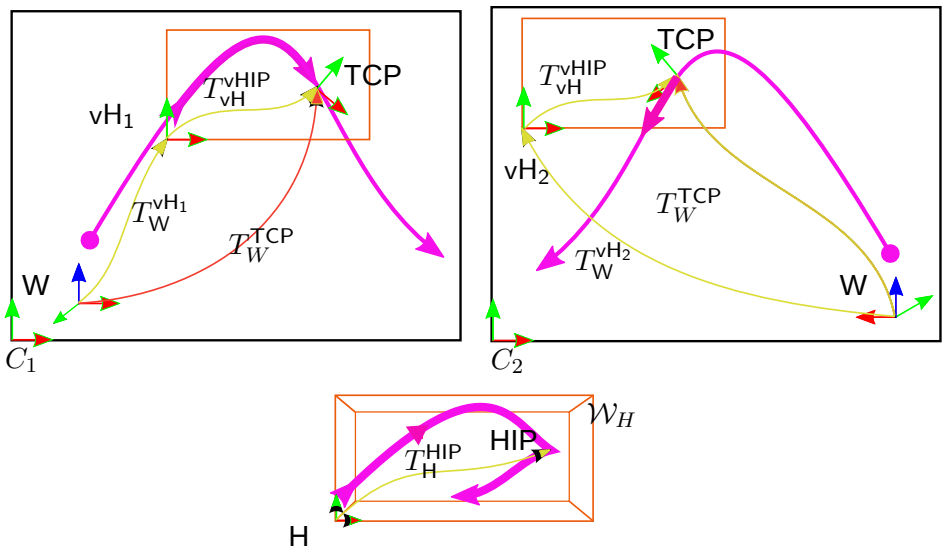


Figura 3.4: Cambio entre dos cámara enfrentadas: Las figuras superiores muestran en magenta el recorrido del TCP como es visto desde la cámara (en el de la izquierda, cuando es visto desde la cámara C_1 y en el de la derecha cuando es visto desde la cámara C_2).

la escena.

- Un aumento o disminución de la precisión de los movimientos del robot, que se consigue al cambiar los factores de escala de la correspondencia entre el dispositivo háptico y el robot.

Como ejemplo del primer caso, la Figura 3.4 muestra los movimientos realizados por el operador humano con el HIP cuando éste decide cambiar entre dos cámaras, siendo puestas una en frente de la otra y cubriendo en su mayoría el mismo volumen de la escena (prestar atención en la posición y orientación del sistema coordenado W). Aquí la región cubierta por el espacio de trabajo virtual del dispositivo háptico se muestra como un rectángulo de color naranja. En la figura de abajo se muestra en magenta el camino recorrido por el HIP dentro del espacio de trabajo real del dispositivo háptico, este se mueve hacia la derecha cuando se utiliza la cámara C_1 y luego se mueve para la izquierda cuando se utiliza la cámara C_2.

Como ejemplo del segundo caso, la Figura 3.5 muestra el movimento del sistema coordenado HIP cuando el usuario reposiciona el espacio de trabajo avatar del dispositivo háptico. En la figura inferior muestra el camino recorrido por el HIP dividido en tres partes: la primera y la últi-

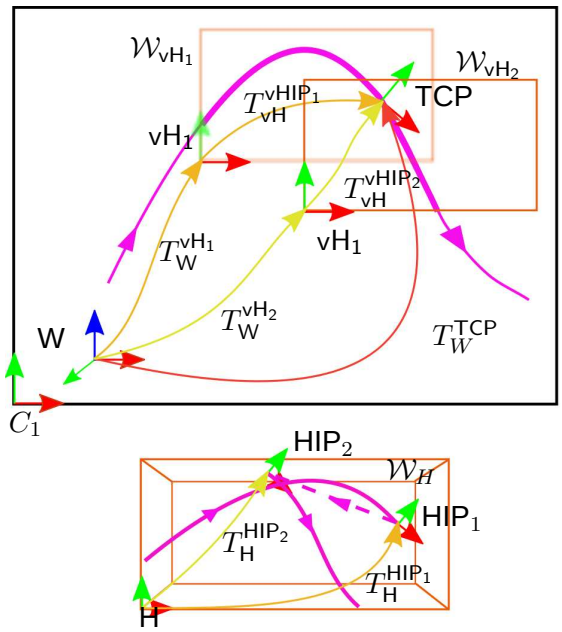


Figura 3.5: Desplazamiento del espacio de trabajo proyectado: La figura superior muestra en magenta el camino recorrido por el TCP y el espacio virtual del dispositivo háptico proyectado en la escena ubicado en dos posiciones diferentes (rectángulos de color naranja).

ma parte corresponden a una teleoperacion activa, o sea cuando la unión virtual está establecida, y la parte rectilínea central con trazo discontinuo, corresponde al movimiento del HIP mientras el robot se encuentra desconectado del mismo.

Más adelante en el Capítulo 5 y en especial en la Sección 5.3, se explora la forma de brindar ayudas automatizadas al teleoperador que le den la posibilidad de aplicar las estrategias anteriormente expuestas y en particular, ayudarle a buscar la mejor posición del avatar para llevar a cabo las resincronizaciones.

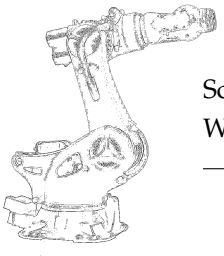
3.4. Conclusiones

Se ha presentado la forma directa y sencilla de relacionar los espacios de trabajo del dispositivo háptico y del robot utilizando el punto de vista de la cámara activa que realimenta las imágenes desde el nodo–operación y que el teleoperador usa para visualizar en la tarea. Con esta relación se ofrece un método para cambiar de punto de vista dentro del nodo–operación y así

poder hacer frente a eventuales oclusiones de la visual de la tarea o de aumentar o disminuir la precisión de los movimientos a través de las escalas. Este método aporta flexibilidad a la hora de vincular el dispositivo háptico al sistema de referencia que se está teleoperando con el robot remoto, permitiendo desconectar y reconectar este vínculo una vez se haya posicionado el espacio avatar en un lugar más adecuado para proseguir con la tarea.

4

Planificación de movimientos



Science is organized knowledge.
Wisdom is organized life.

Immanuel Kant, Critique of
Pure Reason

En este capítulo se presenta el sistema de planificación que es utilizado para obtener el camino solución de la tarea que va a ser realizada por los robots manipuladores durante la teleoperación. En primer lugar se describe la forma en la que interactúa este sistema dentro del marco de teleoperación y posteriormente se detallan las diferentes partes en las que se divide el planificador.

4.1. Planificación y teleoperación

Dentro del marco donde se desarrolla este trabajo, la planificación del camino solución a la tarea teleoperada, se utiliza como herramienta de soporte al teleoperador con la finalidad de brindarle la ayuda adecuada para desarrollar la tarea sin tener en cuenta los detalles mecánicos y constructivos del robot con la que se ejecuta. El camino encontrado cumple con los requisitos de ser libre de colisión con el entorno para el instante inicial. Este camino inicial es la semilla de los procesos iterativos de validación y replanificación que se llevan a cabo de forma paralela a la ejecución de la tarea por parte del operador y que además de mantener la característica de ser libre de colisión, incorpora comportamientos reactivos frente las oclu-

siones temporales que los obstáculos móviles puedan ocasionar al camino deseado $\mathbf{P}_d^q$.

La ejecución de tareas por parte de un operador humano, que forma parte del lazo de control del sistema, tienen la característica de no seguir un patrón claramente definido en el tiempo. Los parámetros temporales no están definidos de forma determinística, dado que no se establecen ni las velocidades de los movimientos ni las posibles pausas que se puedan presentar durante ella, ocasionados ya sea por procesos cognitivos del usuario como la toma de decisiones sobre la evolución de la tarea, o de los retardos propios de la información de control proveniente del nodo–mando.

El planificador propuesto en este trabajo se ha desarrollado teniendo en cuenta las características de ejecución de las tareas teleoperadas de forma que se aprovecha la flexibilidad temporal con la que se desarrollan. En un primer instante se planifica teniendo en cuenta la parte estática del entorno y luego se incorpora un comportamiento reactivo frente a los cambios de la configuración inicial a medida que se ejecuta el plan inicial. Estos cambios son tenidos en cuenta si ocurren dentro de un radio de acción temporal definido por la posición actual del robot y una medida ponderada de la velocidad con la que se lleva a cabo la tarea.

Por otro lado, al ser el operador el elemento inteligente del lazo de control de todo el sistema, siempre es necesaria la aceptación de un nuevo camino que la ejecución propia del planificador puede arrojar como resultado a una modificación en el entorno del trabajo y que impliquen grandes cambios sobre el plan inicialmente aceptado por el usuario.

4.2. Planificación multirobot

La planificación en un entorno multirobot implica la búsqueda de un camino solución que pueda llevar a cada uno de los robots a cumplir su tarea específica. En los mismos términos expuestos a lo largo de este trabajo, esto significa encontrar para cada robot el camino que conduce de su configuración inicial a su configuración objetivo evitando las posibles colisiones con el entorno y con los demás robots.

En las tareas ejecutadas de forma automática, existen criterios de temporización entre el inicio y fin de las tareas y de la ejecución de cada una de ellas, de manera que se pueden iniciar en el mismo instante, establecer

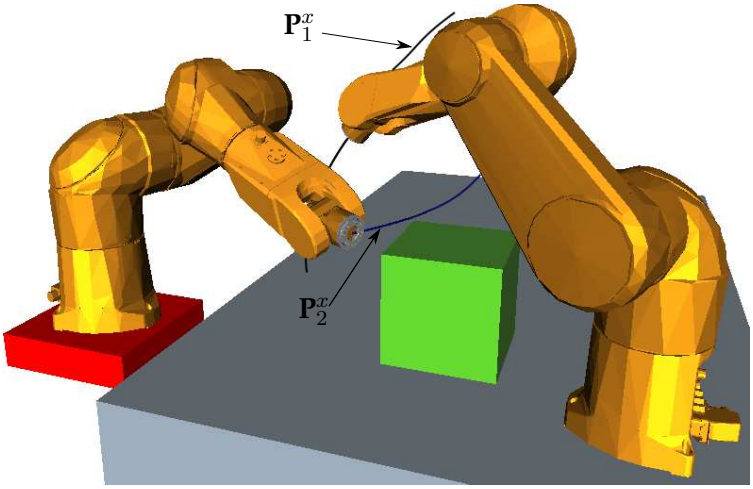


Figura 4.1: Ejemplo de problema de planificación con dos robots donde se pueden observar los caminos $\mathbf{P}^x$ que se desean guiar en cada uno de ellos.

retardos en el inicio de alguna de ellas y/o definir las velocidades de ejecución. Estableciendo estos criterios, la planificación permite encontrar los caminos y las soluciones más adecuadas que los cumplan. Otra estrategia es hacer la planificación en el espacio de configuraciones $\mathcal{C}$ conjuntos que involucren todos los grados de libertad de los robots y en el que cada punto del espacio representa una configuración particular de todos los robot implicados en el problema. Este enfoque genera espacios de configuración de un número de grados de libertad considerable y que pueden representar grandes tiempos de cómputo que no pueden ser asumidos en tiempo real.

En el caso de que cada robot estuviera teleoperado por un operador no se garantizaría la ausencia de colisiones a pesar de que siguieran el camino planificado porque el *tempus* lo marca independientemente cada operario.

Se deben establecer por lo tanto estrategias para llevar a cabo esta planificación de forma individualizada para cada uno de los robots, considerando los otros robots como obstáculos móviles, estableciendo criterios que permitan la ejecución simultanea de todas las tareas. Estos criterios se basarán en la replanificación y en la modificación del amortiguamiento en el algoritmo de control de la teleoperación.

Este enfoque mantiene la dimensión del espacio de configuraciones $\mathcal{C}$ de cada robot donde se llevan a cabo las búsquedas del camino solución con el menor número posible, de forma que los tiempos de cómputo de las soluciones y en especial de las replanificaciones, tiendan a disminuir ha-

ciendo posible su incorporación en sistemas a tiempo real. Un sistema de planificación con estas características puede ser utilizado como herramienta de apoyo durante la ejecución de tareas teleoperadas.

El planificador propuesto en este trabajo, encuentra una solución a la tarea establecida para un único robot de los que conforman el problema, y mantiene la evaluación de su factibilidad dentro de los rangos de acción temporal de cada uno de ellos, es decir, que mientras exista actividad por parte del robot que ha sido objeto de la planificación, existe la necesidad de evaluar continuamente si el camino inicialmente propuesto sigue siendo válido frente a los cambios que se presentan en su entorno y muy particularmente dentro del radio de acción que se establece con relación directa a su posición y velocidad. Esta vigilancia constante activa cuando hace falta la replanificación y/o la modificación del amortiguamiento.

4.3. Propuesta

Los métodos de planificación basados en muestreo han demostrado ser muy útiles frente a los métodos completos donde se determina analíticamente el espacio de configuraciones asociado, y esta utilidad se incrementa en la medida que crece el número de grados de libertad involucrados. Una característica interesante de estos métodos es que se va descubriendo el espacio de configuraciones a medida que se extraen muestras y se valida su estado de colisión o no. En este capítulo se describe el método propuesto para planificar la tarea que se va a realizar con el robot teleoperado, que comparte su espacio de trabajo total o parcialmente con otros robots en el nodo–operación y que son considerados como obstáculos móviles, que pueden afectar el desarrollo de la tarea, para lo cual se responde con una replanificación de la solución cuando el teleoperador así lo desee.

La planificación se lleva a cabo con un mapa de carreteras (*RoadMap*) que cambia dinámicamente frente a los cambios de los obstáculos móviles (este mapa de carreteras dinámico siempre se centra sobre el último camino solución propuesto). La eficiencia se logra a través de la evaluación tardía (*Lazy approach*), es decir que el grafo que representa el mapa de carreteras es un subgrafo de uno más grande sin evaluar y que cubre todo el espacio de configuraciones. En este subgrafo solamente son evaluados aquellos vértices y aristas que han contribuido a encontrar el camino solución.

4.3.1. Planificación basada en muestreo

Los métodos de planificación basados en muestreo han probado ser eficientes en la solución de problemas que involucran gran cantidad de grados de libertad, extendiendo su uso a una amplia variedad de problemas de diferente complejidad (ver Apartado 2.1.1). Los planificadores de tipo mapa de carreteras probabilísticos (*PRM*) están enfocados principalmente a problemas donde se realizan multiples búsquedas, mientras que aquellos basados en árboles aleatorios (*RRT*) están más enfocados a una única pregunta sobre la cual se va construyendo la solución (ver Capítulo 2). Esta característica, entre otras, es determinante en la elección del tipo más conveniente de planificador a utilizar en la resolución de un tipo específico de tarea.

Por otra parte, no sólo la topología de la conectividad de las muestras, sino las muestras mismas son decisivas en el desempeño y efectividad de los planificadores. Existen varios estudios que evalúan la contribución de la generación de las muestras en el desempeño global del planificador. El muestreo aleatorio es un esquema ampliamente utilizado, pero algunas veces son más deseables otras características, como ubicar las muestras uniformemente con una mejor discrepancia, o cerca de los obstáculos y los pasajes estrechos, con el ánimo de disminuir el número de muestras utilizadas con la consecuente mejora en la eficiencia computacional. Tambien en vistas a mejorar dicha eficiencia, el procedimiento de evaluación tardía (*Lazy*) que posterga lo más posible el costoso proceso de validación de colisión es de gran interés.

4.3.2. Nomenclatura

Antes de continuar con la descripción de los algoritmos de planificación de los cuales trata este capítulo, es necesario definir algunos de los términos que forman parte de ellos. Sea:

$\mathcal{C}$**:** Espacio de configuraciones.

S**:** Conjunto de muestras en el espacio de configuraciones.

$\mathbf{q}_s$**:** Configuración inicial del robot.

$\mathbf{q}_g$**:** Configuración objetivo.

$\mathbf{P}^q$: Conjunto de nodos y aristas que constituyen un camino en $\mathcal{C}$.

$\mathbf{P}_i^q$: Camino candidato a ser una solución que se encuentra en proceso de evaluación.

$\mathbf{P}_d^q$: Camino evaluado libre de colisión que satisface los requerimientos de la tarea.

K_{neig}: Índice de conectividad que describe la cantidad de vecinos con los cuales se debe conectar una muestra.

G_T: Grafo de cubrimiento total de $\mathcal{C}$.

G_W: Subgrafo de G_T constituido por los nodos de $\mathbf{P}_i^q$ y sus sucesores.

N_{pd}: Conjunto de nodos de un $\mathbf{P}^q$ y sus sucesores.

v_i: Vértice o nodo i de un grafo.

e_{ij}: Arista existente entre los vértices i y j de un grafo.

Adicionalmente, se definen cuatro funciones para el manejo de grafos:

CComp$(\mathbf{q})$**:** Devuelve la componente conectada del grafo que contiene $\mathbf{q}$.

UpdateGraph(G, v_n, e_d)**:** Devuelve el grafo G con el conjunto de nuevos vértices v_n agregados, y el conjunto de aristas existentes e_d borradas.

Succ$(\mathbf{P}^q)$**:** Devuelve los nodos de G_T que se encuentran conectados directamente de los nodos de $\mathbf{P}^q$.

SelectNodes$(G_T, \mathbf{P}^q, level)$**:** Devuelve un conjunto formado por los nodos de $\mathbf{P}^q$, sus sucesores y los sucesores de ellos hasta el nivel determinado por $level$, es decir $N_p = \{v_i \in \mathbf{P}^q \cup \text{Succ}(\mathbf{P}^q) \cup \text{Succ}(\text{Succ}(\mathbf{P}^q)) \ldots\}$.

4.3.3. Enfoque

El presente trabajo está basado en las técnicas de planificación con *PRM* y evaluación tardía de las colisiones, esto quiere decir que el camino solución sólo se valida completamente como libre de colisión hasta el último momento. Este camino se sugerirá posteriormente al teleoperador, pero éste no esta obligado a seguirlo, así que el planificador tiene la característica

Algoritmo 1 FollowPath

Require: $\mathbf{q}_s$, $\mathbf{q}_g$
  $(\mathbf{P}_d^q, G_T)$=FindPath($\mathbf{q}_s$, $\mathbf{q}_g$, $\emptyset$, $\emptyset$)
  while $\mathbf{q}_c \neq \mathbf{q}_g$ AND $\mathbf{P}_d^q \neq \emptyset$ **do**
    $\mathbf{q}_c = \text{Move}(\mathbf{P}_d^q)$
    $(\mathbf{P}_d^q, G_T)$=FindPath($\mathbf{q}_s$, $\mathbf{q}_g$,$\mathbf{P}_d^q$,G_T)
  end while

de replanificación para adaptarse a los movimientos reales del usuario así como a los obstáculos móviles.

El procedimiento que se lleva a cabo para encontrar el camino entre $\mathbf{q}_s$ y $\mathbf{q}_g$ y su utilización en tareas de teleoperación, se define en el procedimiento FollowPath del Algoritmo 1. En este procedimiento se describe globalmente el funcionamiento del planificador como un proceso paralelo a la ejecución del movimiento teleoperado.

El algoritmo FindPath es usado iterativamente por el procedimiento FollowPath, mostrado en el Algoritmo 1. Primero, se busca el camino, y si se encuentra una solución válida, se inicial el movimiento sugerido por ese camino, y de ser necesario se vuelve a calcular un nuevo camino. Los sucesivos llamados a FindPath, pueden dar diferentes soluciones dependiendo de la existencia de obstáculos móviles o de los movimientos efectivamente ya realizados por el robot. La función Move en el Algoritmo 1 esquematiza la evolución del seguimiento del camino sugerido al teleoperador. También puede programarse una ejecución automática (no teleoperada) en cuyo caso la función Move realiza un paso del movimiento a lo largo del camino actual.

La primera vez que se llama al procedimiento FindPath, se hace sólo con las configuraciones inicial y objetivo, $\mathbf{q}_s$ y $\mathbf{q}_g$ respectivamente, y si este método arroja como resultado un camino válido para ser teleoperado $\mathbf{P}_d^q$ y mientras el robot no haya alcanzado la configuración objetivo $\mathbf{q}_g$ se actualiza la posición actual del robot $\mathbf{q}_c$ y se procede a la validación continua de ese camino deseado $\mathbf{P}_d^q$ usando la misma función FindPath. Este nuevo llamado se hace con $\mathbf{P}_d^q$ y G_T como parámetros, de forma que se inicia una iteración de validación de $\mathbf{P}_d^q$ y en caso de resultar ahora no válida, se procede a encontrar un nuevo camino candidato $\mathbf{P}_i^q$ que comparta la mayor cantidad de nodos y aristas de $\mathbf{P}_d^q$, es decir, con la menor cantidad de cam-

bios posibles. El funcionamiento general del planificador implementado en la función FindPath se puede describir a través de los siguientes pasos:

1. Se crea un grafo de cubrimiento G_T construido con un conjunto inicial de muestras no evaluadas (S), obtenidas con una secuencia Halton (secuencia de baja discrepancia) sobre todo el espacio de configuraciones $\mathcal{C}$, donde cada una de las muestras está conectada con sus K_{neig} vecinos más cercanos, usando la métrica respectiva del espacio de configuraciones del problema, mediante aristas no evaluadas.

2. Se crea un grafo de trabajo G_W como subgrafo de G_T a partir de los nodos de un camino solución $\mathbf{P}^q$. Inicialmente no existe dicho camino y entonces $G_W = G_T$ (ver Apartado 4.4.1).

3. Se busca en G_W un camino de mínimo coste mediante el algoritmo A*.

4. Se evalúa el camino siguiendo técnicas de evaluación tardía (ver Apartado 4.4.2).

La estrategia de remuestreo se lleva a cabo en los Pasos 2 y 4. En el primero de ellos se realiza una generación de muestras en todo el espacio de configuraciones que garantiza la conectividad de G_T siguiendo la secuencia inicial Halton, y en el segundo caso se obtienen muestras adicionales en las cercanías de los obstáculos sobre los cuales cae una arista no válida. En los Pasos 1 y 4, en el caso de que exista un $\mathbf{P}_d^q$ calculado en una iteración anterior tiene la particularidad de que:

1. Actualiza la evaluación de nodos alrededor de la configuración inicial (ver Apartado 4.4.3).

2. Si la evaluación de los nodos de $\mathbf{P}_i^q$ no presenta cambios frente a $\mathbf{P}_d^q$, el proceso de replanificación retorna exactamente el mismo $\mathbf{P}_d^q$ utilizado como parámetro, o en caso contrario devuelve un nuevo camino $\mathbf{P}_d^q$ con el menor numero de variaciones posibles con respecto al anterior.

4.3.4. Evaluación de nodos y aristas

La evaluación de los nodos, a diferencia de un *PRM* convencional, no se hace directamente para saber si forman parte o no del espacio libre de $\mathcal{C}$.

En este caso se va a dar un valor continuo entre 0 y 1 a manera de "probabilidad de ser obstáculo" basada en la mínima distancia del robot a los obstáculos de su entorno.

Esta distancia a los obstáculos tanto estáticos cómo dinámicos, se calcula en el espacio operacional y se utiliza para evaluar las muestras y también las aristas que se establecen entre ellas. Esta función de evaluación H de una muestra se define como:

$$H = \begin{cases} 1 & \text{si } d_{min} \leq d_{th} \\ \frac{d_{th}}{d_{min}} & \text{si } d_{min} > d_{th} \end{cases} \tag{4.1}$$

con d_{min} la distancia mínima (positiva) entre el robot y los obstáculo y d_{th} una distancia umbral de seguridad al rededor de los obstáculos (para asegurar caminos con una mínima holgura). La muestras evaluadas con $H = 1$ serán removidas del grafo sólo si corresponden a obstáculos estáticos, dado que la oclusión causada por los obstáculos dinámicos es temporal.

Las aristas del grafo tendrán asignado el siguiente costo:

$$\text{cost}(i, j) = \text{dist}(v_i, v_j) + H_{obs} \cdot |H_i - H_j| + C_p, \tag{4.2}$$

donde $\mathsf{dist}(v_i, v_j)$ es la distancia entre los dos vértices en $\mathcal{C}$ con la métrica del respectivo espacio, H_{obs} es un peso de escala, $C_p = 0$ si v_i y v_j forman parte del último camino calculado, y $C_p > 0$ en cualquier otro caso. Esta selección de costos orienta la búsqueda hacia el camino más corto con suaves cambios en el valor de H, y con la menor cantidad de cambios con respecto al anteriormente calculado, cuando hay uno disponible.

4.4. Partes del planificador

4.4.1. Selección del grafo de trabajo G_W

Los algoritmos de planificación de trayectorias deben manejar una gran cantidad de muestras cuando el espacio de configuraciones tiene un número elevado de dimensiones. De aquí que la correcta selección de muestras es un aspecto muy importante que mejora la efectividad de un algoritmo en particular.

Una forma de contribuir a minimizar este perjudicial efecto, se logra

Algoritmo 2 WorkGraph

Require: G_T, $\mathbf{P}^q$, $level$
Ensure: G_W
  $N_{pd} = \text{SelectNodes}(G_T, \mathbf{P}^q, level)$
  $G_W = \text{SetGraph}(N_{pd}, K_{neig})$
  if $\text{CComp}(G_W, \mathbf{q}_s) \neq \text{CComp}(G_W, \mathbf{q}_g)$ **then**
    while $\text{CComp}(G_T, \mathbf{q}_s) \neq \text{CComp}(G_T, \mathbf{q}_g$ **do**
      $G_T = \text{SetGraph}(\text{nodes}(G_T) \cup \text{Sampling}(N_{samp}), K_{neig})$
    end while
    $G_W = G_T$
  end if
  return G_W

al reducir el número de muestras del grafo donde se realiza la búsqueda. Siguiendo esta idea, en este trabajo se crea un subgrafo G_W a partir del grafo G_T (grafo más grande, uniformemente distribuido sobre el espacio de configuraciones $\mathcal{C}$ y que no ha sido evaluado aún) donde efectivamente se realiza la búsqueda. Dicho grafo se crea en el procedimiento WorkGraph mostrado en el Algoritmo 2, que se explica a continuación.

Dado el grafo total G_T y un $\mathbf{P}^q \in G_T$, el Algoritmo 2 construye un grafo G_W con los nodos y sus sucesores hasta un nivel $level$ usando una vecindad K_{neig} en donde G_W puede tener aristas no existentes en G_T.

El procedimiento WorkGraph construye G_W con base en el conjunto de vértices de $\mathbf{P}^q$ pasado como parámetro y por sus sucesores hasta el nivel $level$ obtenidos con la función SelectNodes. $\mathbf{P}^q$ será un camino solución obtenido de una iteración anterior si existe, o en caso contrario será el camino candidato actual $\mathbf{P}^q_i$. La primera vez que se llama este procedimiento $\mathbf{P}^q = \emptyset$ y por lo tanto el resultado será $G_W = G_T$. El algoritmo garantiza la devolución de un grafo en el que $\mathbf{q}_s$ y $\mathbf{q}_g$ forman parte de una misma componente conectada. Es decir, si las configuraciones de inicio y objetivo no se encuentran en la misma componente conectada del grafo G_W, entonces todos los vértices de G_T son considerados ($G_W = G_T$). Si aún así las configuraciones inicial y objetivo siguen sin pertenecer a la misma componente conectada de G_W, entonces algunas nuevas muestras (no evaluadas) son agregadas a G_T, siguiendo la secuencia determinística usada en la creación inicial del grafo (normalmente con unas pocas muestras de más, el grafo G_T puede ser reconectado).

Algoritmo 3 Validate

Require: $\mathbf{P}_i^q$, $\mathbf{P}_{d'}^q$, G_T
Ensure: $valid$, G_T
  $[valPath, v_{ns}, e_{ds}] =$ SoftValidation$(\mathbf{P}_i^q, \mathbf{P}_d^q)$
  if $valPath$ **then**
    $[valPath, v_{nh}, e_{dh}] =$ HardValidation$(\mathbf{P}_i^q, \mathbf{P}_d^q, d_{val})$
    if $valPath$ **then**
      return true, G_T
    else
      $v_n = v_{ns} \cup v_{nh}$
      $e_d = e_{ds} \cup e_{dh}$
    end if
  end if
  $G_T =$ UpdateGraph(G_T, v_n, e_d)
  return false, G_T

4.4.2. Validación del camino

Otro de los factores que contribuyen a la correcta selección de muestras utilizadas para construir la solución del problema y que mejoran la efectividad de los algoritmo de planificación, es la distribución que adoptan estas muestras en el espacio de configuraciones y en particular frente a los obstáculos.

Algunos algoritmos de validación tardía conocidos, evalúan el camino candidato a ser la solución arista por arista. Este tipo de validación puede llegar a ser muy costosa si este camino resulta ser descartado en la última arista. Normalmente este proceso es realizado por el planificador local (*local planner*), que siguiendo una línea recta entre dos muestras evalúa muestras adicionales obtenidas por interpolación para determinar si toda la arista es libre de colisión con una resolución determinada por el paso (*step size*). En el enfoque propuesto aquí, la validación del camino candidato se divide en dos partes denominadas respectivamente validaciones SoftValidation y HardValidation. Adicionalmente, se ha incorporado un método de sobre–muestreo con el fin de generar muestras aleatorias en la vecindad de los obstáculos cuando alguna de estas validaciones falla. El proceso de validación se muestra en el Algoritmo 3, que además de controlar la ejecución de las dos validaciones, usa la función UpdateGraph.

El método de SoftValidation evalúa progresivamente el camino candidato, desde $\mathbf{q}_s$ hasta $\mathbf{q}_g$ y devuelve verdadero (*true*) si todas las aristas cumplen sus criterios de validación o en caso contrario devuelve falso (*false*), una lista de nuevas muestras a ser agregadas al grafo G_T y, eventualmente, una lista con aristas existentes en el mismo grafo a ser borradas (en el caso de dividir una arista, al agregar un nodo intermedio y es necesario borrar la arista original). Una vez que el algoritmo SoftValidation aprueba el camino candidato, el procedimiento HardValidation es invocado para realizar el segundo paso de la validación y obtener el camino deseado $\mathbf{P}_d^q$.

La validación SoftValidation se detalla en el Algoritmo 4 y se basa en el uso del valor de H de los vértices que definen una arista. Primero, una arista no necesita ser validada si previamente lo ha sido y el valor de H de sus vértices no ha cambiado. Esta evaluación es realizada por la función Change. Cuando se hace necesaria la validación, se debe tener en cuenta que una arista e_{ij} será válida si:

$$H_i < 1, H_j < 1 \wedge |H_i - H_j| < K \cdot \text{máx}(H_i, H_j) \qquad \text{C.1}$$

donde $0 < K < 1$. Esta condición evalúa la pendiente de esa arista de forma que variando el valor de K se permiten acercamientos más agresivos hacia los obstáculos. Si la arista e_{ij} no supera esta condición, se obtiene un vértice temporal v_t correspondiente al punto medio de e_{ij} y que valida la arista si:

$$H_t < \text{máx}(H_i, H_j) \qquad \text{C.2}$$

siendo H_t el valor de H correspondiente al vértice v_t. Esta condición asegura que al menos el punto medio de la arista se encuentra tan lejos de los obstáculos como el más cercano de sus vertices. En cualquier otro caso, la arista no es válida y se ejecuta un método de sobre–muestreo que obtiene muestras cerca del obstáculo sobre el cual cae la arista no válida con centro en v_t.

El objetivo principal del método de sobre–muestreo, realizado por el método Oversample, es el de generar muestras utilizando al máximo la información obtenida del proceso de validación. Se generan N_{samp} muestras aleatorias dentro de la hiperesfera determinada por un radio dado y centrada en uno de los vértices o en el punto medio de la arista. El método de

Algoritmo 4 SoftValidation

Require: $\mathbf{P}_i^q$, $\mathbf{P}_d^q$
Ensure: $valid$,v_n,e_d
  $v_n = \emptyset$, $e_d = \emptyset$
  for all $e_{ij} \in \mathbf{P}_i^q$ **do**
    if Change(e_{ij}, $\mathbf{P}_d^q$) **then**
      if e_{ij} pass conditions C.1 and C.2 **then**
        Label e_{ij} as valid
      else
        $[v_n, e_d] = \text{Oversample}(N_{samp}, e_{ij}, v_n, e_d)$
      end if
    end if
  end for
  return AllValid($\mathbf{P}^q$), v_n, e_d

sobre–muestreo es llamado tanto si los vértices como el punto medio de la arista caen dentro de un obstáculo. En el primer caso, la hiperesfera se centra sobre el vértice que permanece fuera del obstáculo y en el segundo caso se centra directamente sobre el punto medio. Un caso especial de sobre–muestreo ocurre cuando el valor H del punto medio es mayor al valor H de sus vértices pero menor que 1, en el que solamente se agrega el punto medio como muestra al grafo con la finalidad de dividir la arista en dos partes.

El método HardValidation obtiene muestras a lo largo de todas las aristas y evalúa su valor de H a fin de etiquetarlas ya sea como obstáculos o como muestras libres. Este procedimiento asegura que el camino propuesto es efectivamente un camino libre. Este método usa la secuencia de van der Corput para seleccionar cual muestra es extraída y evaluada cada vez. El número de puntos evaluado por arista depende de la longitud de la misma y asegura que la distancia entre dos puntos consecutivos es menor que una distancia umbral y que corresponde al tamaño del paso (*Step size*) del planificador local.

Todas las aristas son validadas en orden consecutivo desde $\mathbf{q}_s$ hasta $\mathbf{q}_g$. Una arista es valida sólo si todos sus puntos evaluados tienen un valor de H menor que 1. En este caso todo los puntos intermedios son descartados y la arista es etiquetada como válida. El proceso es detenido inmediatamente cuando un punto no satisface la condición. Entonces la arista es etiquetada como no válida, todos los puntos son regresados para ser adicionados a G_T a la vez que se elimina la arista.

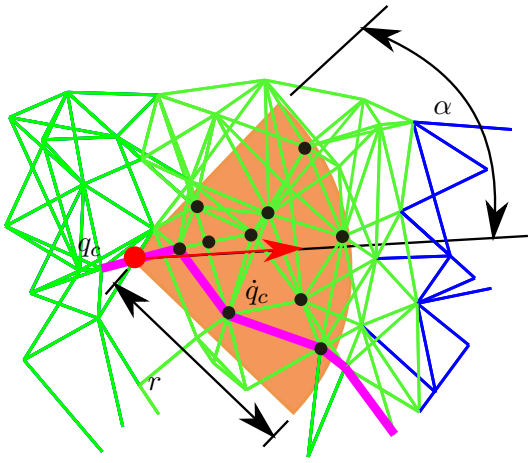


Figura 4.2: Región de actividad de $\mathbf{q}_s$ que define los vértices de G_T (en negro) que deben ser objeto de la actualizacion de su valor H.

4.4.3. Validación local al rededor de $\mathbf{q}_s$

Como se ha comentado al inicio del capítulo, el algoritmo FollowPath continuamente actualiza el camino. Además de la necesidad de recalcular el camino debido a la presencia de obstáculos móviles que lo pueden obstruir, el propio movimiento del robot puede requerir una actualización del camino solución (es de recalcar que dependiendo de la tarea, el camino puede ser seguido sólo de forma aproximada como en el caso de las tareas teleoperadas donde el camino sólo es una guía). Por esto, con el fin de facilitar el recálculo del camino cerca de la configuración actual del robot, $\mathbf{q}_c$, se introduce la evaluación del valor de H de algunos nodos en la vecindad de $\mathbf{q}_c$. Este proceso es llevado a cabo por la función UpdateRange$(G_T, \mathbf{q}_c, \dot{\mathbf{q}}_c, \mathbf{P}_d^q)$ en el algoritmo FindPath como se indica a continuación. Se define una región cónica con vértice en $\mathbf{q}_c$ y eje definido por $\dot{\mathbf{q}}_c$ que tiene un casquete esférico como base como se aprecia en la Figura 4.2. Tanto el ángulo α como la altura r son proporcionales a la magnitud de $\dot{\mathbf{q}}_c$. Así, todos los vértices de G_T que caen dentro de esta región son actualizados en su valor H. Este procedimiento agrega un comportamiento reactivo sencillo al algoritmo, frente a los cambios en la configuración del robot, de forma que se pueda ofrecer rápidamente una solución libre y segura desde cualquier nueva configuración de partida.

Algoritmo 5 FindPath

Require: $\mathbf{q}_s$, $\mathbf{q}_g$, $\mathbf{P}_d^q$, G_T
Ensure: $\mathbf{P}_d^q$, G_T
  if $\mathbf{P}_d^q \neq \emptyset$ **then**
    if $\mathbf{q}_s \neq$ init$(\mathbf{P}_d^q)$ **then**
      UpdateRange$(G_T, \mathbf{q}_c, \dot{\mathbf{q}}_c, \mathbf{P}^q)$
    end if
  else
    $G_T =$ SetGraph(Sampling$(M) \cup \mathbf{q}_s \cup \mathbf{q}_g, K_{neig})$
  end if
  $\mathbf{P}_i^q = \mathbf{P}_d^q$
  while nodes$(G_T) \leq$ MAXSAMPLES **do**
    $G_W =$ WorkGraph$(G_T, \mathbf{P}_i^q, level)$
    $\mathbf{P}_i^q =$ ShortestPath$(G_W, \mathbf{q}_s, \mathbf{q}_g)$
    if UpdateH$(\mathbf{P}_i^q, \mathbf{P}_d^q)$ **then**
      return $\mathbf{P}_i^q$, G_T
    end if
    $[validPath, G_T] =$ Validate$(\mathbf{P}_i^q, \mathbf{P}_d^q, G_T)$
    if $validPath$ **then**
      return $\mathbf{P}_i^q$, G_T
    end if
  end while
  return $\emptyset$, G_T

4.5. Algoritmo completo

El procedimiento mostrado en el Algoritmo 5 devuelve un camino $\mathbf{P}_d^q$ desde la configuración inicial $\mathbf{q}_s$ hasta la objetivo $\mathbf{q}_g$ y el grafo G_T (evaluado tardíamente) usado para este propósito. Cuando ya se cuenta con un camino validado previamente, este puede ser usado por el algoritmo para encontrar, si es necesario, un nuevo plan. Se usan las siguientes funciones:

- **UpdateRange**$(G_T, \mathbf{q}_c, \dot{\mathbf{q}}_c, \mathbf{P}^q)$: Actualiza el valor de H de todos los vértices dentro de la región circundante a la configuración actual, como se detalla en el Apartado 4.4.3.

- **SetGraph**(S): Crea un grafo a partir de un conjunto de muestras S usando la conectividad dada por K_{neig}. En este procedimiento S está compuesto por la configuración inicial $\mathbf{q}_s$, la configuración objetivo $\mathbf{q}_g$ y un conjunto M de muestras generadas por una función que utiliza la secuencia Halton (secuencia de baja discrepancia) de forma que

se pueda tener una cobertura completa del espacio de configuraciones $\mathcal{C}$.

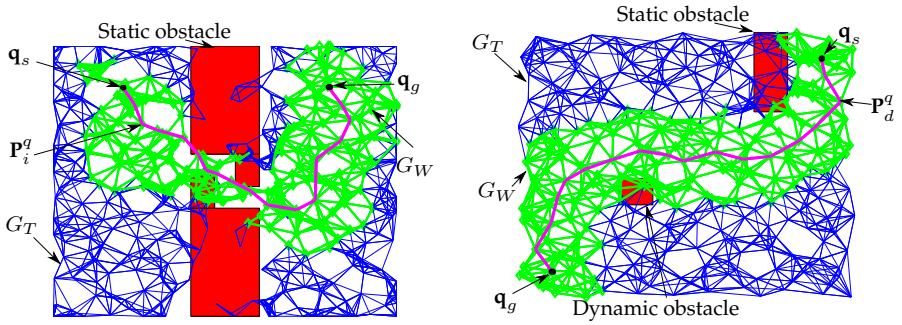
- **WorkGraph**$(G_T, \mathbf{P}^q, level)$: Devuelve el subgrafo G_W de G_T que contiene el conjunto de vértices que forman la vecindad del camino $\mathbf{P}^q$ hasta el nivel solicitado, donde $\mathbf{P}^q$ es $\mathbf{P}^q_d$ si existe, o $\mathbf{P}^q_i$ en caso contrario. Operaciones adicionales pueden ser requeridas para obtener un subgrafo valido, como se explica en el Apartado 4.4.1.

- **ShortestPath**$(G_F, \mathbf{q}_s, \mathbf{q}_g)$: Devuelve el camino de menor costo $\mathbf{P}^q_i$ dentro de G_W desde $\mathbf{q}_s$ hasta $\mathbf{q}_g$, usando el algoritmo de búsqueda A^*.

- **UpdateH**$(\mathbf{P}^q_i, \mathbf{P}^q_d)$: Calcula el valor de H para cada vértice en $\mathbf{P}^q_i$ usando la Ec. (4.1), y devuelve verdadero si $\mathbf{P}^q_i$ y $\mathbf{P}^q_d$ son iguales, y falso en cualquier otro caso.

- **Validate**$(\mathbf{P}^q_i, \mathbf{P}^q_d, G_T)$: Evalúa las aristas del camino $\mathbf{P}^q_i$ de forma tardía, basado en el valor de H. Una arista es considerada potencialmente libre si el valor de H de sus extremos y el de su punto medio is menor que 1. Un camino con todas sus aristas clasificadas como potencialmente libres es efectivamente evaluado por medio del cálculo de la distancia a los obstáculos cuando el robot está en un conjunto de configuraciones intermedias de la arista, calculadas siguiendo la secuencia de **van der Corput**. Como resultado de este proceso de evaluación, algunos vértices pueden ser añadidos al grafo, y algunos otros vértices y aristas pueden ser removidos. Este proceso de validación se ha detallado en el Apartado 4.4.2.

En la Figura 4.3a se puede apreciar la evaluación (no satisfactoria) del camino candidato $\mathbf{P}^q_i$ durante el proceso inicial de búsqueda del camino deseado $\mathbf{P}^q_d$. En la Figura 4.3b se observa una iteración de validación y replanificación del camino inicialmente validado $\mathbf{P}^q_d$ teniendo en cuenta el obstáculo dinámico que se mueve hacia arriba con velocidad $\dot{O}_i$. Más adelante en el Capítulo 7 se desarrollan los ejemplos que permiten evaluar la efectividad del planificador propuesto en este trabajo.



(a) Búsqueda inicial de $\mathbf{P}_d^q$. (b) Replanificación de $\mathbf{P}_d^q$.

Figura 4.3: Componentes conceptuales de la planificación.

4.6. Conclusiones

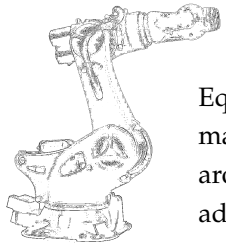
En este capítulo se ha presentado el sistema de planificación utilizado como herramienta para proveer caminos libres de colisión de todo el brazo manipulador con el que se lleva a cabo la teletarea. Este camino es libre de colisiones tanto con los obstáculos estáticos del entorno del robot como de colisiones con otro robots teleoperados que pueden estar presentes en el nodo–operación y que pueden compartir todo o parte de su espacio de trabajo. Estos otros robots son considerados como obstáculos móviles de los cuales se puede saber su posición y orientación en cada instante que se actualiza la información de la evolución de la tarea.

El proceso de validación de las aristas del camino, normalmente desarrollado por el planificador local, aquí se ha dividido en dos partes: una validación suave y otra fuerte. Esta división permite descartar más rápidamente un camino candidato que no resulta viable y toma aproximandamente el mismo esfuerzo para validar un camino candidato factible, que un planificador local lineal estándar.

Este algoritmo de planificación se ejecuta en paralelo durante la ejecución de la teletarea de forma que reacciona frente a los cambios del entorno y a las decisiones tomadas por parte del teleoperador para ofrecer en todo momento un camino viable.

5

Ayudas a la teleoperación



Equipped with his five senses, man explores the universe around him and calls the adventure Science.

Edwin Hubble

En este capítulo se abordan las ayudas que se brindan al teleoperador con el ánimo de dar soporte al cumplimiento de la tarea y que pueden ser generados a partir de un camino libre de colisión previamente obtenido con un planificador de movimientos. La generación de fuerzas de guiado se hace para los momentos de teleoperación activa o sea, cuando se están comandando movimientos del robot remoto, y se extiende de la misma forma para ayudar al operador a encontrar el mejor emplazamiento de espacio de trabajo avatar del dispositivo háptico cuando se lleva a cabo una resincronización (ver Sección 3.3).

De la misma forma se describe un mecanismo de ayuda para el cambio de configuraciónes cinemáticas del robot, para los casos en los que la solución de la tarea incluye uno o varios de ellos. La generación de este tipo particular de sensación háptica a través de fuerza se enmarca dentro de una aplicación de *rendering* háptico (Salisbury et al., 2004).

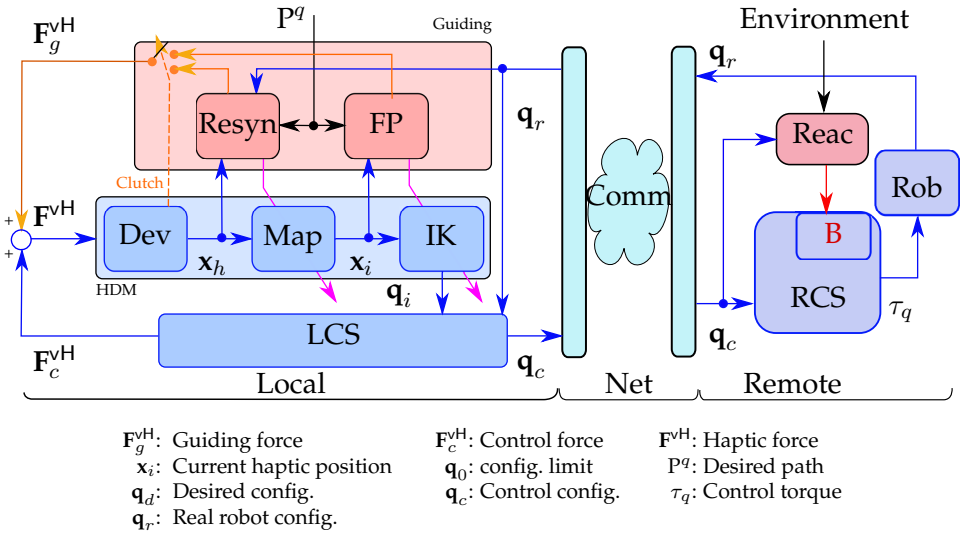


Figura 5.1: Sistema de teleoperación ampliado con las ayudas.

5.1. Enfoque global

El marco de teleoperación bilateral utilizado está compuesto por el nodo local, la celda remota y el canal de comunicaciones tal como se ha mostrado en la Figura 3.2. Los bloques básicos, de color azul, comprenden en el nodo local el dispositivo háptico (HDM) y el sistema local de control o LCS (*Local Control System*) y en el nodo remoto el robot (Rob) y el el sistema de control remoto o RCS (*Remote Control System*). El esquema de control bilateral y la correspondencia de los espacios de trabajo entre el robot y el dispositivo háptico se han descrito anteriormente en las Secciones 3.1 y 3.2 respectivamente.

Este esquema básico se ha extendido con las ayudas aquí propuestas como asistencia a la teleoperación que se pueden apreciar en la Figura 5.1, que corresponden a los bloques de color rojo y comprenden el bloque de guiado en el nodo local y bloque de comportamiento reactivo en el nodo remoto. El procedimiento de asistencia en la teleoperación está esbozado en el Apartado 5.2 junto con las ayudas utilizadas.

El bloque de guiado permite la generación de fuerzas en el dispositivo háptico ($\mathbf{F}_g^{vH}$) que sugieren el seguimiento de un camino libre de colisiones para el robot teleoperado y que unido con el bloque de LCS, permite realizar de forma suave y robusta los cambios de configuración cuando sean necesarios. En el momento que el robot se desconecta del dispositivo háp-

tico, este bloque se utiliza para generar fuerzas de asistencia durante la resincronización. El bloque reactivo ubicado en la parte remota del sistema, se usa para incrementar la ganancia de amortiguamiento B_r del lazo de control remoto (RCS) cuando se detectan posibles colisiones del robot con su entorno. Esta característica es muy importante en el caso de entornos multirobot para evitar colisiones en el espacio de trabajo que comparten.

La asistencia a la teleoperación propuesta, esta basada en la hipótesis que la tarea a ser teleoperada es conocida, esto es que la configuración inicial y la configuración objetivo, respectivamente $\mathbf{q}_s$ y $\mathbf{q}_g$, así como que el modelo del robot y de su entorno de trabajo son conocidos. Asumir esta hipotesis es común y razonable en muchas tareas teleoperadas(manipulación de objetos en plantas nucleares, mantenimiento en estaciones espaciales, algunos procedimientos quirúrgicos, etc.). Es por esta razón que la asistencia a la teleoperación esta diseñada con base en la disponibilidad de un camino libre de colisión calculado por un planificador en el espacio de configuraciones del robot ($\mathcal{C}$), en este caso un *PRM*, que conecta $\mathbf{q}_s$ con $\mathbf{q}_g$ (ver Capítulo 4) con las siguientes características:

- Sea $\mathbf{P}^q = \{\mathbf{q}_s, \mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_g\} \in \mathcal{C}$, el camino solución donde las $\mathbf{q}_j$ son las configuraciones que han sido probadas para ser libres de colisión y que permiten validar las aristas y la conexión de todo el camino. Esto quiere decir que las configuraciones $\mathbf{q}_j$ y la $\mathbf{q}_{j+1}$ son configuraciones muy cercanas entre si, separadas por una distancia definida por el planificador local del *PRM*. Este planificador local del *PRM* es de tipo lineal y por esta razón el camino resultante $\mathbf{P}^q$ es lineal a tramos. Cuando dos configuraciones consecutivas en $\mathbf{P}^q$ corresponden a configuraciones cinemáticas diferentes del robot, se obtiene una configuración intermedia por interpolación lineal cuando la variable articular que cambia de signo se torna cero. La configuración interpolada se agrega a $\mathbf{P}^q$.

- Sea $\mathbf{P}^x = \{\mathbf{x}_s, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_g\} \in$ *SE(3)*, el camino seguido por el TCP del robot en el espacio físico, correspondiente a $\mathbf{P}^q$ en el espacio de configuraciones, calculado por la cinemática directa (FK por su sigla en inglés *Forward Kinematics*) del robot, esto es que cada $\mathbf{x}_j$ es la posición y orientación del sistema coordenado TCP cuando el robot se encuentra en $\mathbf{q}_j$. Dado que las configuraciones del camino $\mathbf{P}^q$ se en-

Algoritmo 6 Guided teleoperation procedure.

Require:
$\mathbf{q}_s, \mathbf{q}_g$: configuraciones inicial y objetivo.
δ: Paso del planificador local del *PRM*.

$\mathbf{P}^q = \text{PRM\_Solve}(\mathbf{q}_s, \mathbf{q}_g, \delta)$
$\mathbf{P}^x = \text{FK}(\mathbf{P}^q)$
while teleoperation **do**
  $\mathbf{x}_i = \text{MapHaptic}()$
  if clutch **then**
    $\mathbf{F}^{\mathsf{vH}} = \text{Resynchronization}(\mathbf{x}_i)$
  else
    $\mathbf{q}_r = \text{SubscribeRobotConfig}()$
    $\mathbf{F}_g^{\mathsf{W}} = \text{ComputeForce}(\mathbf{x}_i)$
    $\mathbf{F}_g^{\mathsf{vH}} = \text{ChangeFrame}(\mathbf{F}_g^{\mathsf{W}})$
    $\mathbf{q}_i = \text{IK\_filtered}(\mathbf{x}_i)$
    $[\mathbf{F}_c^{\mathsf{vH}}, \mathbf{q}_c] = \text{Control}(\mathbf{q}_i, \mathbf{q}_r)$
    $\text{PublishRobotCommand}(\mathbf{q}_c)$
    $\mathbf{F}^{\mathsf{vH}} = \mathbf{F}_g^{\mathsf{vH}} + \mathbf{F}_c^{\mathsf{vH}}$
  end if
  Exert $\mathbf{F}^{\mathsf{vH}}$ in the haptic device
end while

cuentran muy cerca unas de otras, el camino $\mathbf{P}^x$ también puede ser considerado lineal a tramos.

El Algoritmo 6 esboza el ciclo del procedimiento propuesto de asistencia a la teleoperación donde primero se calcula el camino libre de colisión $\mathbf{P}^q$ en el espacio de configuraciones $\mathcal{C}$ y su correspondiente camino $\mathbf{P}^x$ en *SE(3)* permitiendo iniciar el ciclo. El algoritmo usa las siguientes funciones:

- **PRM_Solve**: Usa el planificador basado en *PRM* del Capítulo 4 para calcular el camino libre de colisión $\mathbf{P}^q \in \mathcal{C}$.
- **FK**: Obtiene el camino $\mathbf{P}^x \in$ *SE(3)* a través de la cinemática directa FK.
- **MapHaptic**: Usa la Ec. (3.2) para calcular la posición y orientación del sistema coordenado TCPdel robot, como se ha detallado en la Sección 3.2.

- **Resychronization**: Usa la Ec. (3.4) para definir una nueva correspondencia entre los espacios de trabajo del robot y del dispositivo háptico. Durante este proceso se generan fuerzas de guiado que sugieren al operador humano el mejor lugar para llevar a cabo esta resincronización, como se detalla en la Sección 5.3.
- **SubscribeRobotConfig**: Recibe la configuración actual del robot $\mathbf{q}_r$ proveniente de la celda remota.
- **ComputeForce**($\mathbf{F}_g^{\mathsf{W}}$): Calcula las fuerzas de guiado que permiten seguir el camino libre de colisión, como se detalla más adelante en la Sección 5.2.
- **ChangeFrame**($\mathbf{F}_g^{\mathsf{W}}$): Transforma la fuerza $\mathbf{F}_g^{\mathsf{W}}$ del sistema coordenado del robot al sistema coordenado vH del dispositivo háptico.
- **IK_filtered**($\mathbf{x}_i$): Calcula la configuración del robot $\mathbf{q}_i$ a partir de $\mathbf{x}_i$ utilizando la cinemática inversa o IK (por su sigla en inglés *Inverse Kinematics*) y el método de cambio robusto de configuración, detallado en el Apartado 5.2.2.
- **Control**($\mathbf{x}_i, \mathbf{q}_r$): Ejecuta el algoritmo de control bilateral P+d.
- **PublishRobotCommand**($\mathbf{q}_c$): Envía a la celda remota la configuración comandada para el robot.

Este ciclo de teleoperación puede volver a comenzar cuando el usuario lo desee, ya que en cualquier momento se puede pedir al sistema un nuevo camino desde la configuración donde se encuentre el robot de forma que pueda ser utilizado para brindar la asistencia en la teleoperación desde allí.

5.2. Soporte en la teleoperación

5.2.1. Fuerzas de guiado

La ayuda de guiado ha sido concebida para cumplir con los siguientes requerimientos:

- No ejercer realimentación de fuerza y de torque al operador humano simultáneamente, dado que es muy difícil por parte del usuario identificar correctamente la sugerencia que está sintiendo en ese momento

y casi siempre se responde a un torque con una translación en lugar de con una rotación (ver Sección 7.1). El usuario debe tener la posibilidad de seleccionar cuál tipo de realimentación desea recibir, entre fuerzas y torques, y así estar preparado para responder acorde al estímulo.

- Las fuerzas/torques realimentadas deben guiar al usuario hacia el camino elegido, desapareciendo en una zona muerta cercana a él, es decir que dadas las sugerencias sentidas, el dispositivo háptico debe ser posicionado y orientado correctamente en el camino. Opcionalmente, cuando el usuario se encuentra dentro o muy cerca de la zona muerta, debe tener la posibilidad de recibir o no una fuerza de empuje que le indica la dirección del camino, es decir, la dirección hacia donde se encuentra $\mathbf{q}_g$.

Antes de seguir con los métodos de generación de las fuerzas/torques de guiado, es necesario definir la siguiente nomenclatura:

- $\mathbf{x} = \left(\begin{array}{c|c} \mathcal{R} & \mathbf{p} \\ \hline 0 & 1 \end{array}\right)$ describe la configuración (posición y orientación) del TCP del robot, siendo $\mathbf{x}_i$ la actual y $\mathbf{x}_d \in \mathbf{P}^x$ la configuración del camino más cercana a $\mathbf{x}_i$.

- $\mathbf{d}_t$ es el vector $\mathbf{d}_t = \mathbf{p}_d - \mathbf{p}_i$ y $(\mathbf{d}_r, \theta_r)$ la representación en eje-ángulo de $\mathcal{R}_i^{-1}\mathcal{R}_d$.

- d_t es la distancia traslacional entre $\mathbf{x}_i$ y $\mathbf{x}_d$, es decir $d_t = |\mathbf{d}_t|$, y d_r la distancia rotacional definida como $d_r = \theta_r$.

- $\mathbf{x}_k$ y $\mathbf{x}_{k+1}$ son los nodos de $\mathbf{P}^x$ dentro del los cuales se encuentra $\mathbf{x}_d$, y $\mathbf{s}$ es el vector $\mathbf{s} = \mathbf{p}_{k+1} - \mathbf{p}_k$.

- ϵ_{t_n} y ϵ_{r_n} son, respectivamente, los umbrales de distancia traslacional y rotacional, para $n = \{1, 2, 3\}$.

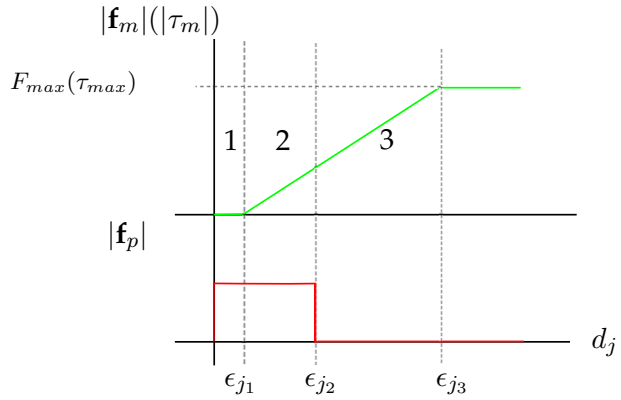


Figura 5.2: Zonas definidas para la generación de las fuerzas de guiado: 1) Zona muerta, 2) Zona de empuje, 3) Zona de sólo atracción. j representa a t o a r.

Así, la función **ComputeForce**($\mathbf{x}_i$) usada en el Algoritmo 6 calcula las fuerzas generalizadas de guiado $\mathbf{F}_g^{\mathsf{W}} = (\mathbf{f}_g^{\mathsf{W}}, \tau_g^{\mathsf{W}})^T$ a partir de las dos componentes $\mathbf{F}_m$ y $\mathbf{F}_p$ de forma que:

- $\mathbf{F}_m = (\mathbf{f}_m, \tau_m)^T$ es una fuerza de atracción sobre el TCP del robot hacia el camino. La fuerza y el torque son realimentados de forma separada siempre que las distancias d_t y d_r sean superiores a los umbrales definidos.

- $\mathbf{F}_p = (\mathbf{f}_p, 0)^T$ es una fuerza de empuje a lo largo de $\mathbf{s}$ en dirección al siguiente nodo del camino. Esta fuerza generalizada se genera a solicitud del usuario siempre que d_t sea superior al umbral definido.

Las fuerza $\mathbf{f}_g^{\mathsf{W}}$ y el torque τ_g^{W} están determinados por (Figura 5.2):

$$\begin{aligned} \mathbf{f}_g^{\mathsf{W}} &= \begin{cases} \mathbf{f}_p \cdot \mathrm{A} & d_t < \epsilon_{t_1} \\ \mathbf{f}_m + \mathbf{f}_p \cdot \mathrm{A} & \epsilon_{t_1} < d_t < \epsilon_{t_2} \\ \mathbf{f}_m & d_t > \epsilon_{t_2} \end{cases} \\ \tau_g^{\mathsf{W}} &= \begin{cases} 0 & d_r < \epsilon_{r_1} \\ \tau_m & d_r > \epsilon_{r_1} \end{cases} \end{aligned} \tag{5.1}$$

donde:

$$\begin{aligned}
\mathbf{f}_m &= \frac{\mathbf{d}_t}{|\mathbf{d}_t|}\,\text{mín}\,(F_{max}, F_{max}\frac{d_t-\epsilon_{t_1}}{\epsilon_{t_3}-\epsilon_{t_1}}) \\
\tau_m &= \frac{\mathbf{d}_r}{|\mathbf{d}_r|}\,\text{mín}\,(\tau_{max}, \tau_{max}\frac{d_r-\epsilon_{r_1}}{\epsilon_{r_3}-\epsilon_{r_1}}) \\
\mathbf{f}_p &= \text{Fuerza constante a lo largo de } \mathbf{s} \\
\text{A} &= 1/0 \text{ (Activa/Inactiva la fuerza)} \\
F_{max} &= \text{Máxima fuerza que el dispositivo puede ejercer.} \\
\tau_{max} &= \text{Máximo torque que el dispositivo puede ejercer.}
\end{aligned}$$

Con el fin de aplicar la fuerza de guiado al teleoperador a través del dispositivo háptico, la fuerza generalizada $\mathbf{F}_g^{\mathsf{W}}$ es trasladada al sistema coordenado vH, aplicando el principio de trabajo virtual. Esto se lleva a cabo en la función **ChangeFrame**($\mathbf{F}_g^{\mathsf{W}}$) del Algoritmo 6:

$$\mathbf{F}_g^{\mathsf{vH}} = \begin{bmatrix}\mathbf{f}_g^{\mathsf{vH}} \\ \tau_g^{\mathsf{vH}}\end{bmatrix} = \begin{bmatrix}\mathcal{R}_{\mathsf{vH}}^{\mathsf{W}} & \mathbf{0} \\ \mathbf{0} & \mathcal{R}_{\mathsf{vH}}^{\mathsf{W}}\end{bmatrix} \cdot \begin{bmatrix}\mathbf{f}_g^{\mathsf{W}} \\ \tau_g^{\mathsf{W}}\end{bmatrix} \tag{5.2}$$

5.2.2. Ayuda para cambio de configuración cinemática

La función de **Control** en el Algoritmo 6 realiza el control bilateral siguiendo el esquema P+d, como se ha explicado en la Sección 3.1. Esta función ha sido extendida con un sistema de ayuda para cruzar de forma suave y robusta, por los puntos del camino donde cambia la configuración cinemática del robot. Esto se realiza modificando la configuración comandada con una saturación a forma de efecto embudo que fuerza el cambio de configuración en la forma deseada, en este caso, siguiendo el camino solución $\mathbf{P}^q$. La Figura 5.3 ilustra la siguiente nomenclatura y definiciones:

$\mathbf{q}_0$: Es una configuración donde cambia la configuración cinemática.

$\mathbf{q}_p$,$\mathbf{q}_n$: Son respectivamente la configuración anterior y siguiente en el camino, con respecto a $\mathbf{q}_0$.

$\mathbf{u}_p, \mathbf{u}_n$: Son los vectores unitarios con origen en $\mathbf{q}_0$ y que apuntan en dirección a $\mathbf{q}_p$ y a $\mathbf{q}_n$, respectivamente.

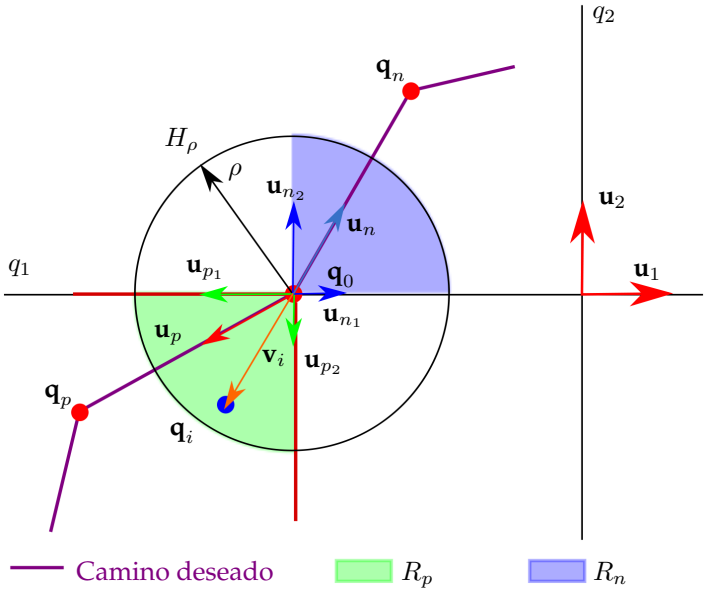


Figura 5.3: Zonas definidas para el correcto cambio de la configuración cinemática en el ejemplo de 2 *gdl* del robot RR mostrado.

$\mathbf{u}_j$: Es el vector unitario de la j-ésima coordenada en el espacio de configuraciones, que corresponde a la j-ésima articulación. En este ejemplo $j = \{1; 2\}$.

$\mathbf{q}_i$: Es la configuración del robot correspondiente a la posición actual del usuario $\mathbf{x}_i$ (se calcula a partir de $\mathbf{x}_i$ utilizando la cinemática inversa y eligiendo la misma solución cinemática que $\mathbf{q}_d \in \mathbf{P}^q$, que es la configuración correspondiente $\mathbf{x}_d$, el punto sobre el camino más cercano a $\mathbf{x}_i$).

$\mathbf{v}_i$: Es el vector desde $\mathbf{q}_0$ a $\mathbf{q}_i$.

H_ρ: Es la hiperesfera de radio ρ centrada en $\mathbf{q}_0$, que define la región donde el efecto embudo es sentido.

$\mathbf{u}_{p_j}, \mathbf{u}_{n_j}$: Son los vectores unitarios definidos como:

$$\begin{aligned} \mathbf{u}_{p_j} &= \text{sign}(\mathbf{u}_p \cdot \mathbf{u}_j)\mathbf{u}_j \quad \forall j \in 1, \ldots, d \\ \mathbf{u}_{n_j} &= \text{sign}(\mathbf{u}_n \cdot \mathbf{u}_j)\mathbf{u}_j \quad \forall j \in 1, \ldots, d \end{aligned} \tag{5.3}$$

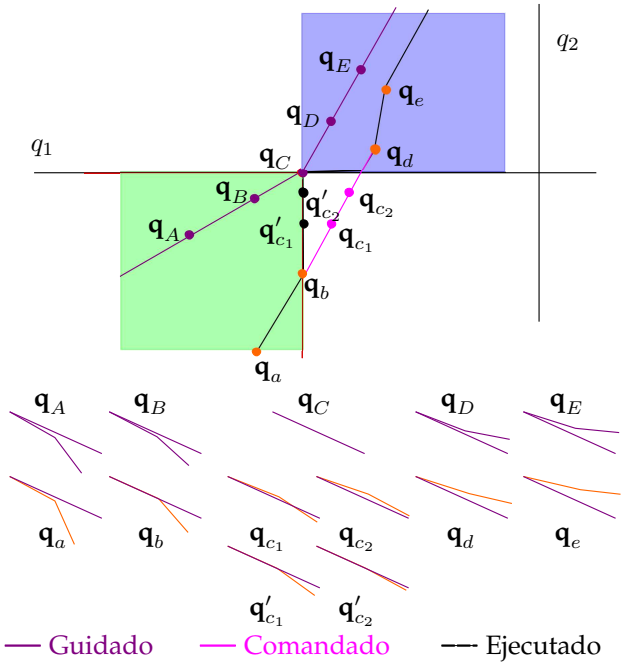


Figura 5.4: Caminos: guiado, comandado y ejecutado tanto en el espacio de configuraciones (superior) como en el espacio físico (inferior).

R_P, R_N**:** Son las regiones dentro de H_ρ que son definidas por la combinación lineal positiva de los vectores $\mathbf{u}_{p_j} \forall j \in 1, \ldots, d$ y $\mathbf{u}_{n_j} \forall j \in 1, \ldots, d$, respectivamente.

La ayuda se activa automáticamente cuando la configuración $\mathbf{q}_i$ cae dentro de la región H_ρ, modificando sus valores para hacer que se mantenga dentro de las zonas $R_P \cup R_N$, es decir que las coordenadas de $\mathbf{q}_i$ son modificadas de acuerdo a:

$$q_{i_j} = \begin{cases} q_{0_j} + v_{i_j} & \text{if } \mathbf{v}_i \cdot \mathbf{u}_{p_j} > 0 \text{ or } \mathbf{v}_i \cdot \mathbf{u}_{n_j} > 0 \\ q_{0_j} & \text{en otro caso.} \end{cases} \tag{5.4}$$

La Figura 5.4 muestra un ejemplo donde el camino a guiar cruza por un punto donde la configuración cinemática cambia, dado por el cambio de signo en q_2. El uso de esta ayuda propuesta se realiza de la siguiente forma: Sea $\mathbf{P}^q = \{\mathbf{q}_A, \mathbf{q}_B, \mathbf{q}_C, \mathbf{q}_D, \mathbf{q}_E\} \in \mathcal{C}$ el camino a guiar y $\mathbf{P}^x$ el camino correspondiente del sistema coordenado TCP del robot en *SE(3)*. Sea también $\{\mathbf{x}_a, \mathbf{x}_b, \mathbf{x}_{c_1}, \mathbf{x}_{c_2}, \mathbf{x}_d, \mathbf{x}_e\} \in$ *SE(3)* el camino comandado por el usuario con el dispositivo háptico y $\{\mathbf{q}_a, \mathbf{q}_b, \mathbf{q}_{c_1}, \mathbf{q}_{c_2}, \mathbf{q}_d, \mathbf{q}_e\}$ el camino en $\mathcal{C}$ calcula-

do usando la misma solución de cinemática inversa que la correspondiente al punto más cercano en el camino $\mathbf{P}^x$. Las configuraciones $\mathbf{q}_a$, $\mathbf{q}_b$, $\mathbf{q}_d$ y $\mathbf{q}_e$ forman parte de alguna de las dos zona ya sea R_P o R_N y por lo tanto se mantienen sin cambios. Sin embargo, $\mathbf{q}_{c_1}$ y $\mathbf{q}_{c_2}$ primero son transformadas, usando la Ec. (5.4), a $\mathbf{q}'_{c_1}$ y $\mathbf{q}'_{c_2}$, es decir que el camino comandado al robot es $\{\mathbf{q}_a, \mathbf{q}_b, \mathbf{q}'_{c_1}, \mathbf{q}'_{c_2}, \mathbf{q}_d, \mathbf{q}_e\}$. De esta forma, el robot cruza el punto crítico tal y como especifica el camino a guiar.

5.2.3. Sistema reactivo remoto

El guiado de asistencia se completa con un modulo llamado **Reactive** en el nodo remoto, como se muestra en la Figura 5.1. Este módulo se ha diseñado para regular el factor de amortiguamiento remoto B_r del sistema de control, como función de las posibles colisiones con objetos presentes en el entorno de trabajo. De acuerdo con la Ec. (5.5), y asumiendo que los valores del amortiguamiento local B_l y de las ganancias tanto local K_l como remota K_r tiene valores fijos, el esquema de control propuesto seguirá siendo estable si el factor de amortiguamiento remoto B_r se mantiene por encima del límite inferior, B_r^{min}, dado por:

$$B_r^{min} = \frac{({}^*T_l + {}^*T_r)^2 K_l K_r}{4B_l} \tag{5.5}$$

Un incremento de B_r por encima de B_r^{min} volverá más lenta la teleoperación, es decir que el teleoperador humano siente una dificultad creciente en los movimientos del robot, y mantendrá estable el sistema.

El módulo reactivo **Reactive** tiene información de los obstáculos presentes en el ambiente de trabajo de los robots (tanto de los obstáculos estáticos como de los dinámicos), y es por esto que puede calcular la distancia d entre el robot y ellos, y hacer que el valor de B_r decrezca con d:

$$B_r = \begin{cases} B_r^{max} & d < d_{th} \\ B_r^{max} + \frac{B_r^{min} - B_r^{max}}{d_{cov} - d_{th}}(d - d_{th}) & d_{th} < d < d_{cov} \\ B_r^{min} & d > d_{cov} \end{cases} \tag{5.6}$$

Donde d_{th} es una distancia fija mínima permitida al rededor de los obstáculos, $B_r^{max} > B_r^{min}$ es el mayor valor de amortiguamiento permitido para ralentizar al máximo los movimientos del robot cuando la distancia a

los objetos es menor que el umbral definido por d_{th}, y d_{cov} es la distancia a partir de la cual el efecto de los obstáculos desaparece.

Este enfoque no genera fuerzas repulsivas ni interfiere con el guiado mientras se generan compartamientos reactivos frente a las posibles colisiones del robot teleoperado con su entorno, aunque al aumentar el amortiguamiento remoto se afecta la transparencia del sistema de teleoperación. Es importante resaltar que al realimentar fuerzas provenientes de diversas fuentes (control, guiado, reflexión del entorno remoto, etc.) al operador a través de un dispositivo háptico, es imposible diferenciar la relevancia que tiene cada componente en la fuerza final aplicada, es por esta razón que se la presencia de este módulo reactivo sirve de diferenciación al usuario sobre lo que está sucediendo durante la teleoperación en la celda remota.

5.3. Soporte a la resincronización

Teniendo en cuenta el sistema de teleoperación propuesto y el esquema de correspondencia entre los espacios de trabajo del dispositivo háptico y el del robot, donde por un lado se pueden generar fuerzas que orientan los movimientos al teleoperador y que le permiten realizar tareas de forma más rápida y segura evitando las posibles colisiones con su entorno, y por otro lado se puede utilizar cualquier cámara de las provistas en el nodo–operación para realizar la correspondecia, se propone ahora extender la utilidad del guiado a estos períodos de resincronización. Es así como se busca aplicar pequeñas fuerzas que guíen al operador a encontrar el mejor emplazamiento para el espacio de trabajo del dispositivo háptico proyectado dentro de la escena o avatar, basándose en la misma capacidad del sistema de obtener un camino libre de colisión que soluciona la tarea. Esta solución es posible dado el conocimiento del entorno de trabajo del robot y a la capacidad de capturar sus cambios mediante sensores durante la ejecución de forma que se puede ofrecer un camino solución valido cuando es necesario.

Cuando todo el camino que va a ser teleoperado no está contenido dentro de este espacio de trabajo proyectado (dado por la diferencia del tamaño entre el espacio de trabajo del dispositivo háptico y el del robot y las escalas de traslación $\mathcal{S}_t$ y rotación $\mathcal{S}_r$ utilizadas) se hace necesaria una secuencia de resincronizaciones (Apartado 3.2.2). Como ejemplo la Figura 5.5 ilustra una

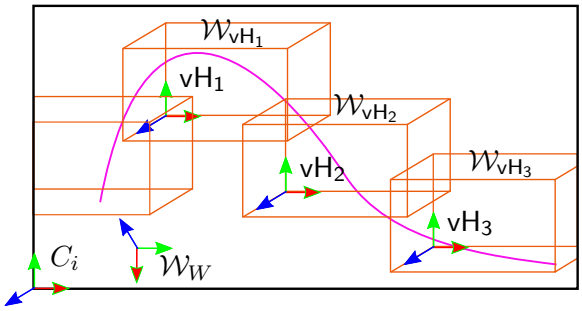


Figura 5.5: Varios espacios de trabajo avatar consecutivamente ubicados para cubrir completamente el camino que va a ser teleoperado.

de las posibles soluciones para encontrar ubicaciones sucesivas del espacio de trabajo avatar que permiten cubrir todo el camino solución.

Este proceso de resincronización se puede ver como un problema de optimización, ya sea global o local. Desde el punto de vista de problema de optimización global el problema puede ser planteado como la búsqueda del menor número de emplazamientos sucesivos del espacio de trabajo avatar necesarios para cubrir por completo el camino a teleoperar con el robot. La posición y orientación de cada una de estas ubicaciones dependen del espacio de trabajo del dispositivo háptico, las escalas utilizadas para la correspondencia (Apartado 3.2.1), el camino solución propuesto para el robot y de la posición y orientación de la cámara usada para ejecutar la tarea. El uso de la optimización global, sin embargo, no es muy útil en teleoperación dado que estos resultados tendrían que ser totalmente recalculados cuando el operador decide no seguir exactamente los movimientos sugeridos, si decide cambiar de cámara o el camino.

En el ámbito local, por otro lado, el problema se reduce a encontrar la siguiente ubicación de $\mathcal{W}_{vH}$ que cubra la mayor cantidad posible del camino solución, minimizando de forma local la necesidad de una nueva resincronización. Este enfoque asume que la resincronización se hace bajo petición del teleoperador, es decir que aunque el camino pueda aún ser seguido desde la posición actual de $\mathcal{W}_{vH}$, el operador puede solicitar la realización de una sincronización porque no se encuentra cómodo con la postura de su mano/brazo. En el momento que la resincronización es requerida, el usuario tiene la posibilidad de cambiar también la cámara activa o de seleccionar un nuevo factor de escala tanto traslacional como rotacional.

En el resto de este capítulo se presenta el enfoque utilizado para resol-

ver este problema de optimización local, que no sólo calcula la mejor nueva ubicación de $\mathcal{W}_{\mathsf{vH}}$ (Apartado 5.3.1) sino que calcula las fuerzas para guiar al usuario hasta la nueva ubicación de HIP del dispositivo háptico (Apartado 5.3.2). El enfoque actual está restringido a la parte traslacional, dado que la mayoría de dispositivos hápticos no están provistos de realimentación de torques y, como se ha probado experimentalmente, los movimientos de rotación son muy difíciles de sugerir a partir de una realimentación de torque (exceptuando la rotación sobre el eje del propio apuntador o *stylus*).

El camino que se va a teleoperar para el TCP del robot se considera un camino lineal a tramos en *SE(3)*, representado como una secuencia ordenada de sistemas coordenados de referencia (ver Sección 5.2). Los puntos del camino $\mathbf{P}^x$ (ver Sección 5.2) están definidos como los orígenes de esos sistemas coordenados y se asume que satisfacen que la distancia entre dos puntos consecutivos es pequeña comparado con el tamaño traslacional del espacio de trabajo avatar.

5.3.1. Optimización

Teniendo en cuenta la misma nomenclatura que se ha descrito en las Sección 3.2 donde se habla de la correspondencia entre los espacios de trabajo del dispositivo háptico y el del robot y principalmente:

- $\mathcal{W}_{\mathsf{vH}}$: Espacio de trabajo virtual del dispositivo háptico dentro de la escena, es decir que se proyecta $\mathcal{W}_H$ dentro de $\mathcal{W}_W$. Es también llamado **Avatar** del espacio de trabajo.

  - vH: Sistema coordenado de referencia $\mathcal{W}_{\mathsf{vH}}$.
  - vHIP: Sistema coordenado de referencia asociado a la proyección del efector final del dispositivo háptico en la escena.
  - $T_{\mathsf{vH}}^{\mathsf{vHIP}}$: Transformación entre vH y vHIP. Se calcula a partir de $T_{\mathsf{H}}^{\mathsf{HIP}}$ y usando los factores de escala.
  - $T_{\mathsf{W}}^{\mathsf{vH}}$: Transformación entre W y vH. Se define durante el proceso de sincronización.
  - $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$: Transformación entre vHIP y TCP. Se define durante el proceso de sincronización y es una rotación pura.

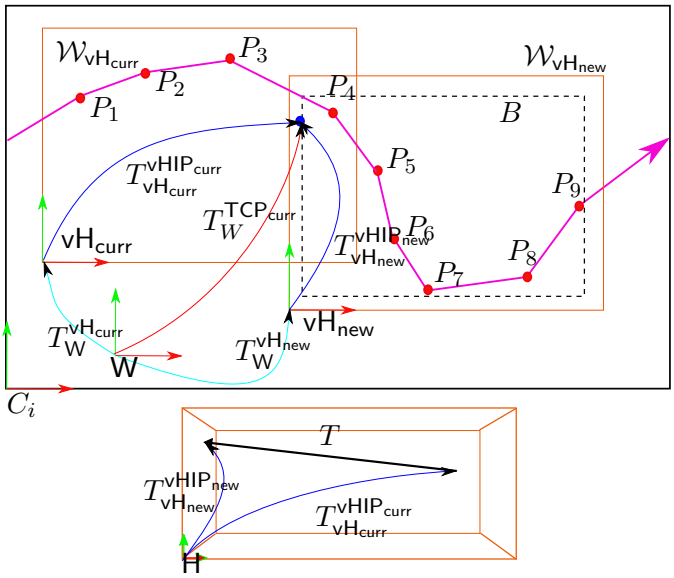


Figura 5.6: Obtención del HIP_{new} usando el Algoritmo 7.

El proceso de optimización primero calcula la mayor caja englobante B alineada a la cámara activa, que satisfaga las siguientes dos condiciones:

1. Contienen la posición actual del vHIP y la mayor cantidad posible de puntos siguientes del camino.
2. Se ajusta dentro del espacio de trabajo avatar.

Entonces ubica $\mathcal{W}_{\mathsf{vH}}$ tal que B es centrado en él y calcula la correspondiente nueva posición de vHIP. El Algoritmo 7 formaliza el procedimiento ilustrado en la Figura 5.6, y usa las siguientes funciones y nomenclatura:

- $\mathbf{P}^x$: Conjunto ordenado de puntos del camino del robot.
- $\mathbf{x}_i$: Punto número i del conjunto $\mathbf{P}^x$.
- P: Conjunto de puntos.
- **Box**(P): Función que calcula la caja englobante de los puntos del conjunto P.
- **Dist2Nearest**$(\mathbf{P}^x, \mathbf{x}_i)$: Función que devuelve el índice del punto en $\mathbf{P}^x$ que es el más cercano a $\mathbf{x}_i$.

Algoritmo 7 Resynchronization Goal.

Require: $T_W^{\mathsf{TCP}_{\mathsf{curr}}}$, transformación del actual TCP.

Ensure: $T_{\mathsf{vH}_{\mathsf{new}}}^{\mathsf{vHIP}_{\mathsf{new}}}$, nueva transformación sugerida para vHIP.

i = Dist2Nearest($\mathbf{P}^x$, pos($T_W^{\mathsf{TCP}_{\mathsf{curr}}}$))
P = {pos($T_W^{\mathsf{TCP}_{\mathsf{curr}}}$)}
B = Box($\emptyset$)
repeat
  $B_{prev} = B$
  $P = P \cup$ pos($\mathbf{x}_i$)
  B = Box(P)
  $i = i + 1$
until Fit(B,$\mathcal{W}_{\mathsf{vH}}$) = FALSE
$T_{\mathsf{W}}^{\mathsf{vH}_{\mathsf{new}}}$ = Find-Workspace(B_{prev})
return $T_{\mathsf{vH}_{\mathsf{new}}}^{\mathsf{vHIP}_{\mathsf{new}}} = \left[T_{\mathsf{W}}^{\mathsf{vH}_{\mathsf{new}}}\right]^{-1} T_W^{\mathsf{TCP}_{\mathsf{curr}}}$

- **Fit**(X, Y): Función que devuelve verdadero si el volumen X se ajusta dentro del volumen Y, y falso en caso contrario.
- **Find-Workspace**(B): Función que devuelve la transformación $T_{\mathsf{W}}^{\mathsf{vH}}$ que localiza a vH tal que la caja B se encuentre centrada en $\mathcal{W}_{\mathsf{vH}}$.

5.3.2. Guiado de sincronización

Una vez obtenido $T_{\mathsf{vH}_{\mathsf{new}}}^{\mathsf{vHIP}_{\mathsf{new}}}$ a través del Algoritmo 7, el proceso de escalado detallado en el Apartado 3.2.1 puede ser invertido para obtener $T_{\mathsf{H}}^{\mathsf{HIP}_{\mathsf{new}}}$. El movimiento desde la posición actual del dispositivo háptico HIP_{curr}, hacia la nueva posición HIP_{new} es descrita por la siguiente transformación (Figura 5.6 inferior):

$$T = \left[T_{\mathsf{H}}^{\mathsf{HIP}_{\mathsf{curr}}}\right]^{-1} T_{\mathsf{H}}^{\mathsf{HIP}_{\mathsf{new}}} \tag{5.7}$$

El dispositivo háptico puede ejercer fuerzas en la dirección definida por $pos(T)$ con el objetivo de guiar el movimiento. A medida que el usuario mueva el dispositivo háptico, $T_{\mathsf{H}}^{\mathsf{HIP}_{\mathsf{curr}}}$ continuamente es actualizada la transformación T, es por esta razón que resulta como un campo magnético que atrae HIP_{curr} hacia HIP_{new}, tal y como se ilustra en la Figura 5.7 para un caso simplificado en 2D.

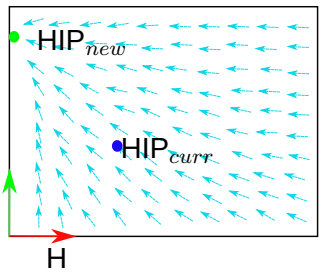


Figura 5.7: Campo de fuerza para guiar al usuario hacia HIP$_{new}$.

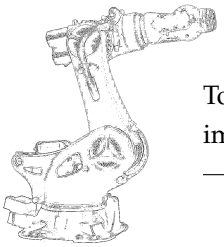
El usuario tiene en todo instante el control del momento en el cual quiere continuar con la teleoperación, es decir que el usuario puede esperar o no a alcanzar la solución propuesta HIP$_{new}$. La fuerza de guiado hacia HIP$_{new}$ desaparece tan pronto como se reanude la teleoperación.

5.4. Conclusiones

Se han presentado los métodos tanto para general las fuerzas de guiado durante la teleoperación activa, como las fuerzas que se le pueden brindar al usuario durante la desconexión del dispositivo remoto con el ánimo de reposicionar el espacio de trabajo virtual del dispositivo háptico dentro de la escena, lo que permite al teleoperador continuar con la tarea desde una ubicación con mayor alcance o comodidad. También se ha descrito la ayuda para cruzar puntos críticos de la trayectoria y que se usa principalmente para ayudar al usuario en el cambio de configuración cinemática del robot teleoperado, ya que esta acción no se puede llevar a cabo sin un adecuado soporte.

6

Marco de simulación y experimentación



To invent, you need a good imagination and a pile of junk.

Thomas A. Edison

En este capítulo se expone el conjunto de desarrollos realizados a manera de infraestructura de *software* que permiten no solo simular sino llevar a cabo la teleoperación con ayudas de guiado háptico. Se describen los modelos utilizados tanto de los robots como del problema en general, que son usados por el planificador para brindar la solución a la tarea propuesta. Esta planificación luego es usada como base para la generación de las ayudas a la teleoperación. En este capítulo también se hace una descripción de la infraestructura física que particulariza los modelos y la estructura de módulos y comunicaciones necesarias para llevarlas a cabo.

El modelo propuesto es simple pero potente y generalizable, capaz de manejar sistemas complejos que incluyen uno o más robots con estructura cinemática de tipo árbol con varios grados de libertad. El enfoque está basado en la definición del espacio de muestreo reducido como una combinación lineal de los grados de libertad del robot.

La escena está compuesta por uno o varios robots industriales antropomórficos que comparten, al menos en parte, su espacio de trabajo y obstáculos tanto dinámicos como estáticos con los cuales se va a interactuar. Estos robots pueden tener su base fija o estar acoplados a sistemas que les permitan tener uno o varios grados de libertad adicionales, como por ejemplo rieles o plataformas móviles.

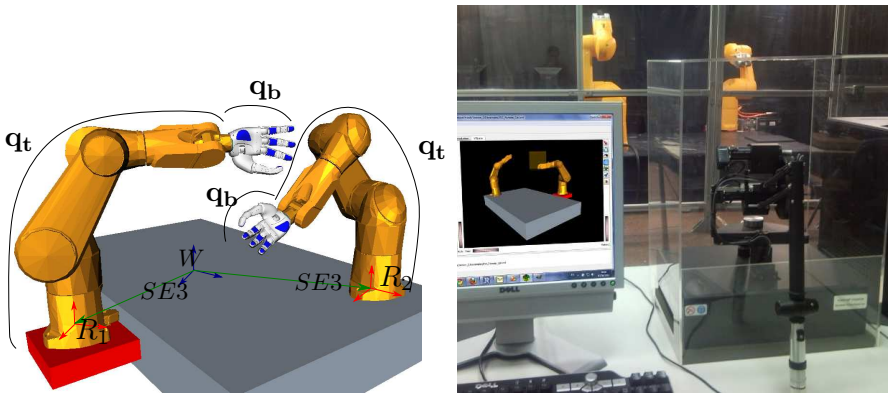


Figura 6.1: Modelo del laboratorio de robótica del IOC, compuesto por dos robots Stäubli, de los cuales uno tiene la base fija y el otro la tiene montada sobre un riel. Están equipados con manos robóticas: una antropomórfica (*Schunk Anthropomorphic Hand*) y una diestra (*Schunk Dexterous Hand*). Tareas cooperativas complejas que involucran 39 *gdl*.

Es de resaltar que aunque el modelo aquí propuesto es generalizable para robots equipados con manos mecánicas, el alcance del trabajo de teleoperación está enfocado únicamente a brazos robots industriales.

6.1. Robots

El modelo general de robot está basado en las estructuras cinemáticas de tipo árbol con una base móvil en *SE(3)*. La cadena se divide en el tronco y en las ramas. Los eslabones que forman parte del tronco están etiquetados como tal en el archivo descriptivo del robot que se utiliza como entrada. Como un ejemplo, la Figura 6.1 muestra una configuración de celda a teleoperar compleja que consta de dos robots industriales equipados con manos mecánicas antropomórficas, donde uno de ellos tiene la base fija y el otro está montado sobre una base móvil que le confiere un grado de libertad más. Estos robots están modelados como árboles cinemáticos donde los eslabones del manipulador componen el tronco y los eslabones de los dedos de la mano mecánica las ramas.

Cada uno de los eslabones de esta estructura cinemática con forma de árbol tiene una posición y orientación absoluta con respecto al sistema coor-

denado de referencia global (W) denominada T_W^i:

$$T_W^i = T_W^p \cdot T_p^i \tag{6.1}$$

donde T_W^p es la transformación absoluta del padre y $T_p^i = T_{(\alpha,a,\theta,d)}$ es la transformación local entre el eslabón padre y el actual, obtenida a través del procedimiento de Dennavit–Hartemberg que se parametriza con las variables α, a, θ y d (con cualquiera de las dos metodologías, la estandar o la modificada). En aquellos pocos casos donde no es posible definir adecuadamente la transformación local usando los parámetros D–H, se agrega una pretransformación que le confiere mayor flexibilidad:

$$T_p^i = T_{\text{pre}} \cdot T_{(\alpha,a,\theta,d)} \tag{6.2}$$

El primer eslabón de la cadena cinemática, normalmente llamado la base, define la posición de *home* del robot dentro de la escena. La base puede ser estática o móvil, en cuyo caso los límites de este movimiento permitido se deben estipular. El adecuado uso de los limites y la conveniente parametrización de los *gdl* de la base, permiten una simple pero eficiente forma de adaptar el modelo a los movimientos del robot en diferentes situaciones comunes (Apartado 6.2.2).

Todos los datos relativos a cada eslabón del robot estan definidos en un archivo XML diseñado para este fin, que incluye el nombre, la representación gráfica, el nombre del eslabón padre, los parámetros D–H, el tipo de junta (rotacional/prismática) y sus límites (ver Apéndice B).

Desde el punto de vista geométrico para la validación de las colisiones, se usa la representación de mallas triangulares de los objetos en el espacio físico. Los objetos tridimensionales que representan los eslabones de los robots y los demás elementos del entorno, como los obstáculos, pueden ser elementos geométricos básicos como cilindros, conos o esferas, o geometrías más complejas representadas como mallas triangulares. Estos objetos, que pueden tener una forma de representación más condensada, se convierte en mallas triangulares para ser utilizadas en los procesos de validación de las distintas configuraciones del (de los) robot(s) presente(s) en la escena y así determinar cuando una de ellas se encuentra dentro de espacio libre o por el contrario hace parte de la frontera o del interior de un obstáculo en el espacio de configuraciones $\mathcal{C}$. Este proceso de valida-

ción de colisiones es muy costoso computacionalmente y aumenta con el refinamiento que puedan tener las mallas triangulares, es por ello que debe usarse con moderación en aras de la eficiencia ya que una malla muy refinada será más precisa al momento de validar las colisiones de dos elementos muy cercanos, pero irá en contra de los requerimientos de tiempo de cómputo en las situaciones donde las distancias sean mayores.

6.2. Espacio de configuraciones

Como se venía diciendo en apartado anterior, cualquier robot puede ser considerado, en términos generales, como una estructura cinemática de árbol compuesta por un tronco y algunas ramas y con su base móvil. Entonces, un robot genérico tendrá $n_{se} + n_t + n_b$ *gdl*, donde el n_{se} corresponde a los grados de libertad tanto traslacionales como rotacionales de la base móvil en *SE(3)*, n_t es el número de articulaciones en el tronco y n_b los de las ramas, y por esto una configuración puede ser expresada de la siguiente manera:

$$\hat{\mathbf{q}} = \begin{bmatrix} \hat{\mathbf{q}}_{\mathbf{se3}} \\ \hat{\mathbf{q}}_{\mathbf{t}} \\ \hat{\mathbf{q}}_{\mathbf{b}} \end{bmatrix} \tag{6.3}$$

donde $\hat{\mathbf{q}}_{\mathbf{se3}}$, $\hat{\mathbf{q}}_{\mathbf{t}}$ y $\hat{\mathbf{q}}_{\mathbf{b}}$ son, respectivamente, los valores normalizados en el rango $[0; 1]$ de los *gdl* tanto de la base como de tronco y de las ramas.

Sea $\mathbf{d_r} = [\mathbf{d_{se3}}, \mathbf{d_t}, \mathbf{d_b}, 1]^T$ un vector de controles[1] usado para manipular los grados de libertad del robot, siendo $\mathbf{d_{se3}}$, $\mathbf{d_t}$ y $\mathbf{d_b}$ los subconjuntos dedicados a los grados de libertad de la base, a los del tronco y a los de las ramas, respectivamente. Si se define un control por cada grado de libertad, entonces la dimensión de $\mathbf{d_{se3}}$, $\mathbf{d_t}$ y $\mathbf{d_b}$ será 6, n_t y n_b, respectivamente. Sin embargo, se pueden usar menos controles si el robot no tiene en realidad todos los grados de libertad activos, o si algunos de ellos están acoplados. Cada control está definido dentro del rango $[-0{,}5; 0{,}5]$.

Usando el vector de controles $\mathbf{d_r}$, los grados de libertad de todo el mecanismo se pueden obtener con la siguiente expresión:

$$\hat{\mathbf{q}} = K_r \mathbf{d_r} \tag{6.4}$$

[1] Aquí el termino control no hace referencia a ningún elemento de la teoría de control de sistemas, sino que se utiliza en su contexto más amplio de dispositivo de mando

donde se hace uso de la matriz K_r que está definida así:

$$K_r = \begin{bmatrix} A & 0 & 0 & \mathbf{O_{se3}} \\ 0 & B & 0 & \mathbf{O_t} \\ 0 & 0 & C & \mathbf{O_b} \end{bmatrix} \tag{6.5}$$

aquí A es una matriz de dimensiones $6 \times a$ con $a \leq 6$, el número de controles de la base, B es una matriz $n_t \times b$ con $b \leq n_t$ el número de controles del tronco, C es una matriz $n_b \times c$ con $c \leq n_b$ el número de controles de las ramas, y con $\mathbf{O_{se3}}$, $\mathbf{O_t}$ y $\mathbf{O_b}$ que son vectores con los valores de compensación (*offset*).

Este enfoque resulta beneficioso en conjunto con los planificadores de movimiento basados en muestreo, dado que la generación de muestras puede ser llevada a cabo en este espacio de controles porque cada vector de controles define unívocamente una configuración del robot y adicionalmente este espacio es un hipercubo unitario centrado en el origen, lo que facilita tareas como la verificación de límites y la búsqueda de muestras con diversos criterios de cubrimiento y dispersión.

El muestreo, que los planificadores realizan para solucionar la tarea establecida, se realiza en este espacio de control. Los vectores de controles $\mathbf{d_r}$ se obtienen de un hipercubo unitario y su configuración correspondiente se calcula a través de la Ec. (6.4). Se han incorporado muestreadores tanto de tipo aleatorio como determinísticos, estos último basados en la secuencia Halton (Halton, 1960b) y en la secuencia $s_d(k)$ (Rosell et al., 2007a). Cada muestra se almancena con la información correspondiente a los dos espacios, tanto como un conjunto de coordenadas en este espacio de controles, como la configuración que representa. Es necesario precisar que aunque el muestreo se lleve a cabo en este espacio de controles, la planificación de caminos se lleva a cabo en el respectivo espacio de configuraciones $\mathcal{C}$, es decir que la búsqueda de vecinos se realiza con la métrica correspondiente y que los planificadores locales tratan de conectar las muestras vecinas interpolando correctamente en el espacio de configuraciones.

En los siguientes apartados se detalla la paremetrización utilizada para el espacio *SE(3)*, la particularización de las matrices A, B y C utilizada en alguno problemas comunes, cómo es tratado el acoplamiento entre grados de libertad y la extensión de esta metodología de modelado a escenarios con multiples robots.

6.2.1. Parametrización de *SE(3)*

Dado $\hat{\mathbf{q}}_{\mathbf{se3}} = [\hat{x}, \hat{y}, \hat{z}, x_1, x_2, x_3]^T$, el vector que define la configuración normalizada de la base del robot, entonces las coordenadas de traslación $[x, y, z]$ son calculadas considerando los límites de su espacio de trabajo $[x_{min},\ y_{min},\ z_{min}]$ y $[x_{max},\ y_{max},\ z_{max}]$ así:

$$\begin{aligned} x &= x_{min} + \hat{x}(x_{max} - x_{min}) \\ y &= y_{min} + \hat{y}(y_{max} - y_{min}) \\ z &= z_{min} + \hat{z}(z_{max} - z_{min}) \end{aligned} \tag{6.6}$$

Las coordenadas de rotación $[q_x, q_y, q_z, q_w]^T$ del cuaternión que define la orientación de la base, son calculadas de forma muy similar a como está propuesto por Kuffner (2004):

$$\begin{aligned} q_x &= r_1 \sin(\theta_1) & & r_1 = \sqrt{x_1} \\ q_y &= r_1 \cos(\theta_1) & \text{con} \quad & r_2 = \sqrt{1 - x_1} \\ q_z &= r_2 \sin(\theta_2) & & \theta_1 = 2\pi x_2 \\ q_w &= r_2 \cos(\theta_2) & & \theta_2 = \pi x_3 \end{aligned} \tag{6.7}$$

Esta parametrización que se representa en (6.7), difiere de la propuesta original de Kuffner (2004) dado que para este autor la terna $(x_1, x_2, x_3) = \mathbf{0}$ representa el cuaternión $q = (1; 0; 0; 0)$ y en el presente trabajo se quiere equiparar los parámetros en cero con la rotación nula, es decir que para la terna $(x_1, x_2, x_3) = \mathbf{0}$ el cuaternión asociado es $q = (0; 0; 0; 1)$.

6.2.2. Robot básicos

El modelo propuesto puede ser fácilmente configurado para tratar los tipos más comunes de robots que aparecen en los problemas del área de planificación de movimientos. Por ejemplo, robots del tipo cuerpo rígido con movimientos tanto en el plano como en el espacio (tanto *SE(2)* como *SE(3)*), son modelados haciendo que las matrices B y C, y los vectores $\mathbf{d_t}$, $\mathbf{d_b}$, $\mathbf{O_t}$, y $\mathbf{O_b}$ sean cero.

En el caso de *SE(3)*, se define $\mathbf{O_{se3}} = 0{,}5[1; 1; 1; 1; 1; 1]$ y la matriz A como una identidad de 6×6. Para el caso de *SE(2)* se hace $\mathbf{O_{se3}} = 0{,}5[1; 1; 1]$

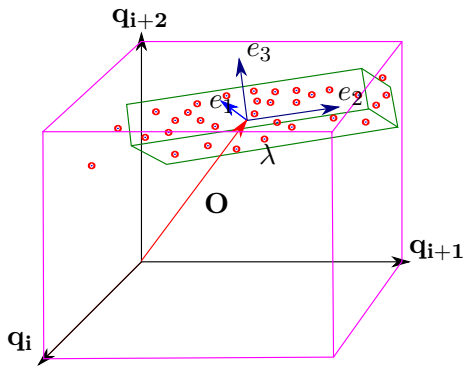


Figura 6.2: Análisis de componentes principales aplicado a un espacio tridimensional: el uso de la dirección(es) con mayor varianza (e_1 en este caso) captura el acoplamiento entre los *gdl* y reduce de forma efectiva la cantidad de parámetros del problema.

y:

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{6.8}$$

Si el robot es un manipulador industrial de seis articulaciones y con la base fija, se puede modelar como una cadena cinemática fija, es decir que las matrices A y C y los vectores $\mathbf{d_t}$, $\mathbf{d_b}$, $\mathbf{O_t}$, $\mathbf{O_b}$ son cero y la matriz B es una identidad de 6×6 y el vector de compensación es $\mathbf{O_t} = 0{,}5[1; 1; 1; 1; 1; 1]$.

6.2.3. Robots con grados de libertad acoplados

En algunos mecanismos pueden existir acoplamientos constructivos entre sus grados de libertad o se pueden imponer algunos acoplamientos artificiales que pueden reducir la dimensión de los problemas u obtener un cierto comportamiento del mecanismo. Algunas manos mecánicas tienen acoplamientos entre las falanges media y distal de sus dedos, como ejemplo de estos acomplamientos constructivos, y que pueden ser modelados fácilmente con este enfoque. Los acoplamientos artificiales pueden obtenerse a través de análisis matemáticos sobre conjuntos de configuraciones que revistan un comportamiento especial y que pueden ser modelados a tra-

vés de análisis de componentes principales (PCA). Como resultado de este análisis se obtiene una nueva base (Figura 6.2), que puede ser de un número menor de componentes y que capturan el acoplamiento de las muestras y a la vez pueden reproducir la mayor cantidad de resultados similares a los expresados con la base original. Los vectores pertenecientes a esta nueva base con los mayores autovalores son seleccionados (es decir, aquellos que definen las direcciones donde existe la mayor dispersión), y son codificadas como una matriz columna C. En trabajos internos del IOC, se ha conseguido que los 13-*gdl* de una mano antropomórfica Schunk (SAH) sean manejados adecuadamente por los tres primeros vectores propios que ha resultado del proceso PCA, en este caso se usa una matriz C de 13×3. Pueden procesarse con esta misma técnica las configuraciones obtenidas durante la realización de tareas cooperativas entre dos o más robots industriales antropomórficos y encontrar con ellas una nueva base que, a la vez que reduce la dimensión del problema, puede expresar de forma sencilla los movimientos acoplados que se pueden presentar.

6.2.4. Espacios múltirobot

Este modelo también puede ser extendido al caso de múltiples robots en una forma directa y fácil:

$$\hat{\mathbf{q}}_\mathbf{s} = \begin{bmatrix} \hat{\mathbf{q}}_{\mathbf{r1}} \\ \hat{\mathbf{q}}_{\mathbf{r2}} \\ \vdots \\ \hat{\mathbf{q}}_{\mathbf{rm}} \end{bmatrix} = \begin{bmatrix} K_{r1} & 0 & \cdots & 0 \\ 0 & K_{r2} & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & 0 & K_{rm} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{d}_{\mathbf{r1}} \\ \mathbf{d}_{\mathbf{r2}} \\ \vdots \\ \mathbf{d}_{\mathbf{rm}} \end{bmatrix} \tag{6.9}$$

donde el vector $[\mathbf{d_{r1}}\ \mathbf{d_{r2}}\ \cdots\ \mathbf{d_{rm}}]^T$ resulta ser el vector de controles del conjunto de robots.

Con este modelo generalizado se pueden tratar los siguientes tipos de robot y sus combinaciones en un espacio de configuraciones unificado así:

- Cuerpo libre $\rightarrow$ *SE(3)*.
- Cadena cinemática fija $\rightarrow \mathbb{R}^n$.
- Cadena cinemática móvil $\rightarrow \langle$ *SE(3)*,$\mathbb{R}^n \rangle$.

De esta forma el espacio de configuración correspondiente al espacio de trabajo, está dado por el conjunto de robots que se encuentren en la escena,

es decir que si existen dos robot antropomórficos uno fijo y otro móvil, el espacio de configuración resulta ser un espacio $\{\mathbb{R}^n \times SE(3) \times \mathbb{R}^n\}$, y la métrica utilizada para la definición de vecindades y conectividad estará dado por:

$$distance = \left(\sum_{i=1}^{m} \left(dist(SE3_{1i}, SE3_{2i})^2 + dist(\mathbb{R}^n_{1i}, \mathbb{R}^n_{2i})^2 \right) \right)^{\frac{1}{2}} \tag{6.10}$$

donde m es el número de robots presentes en el problema.

6.3. Infraestructura física

En el IOC, se cuenta con dos laboratorios, uno denominado laboratorio local y el otro denominado laboratorio remoto, por su dedicación a funcionar como cada uno de los lugares del sistema de teleoperación. En el nodo–mando se dispone de un dispositivo háptico con su respectivo computador, mientras que en el nodo–operación se cuenta con la celda de manufactura flexible provista de dos robots, las cámaras de realimentación de video, el computador central y las respectivas controladoras de cada uno de los robots.

6.3.1. Robots

El laboratorio remoto cuenta con dos robots Staübli TX90 con su controladora CS8 (Figura 6.3a). Estos brazos robot antropomórficos cuentan con 6 *gdl* rotacionales y 8 soluciones de cinemática inversa. Estas soluciones corresponden a las diferentes configuraciones que puede tomar el robot, haciendo referencia a la posibilidad de ubicar como brazo derecho o izquierdo, con el codo arriba o abajo y usando valores angulares de la muñeca ya sean positivos o negativos (*Shoulder, Elbow, Wrist*).

Adicionalmente, se cuenta con un robot ligero KUKA (*Lightweight Robot* o LWR) de 7 *gdl* (Figura 6.3b). Este robot tiene la particularidad de tener articulaciones flexibles y control de pares en cada una de ellas, lo que le permite funcionar a manera de dispositivo háptico. En este modo puede ser manipulado aplicándole fuerzas externas sobre los diferentes eslabones que lo conforman, para modificar la posición y orientación de cada uno de ellos.

(a) Staübli TX90 (b) KUKA LWR

Figura 6.3: Robots industriales y sus controladoras.

Figura 6.4: Dispositivos hápticos Omni y Premium. Físicamente no existen diferencias entre la versión normal y la *highforce* del modelo Premium.

6.3.2. Dispositivos hápticos

Se dispone de tres dispositivos hápticos Phantom® de la casa Sensable, un Omni®, un Premium®, y un Premium® *high-force*. La conexión de estos dispositivos con el computador se hace vía puerto *firewire* para el primer modelo, y usando el puerto paralelo para los dos últimos. Los dispositivos de la línea Premium® permiten realimentar 6 *gdl*, o sea que se pueden realimentar tanto fuerzas como torques, mientras el que Omni® sólo permite realimentar fuerza. La diferencia entre los dos modelos Premium radica en la magnitud de las fuerzas y torques máximos que pueden aplicar al usuario.

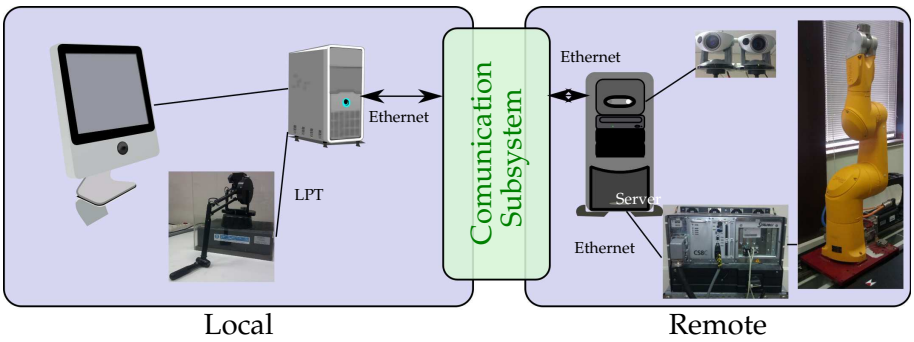


Figura 6.5: Diagrama de comunicaciones entre los diversos dispositivos.

6.3.3. Red y comunicaciones

Se cuenta con una red dedicada de 1Gbit/seg. que permite tener comunicaciones de grandes transferencias de información y baja latencia. Esta red conecta los computadores de los nodos locales con el computador central del nodo remoto y a su vez conecta a la red el servidor de video dedicado que provee las imágenes de las diferentes cámaras con las que se cuenta dentro la celda remota.

6.4. Infraestructura de *Software*

La infraestructura necesaria para el desarrollo de este proyecto ha sido concebida y desarrollada como una herramienta de planificación y teleoperación modular y extensible siguiendo las directivas que fueron propuestas en (Pérez y Rosell, 2010). Esta herramienta ha sido usada satisfactoriamente para probar varios algoritmos de planificación basados en muestreo y para llevar a cabo la teleoperación simultánea de los robots Staübli TX90 con los que cuenta la celda del laboratorio del IOC con la ayuda de dos dispositivos hápticos Phantom. Todo este marco de trabajo se ha denominado el proyecto *The Kautham Project*.

Conceptualmente (Figura 6.6), *The Kautham Project* está compuesto por un sistema de visualización y simulación cinemática de los robots, una herramienta de planificación provista de varios planificadores basados en muestreo y el módulo que permite manejar el dispositivo háptico y comandar el robot remoto. Desde el punto de vista de implementación, se ha elaborado un conjunto de módulos que se encarga desde el modelado de

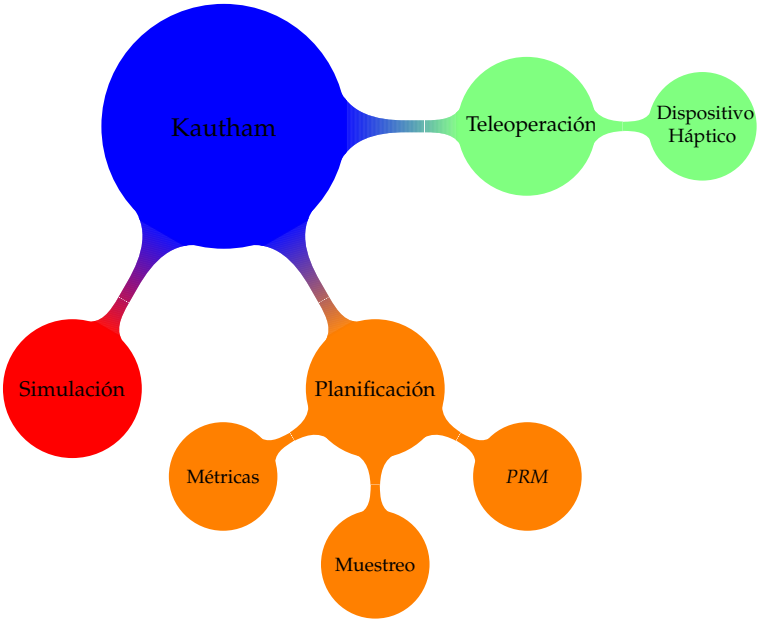


Figura 6.6: En esta figura se puede apreciar un mapa conceptual del proyecto *The Kautham Project* y las utilidades que proporciona tanto al programador como al usuario.

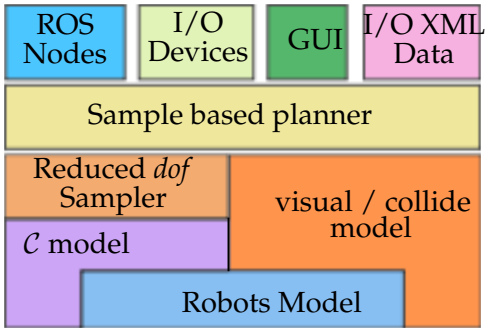


Figura 6.7: Módulos que componen el proyecto *The Kautham Project*. El núcleo es compuesto por los cuatro modulos inferiores. Las dos capas superiores corresponde a las aplicaciones desarrolladas hasta hoy.

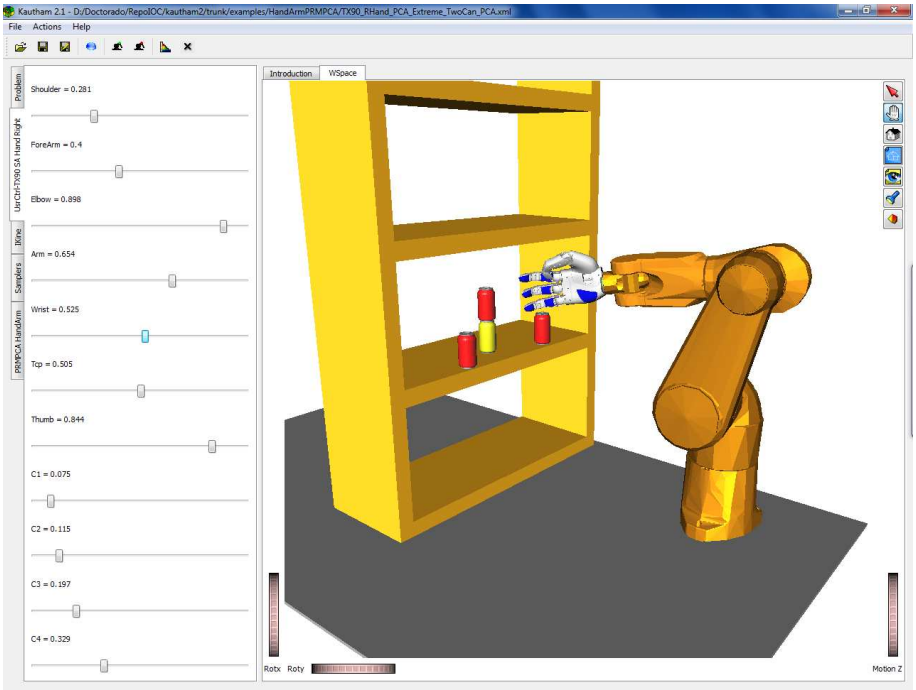


Figura 6.8: Interface grafica de usuario en el nodo–mando

los robots, el espacio de configuraciones y el sistema de muestreo, hasta la planificación de caminos y la comunicación con los diferentes dispositivos (Figura 6.7).

De cara al usuario, se presenta una interfaz gráfica que le permite configurar parámetros de los planificadores, generar muestras siguiendo diferentes pautas y esquemas de generación, elegir las caracteristicas de la teleoperación, visualizar el modelo de la celda remota e ir siguiendo en tiempo real los movimientos del robot. En la Figura 6.8 se puede apreciar de forma general el aspecto de la interfaz que tiene a su disposición el usuario en el nodo local.

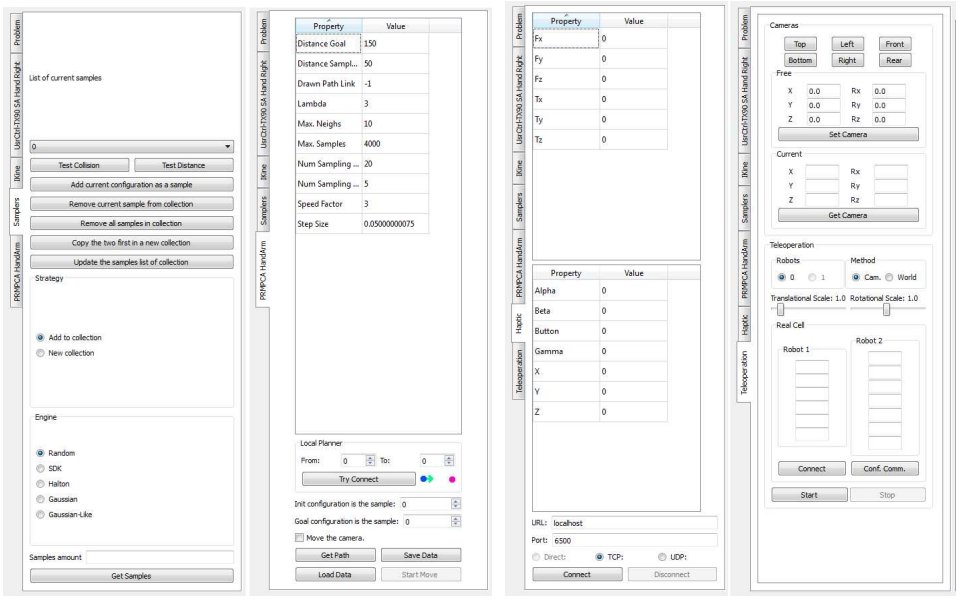
6.4.1. Herramientas de *software*

La infraestructura de software en la que se basa el sistema de teleoperación ha sido desarrollado en C++ usando varias librerías de código abierto y multi-plataforma haciendo uso de los compiladores nativos de cada plataforma (Pérez y Rosell, 2010). La representación gráfica tridimensional del entorno de trabajo de los robots está basada en Inventor con la ayuda de

la librería Coin3D (www.coin3d.org). El resto de componentes gráficos de las interfaces de usuario están basadas en la librería Qt (qt.nokia.com). El sistema de validación de colisiones de los objetos tridimensionales esta desarrollado usando la librería PQP (Eric Larsen y Manocha, 2000) y varias de las librerías Boost (Graph, Date Time, Interprocess) se utilizan tanto para manejar los grafos de los planificadores como para manejar espacios de memoria compartida entre procesos.

Los dos paradigmas de programación utilizados son la programación orientada a objetos (OOP) y la de componentes de software (*Software Components*). En la actualidad la programación orientada a objetos es la metodología que brinda un buen balance entre flexibilidad para reutilizar código y abstracción de los datos con los que trabaja, sin contar con las ventajas de la utilización de las clases genéricas. Por otra parte la descomposición de grandes sistemas de *software*, en pequeños componentes con funciones bien definidas y que se comunican a través de mensajes ha demostrado ser una buena metodología de implementación de las arquitecturas de control y operación de robots como lo ha venido demostrando ROS (*Robot Operating System* disponible en www.ros.org/wiki) y OROCOS (www.orocos.org). La capa de comunicaciones se ha implementado en ROS usando en su mayoría la estrategia de publicadores y suscriptores y también haciendo uso de algunos servicios para la configuración de los mismos. Este sistema ha sido denominado *The Kautham Project* y se encuentra disponible como un proyecto de código abierto en sir.upc.es/kautham. Este sistema mantiene la característica de ser multi-plataforma y ha sido probado con éxito sobre Windows XP y 7, y sobre Linux Debian Squeeze y Ubuntu 10.10 y 11.04.

Actualmente el sistema puede ser usado como simulador de teleoperación (Apartado 6.4.2) con lo cual puede ejecutarse contando únicamente con el dispositivo háptico que brinda la realimentación de fuerzas al operador, o como parte del sistema de teleoperación global que cuenta adicionalmente con una celda multirobot (Apartado 6.4.3). Cuando es usado como simulador, éste puede ser ejecutado tanto en Windows como en Linux, pero cuando se teleopera la celda real, deben intervenir de forma obligatoria algunos componentes en Linux dada la naturaleza del ambiente ROS (Apartado 6.4.3). Es por esta razón que algunos componentes deben ser compilados o creados con compilación cruzada en máquina Linux.



(a) Interface con el planificador. (b) Interface gráfica de la teleoperación.

Figura 6.9: En la figura de la izquierda se puede ver la interface con el planificador, donde se pueden configurar los parámetros de cada planificador. En la de la derecha aparece la interface gráfica de la teleoperación, donde se pueden escoger la cámara activa, las escalas y conectarse o desconectarse de la celda remota.

6.4.2. Planificación y teleoperación

La interface adopta diferentes aspectos de acuerdo a la tarea que se vaya a desarrollar. El la Figura 6.9 se pueden ver las diferentes pestañas que se pueden abrir en el panel izquierdo de la aplicación y que se utilizan para ingresar los parámetros de funcionamiento tanto de los planificadores como de las características de las comunicaciones con los dispositivos cuando se va a realizar una teleoperación. Inicializar los dispositivos hapticos, configurar y ejecutar los nodos de comunicación ROS, elegir la cámara activa entre otras tareas, son las que se pueden realizar a través de los diferentes elementos que proporciona esta aplicación.

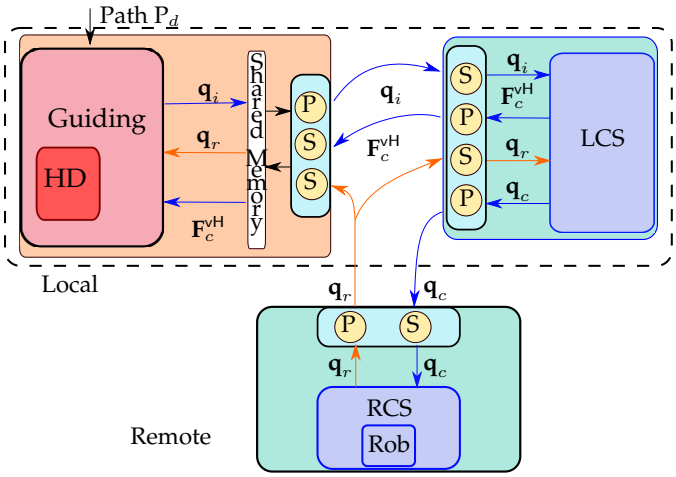


Figura 6.10: Esquema de implementación del nodo–mando y del nodo–operación ejecutándose en diferentes máquinas para una teleoperación que involucra un único robot.

6.4.3. Nodo–mando

En el caso del nodo–mando, por ejemplo, el componente de control del dispositivo háptico Phantom® debe ser implementado en Windows porque el fabricante sólo provee un controlador (*driver*) actualizado para este sistema operativo. De allí que para comunicar este componente con el resto y dadas las particularidades de ROS, se ha creado un módulo por compilación cruzada que lo vincula con el resto de infraestructura en Linux.

Existen varias alternativas de implementación. Actualmente, el nodo–operación del sistema se ejecuta en Linux y en el nodo–operación, el módulo *Local Control System* o LCS lo hace en Linux, mientras que todo el subsistema de guiado lo hace en Windows. El núcleo del subsistema de guiado controla el dispositivo háptico a través de la librería nativa provista por el fabricante y se conecta a un nodo de comunicaciones compuesto por un publicador (P) y dos suscriptores (S), que han sido obtenidos por compilación cruzada utilizando la herramienta eros ofrecida dentro de ROS para este propósito.

El método utilizado para intercambiar información entre estos dos componentes dentro de la máquina local en Windows, es la memoria compartida entre procesos o *IPC shared memory* incluido en la librería Boost (Gaztanaga, 2011), como se muestra en la Figura 6.10. En otra posible alternativa de implementación, el módulo fundamental de guiado puede ser compi-

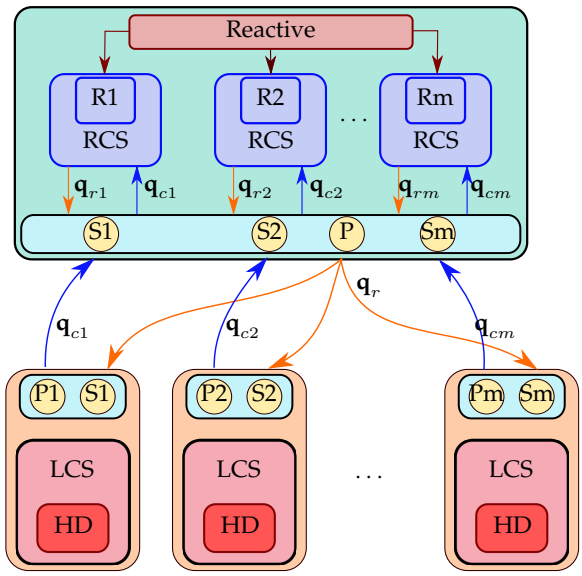


Figura 6.11: Esquema de funcionamiento del nodo remoto cuando existe más de un robot en la celda remota. Para un teleoperador, los robots que no está comandando se convierten en obstáculos dinámicos.

lado para ser ejecutado en Linux, y en ese caso el dispositivo háptico es aislado en un nodo ROS independiente, obtenido por compilación cruzada, haciendo posible que el guiado y el sistema de control sean integrados en un único nodo.

6.4.4. Nodo–operación

Cada uno de los robots presentes en la celda remota va a ser controlado por un operador diferente, y estos operadores pueden estar ubicados en lugares diferentes. En la Figura 6.11 se puede observar la forma en la que interactua este nodo–operación con los diversos nodos–mando que teleoperan cada uno de los robots. De esta forma se compagina el trabajo de cada uno de los usuarios en el módulo de control ubicado en el computador central de la celda remota. Este ordenador centraliza las tareas de comunicación con los diferentes nodos–mando y ejecuta tanto el sistema reactivo de modificación de los parámetros B_r de los controladores como el mismo controlador **P+d** de cada uno de los robots.

En esta figura se observa que la información de la configuración de todos los robots dado por $\mathbf{q}_r = \{\mathbf{q}_{r1}, \mathbf{q}_{r2}, \ldots, \mathbf{q}_n\}$ es enviada a cada uno de los

nodos–mando, ya que los otros robots que no está teleoperando el usuario son considerados como obstáculos móviles y que el operador debe evitar con la ayuda de los sistemas de guiado propuestos en este trabajo.

Las comunicaciones con el nodo–operación se hacen utilizando la estrategia de las parejas publicador–suscriptor de ROS mientras que las comunicaciones entre este computador central y cada una de las controladoras de los robots se hace utilizando un protocolo SOAP que provee directamente el fabricante. Sin embargo esta comunicación puede implementarse a bajo nivel con las utilidades proporcionadas para tal fin por los diferentes fabricantes de robots y que en este caso particular caería dentro del dominio de las librerías LLI (*Low Level Interface*).

6.4.5. Extensiones del simulador

El modo de simulación ha sido utilizado para modelar otros tipos de robots que no son objeto de ser teleoperados ya que cae fuera del alcance de este proyecto, pero que ha servido para enriquecer el trabajo echo en la planificación de caminos. Este es el caso de los trabajos realizado en prensión de objetos con manos mecánicas y con la planificación para la broncoscopía.

Prensión: Para las tareas de prensión se ha tenido en cuenta el brazo robot con la palma de la mano como el tronco de la estructura y los dedos como cada una de las ramas que lo componen. Como punto de partida el modelo incorpora los acoplamientos que presentan las falanges media y distal de cada dedo y posteriormente el modelo se ha extendido para modelar acoplamientos virtuales que aparecen en los movimientos naturales que hace la mano humana. En el programa se incluye un archivo que describe las características de un robot del tipo árbol y particularmente los parámetros constructivos de la mano antropomórfica Schunk.

Kautham ROS: Es una implementación orientada a la estructura ROS que tiene las principales funciones del proyecto Kautham y está provista de una capa de comunicaciones formada por dos nodos ROS: el nodo de planificación y el nodo de visualización. Estos nodos están configurados como servidores de forma que un cliente pueda tener acceso a los diferentes servicios que proveen (Figura 6.12). El nodo ROS de

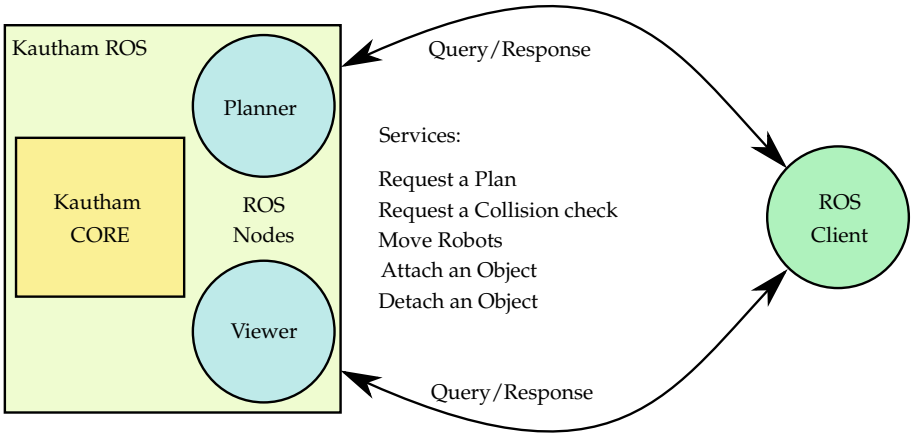


Figura 6.12: Nodos ROS y servicios implementados del proyecto Kautham.

planificación tiene servicios para solicitar un plan, para validar la colisión o no de una configuración, y para adjuntar o despegar un objeto del robot. Por su parte, el nodo ROS de visualización tiene una interfaz gráfica de usuario que muestra la escena y tiene servicios para actualizar la configuración de los robots y la posición y orientación de los obstáculos del entorno.

Validación de colisiones: Basado en el modelo del robot a teleoperar y su entorno, se ha desarrollado un nodo ROS a manera de filtro que valida la configuración comandada desde los dispositivos de entrada, en este caso un guante sensorizado y un *tracker* de posición y orientación de la muñeca, de forma que se puedan evitar las colisiones del mismo.

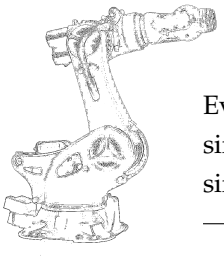
Kautham Consola: Se ha desarrollado también una aplicación de consola que permite ejecutar en modo de lotes las pruebas de diferentes planficadores, o de un planificador con diferentes parámetros tanto en un *cluster* de computadores como en un computador con multiples procesadores, de forma que se hace uso intensivo de los procesadores disponibles, dada la capacidad de paralelización de procesos que proporciona la librería MPICH2 (Gropp et al., 1999).

6.5. Conclusiones

A lo largo de este capítulo se han descrito los modelos realizados para representar los robots, los espacios de configuración asociados, el problema de planificación, el camino solución y la tarea de teleoperación. Estos modelos han sido utilizados para la implementación de la infraestructura de *software* que interconecta todos los elementos tanto físicos como lógicos del sistema de teleoperación.

7

Evaluación y Resultados



Everything must be made as simple as possible. But not simpler.

Albert Einstein

En este capítulo se detallan los experimentos realizados para evaluar el desempeño de cada una de las partes del sistema de guiado háptico desarrollado con el presente trabajo.

7.1. Teleoperación

Dentro del marco de la teleoperación en el espacio Cartesiano, existe la libertad de asociar los movimientos del dispositivo háptico y del robot más allá de las restricciones cinemáticas de cada uno de ellos. El vínculo virtual que se crea entre los dispositivos durante la teleoperación, no tiene por qué ser fija ni en el tiempo ni a una configuración particular.

En el enfoque establecido en este trabajo para la creación del vínculo entre el dispositivo háptico y el robot se han tenido en cuenta dos características muy importantes:

- Mantener la relación mano–ojo entre los movimientos realizados con el dispositivo maestro y los que el ojo realimenta de los realizados en por el robot esclavo.

- Ofrecer realimentación de fuerzas con el dispositivo háptico al operador, sin crearle confusiones o malas interpretaciones, en el tipo de respuesta que debe dar a un estímulo aplicado por el sistema.

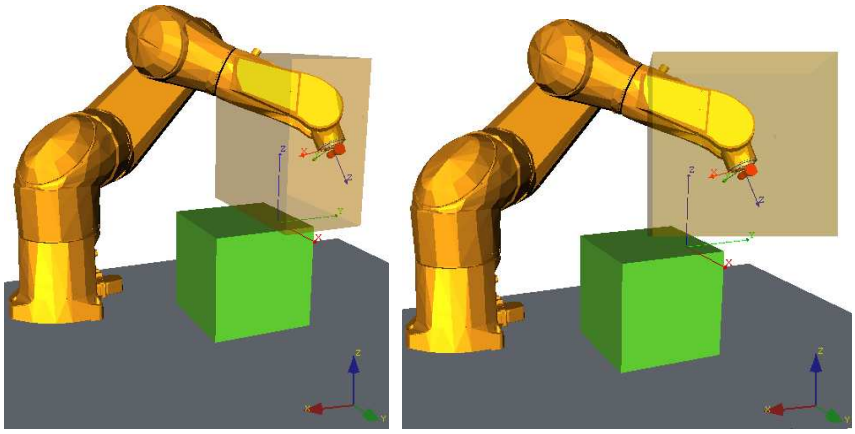
Con la finalidad de validar estas premisas se elaboran dos experimentos. Para el primer caso se desarrolla un experimento en el cual se compara la facilidad de orientación en una escena remota virtual y para el segundo caso se propone una prueba para evaluar la sensibilidad de un usuario a la información de fuerza y torque dado por el dispositivo háptico.

7.1.1. Alineación del avatar

El objetivo de esta prueba es evaluar la facilidad de teleoperar tareas teniendo en cuenta la coordinación existente entre los movimientos realizados por el teloperador con su mano en el dispositivo háptico y el efecto que tienen en los movimientos que realiza el robot remoto y que sólo se observan a través del video de realimentación en el monitor.

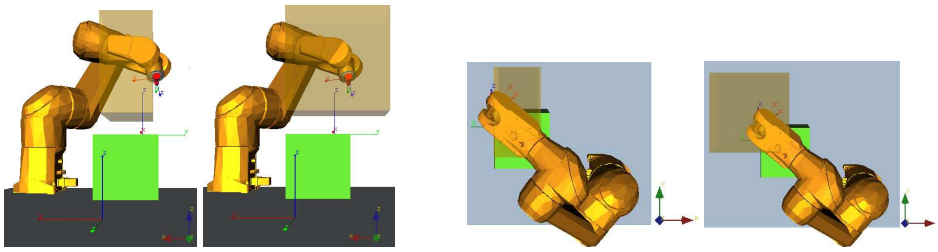
El usuario debe realizar una sencilla tarea de mover una pequeña pieza con el efector final del robot, de una configuración determinada a un lugar marcado con un sistema de referencia sobre una mesa. Este proceso lo lleva a cabo dos veces, uno de ellos con los sistemas de referencia de la base del dispositivo háptico orientado con la misma disposición que el sistema de referencia de la escena. La siguiente vez lo hace con el sistema de referencia alineado con la orientación que tiene la cámara que realimenta el video (Figura 7.1).

A los usuarios sólo se les instruyó en el objetivo que tenia la tarea, y se permitió manipular cuerpos libres en el espacio a aquellos que no habían tenido contacto con un dispositivo háptico con anterioridad. El usuario no podía cambiar las condiciones de la cámara durante la tarea y él mismo indicaba cuando había terminado la tarea. Diez personas con diferentes niveles de acercamiento con los dispositivos hápticos y a los entornos virtuales, realizaron la prueba donde se computó el tiempo de realización de cada una de ellas. Del grupo de 10 personas, dos de ellas eran mujeres y el resto hombres, todos ellos diestros. La mitad realizaron esta prueba iniciando con el problema de la cámara alineada al sistema de referencia de la escena para luego terminar con el otro tipo de orientación, y la otra mitad lo realizaron de con el orden invertido. Dado que el video se realimentará con una



(a) Avatar orientado con la escena. (b) Avatar orientado con la cámara.

Figura 7.1: Se puede apreciar las dos opciones que tiene el usuario para asociar los movimientos de sus manos: con la escena o con la cámara.



(a) Cámara lateral alineada con el eje Y de la escena.

(b) Cámara superior alineada con el eje Z de la escena.

Figura 7.2: Imágenes de los puntos de vista de las otras dos cámaras utilizadas.

cámara a la vez y para eliminar la dependencia del punto de vista de la cámara con la ejecución de la tarea (la sensación de profundidad es limitada) se realiza este mismo proceso con otras dos cámaras esta vez alineadas con uno de los ejes del sistema de referencia de la escena. Adicionalmente, al finalizar la prueba se solicitó que indicaran cual de estos sistemas les había parecido más fácil para realizar la tarea.

De los diez, nueve encontraron más cómoda la teleoperación cuando los movimientos se alineaban con los observados esto es, cuando el espacio de trabajo del dispositivo háptico se alinea con la cámara. De los tiempos mostrados en la Tabla 7.1, se concluye que además de influir de forma muy positiva la adecuada selección del punto de vista de la tarea que se está ejecutando, la coordinación de los movimientos de la mano con los obser-

Cámara	Alineación			
	Escena		Cámara	
	$\bar{t}$	$\sigma(t)$	$\bar{t}$	$\sigma(t)$
Inclinada	25.9	8.8	18.9	8.8
Lateral	87.8	20.9	26.5	2.4
Superior	98.9	40.7	41.9	9.8

Tabla 7.1: Datos obtenidos en la prueba de alineación de los movimientos (en seg.).

		Sentido		
		Fuerza	Torque	Combinación
Aplicado	Fuerza	100	0	0
	Torque	58.3	41.7	0
	Combinación	63.2	5.3	31.6

Tabla 7.2: Matriz de confusión de la sensación de la fuerza realimentada (porcentajes).

vados reduce el tiempo de ejecución en al menos un 27 % que se obtiene con la cámara que mejor capta la perspectiva del problema (Inclinada).

7.1.2. Sensibilidad del operador

Con el objetivo de ofrecer una clara y unívoca información a través de la realimentación de fuerzas al operador, esto es, sin crear confusiones o equivocadas interpretaciones en el tipo de respuesta que debe dar a los estímulos aplicados por el sistema, se diseñó un experimento en el cual el operador es expuesto a una serie de estímulos de fuerza y/o torque. Estos estímulos de diferentes magnitudes dentro del rango de operación del dispositivo háptico, se aplican en una secuencia aleatoria y el operador debe identificar su naturaleza, es decir identificar si es un torque, una fuerza o una combinación de ellos. Antes de iniciar la prueba, se le aplica un estímulo de cada tipo al operador y se le informa su naturaleza. Esta prueba la realizaron diez individuos hombres diestros. La evaluación se hizo tomando el dispositivo con la mano derecha.

De la Tabla 7.2 se puede observar, que para el dispositivo háptico Premium con el que se realizó la prueba, la realimentación de torque no resulta fácil de diferenciar, excepto para los torques aplicados en el eje de giro

del apuntador que sostiene el operador en su mano a manera de lápiz y que en su mayoría corresponden a los torques bien identificados. Este es un problema constructivo de estos dispositivos, ya que los otros dos torques no se aplican en el punto de contacto sino en el extremo del *stylus*, lo que transforma el torque en una fuerza sobre la mano del operador. Estos resultados están basados en una muestra bastante reducida como para generalizarlos, pero van en concordancia con lo expuesto por Verner y Okamura (2009), donde los autores hacen un estudio sobre la realimentación de fueza–torque en comparación con la realimentación de sólo fuerza.

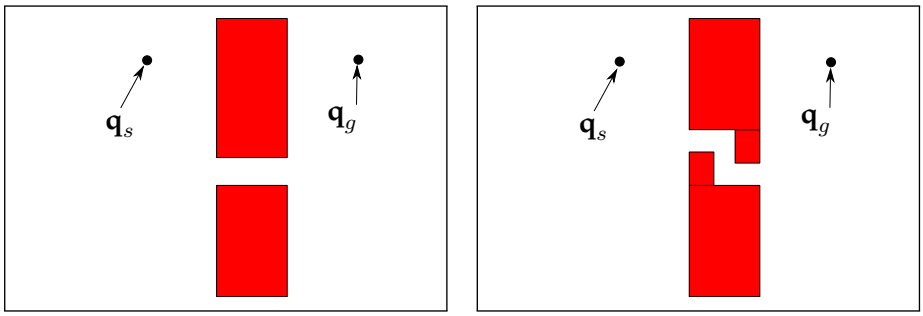
7.2. Planificación

El algoritmo de planificación del Capítulo 4, ha sido implementado tanto en Matlab® como en C++. Todos los grafos utilizados en esta implementación están basados en la librería **Boost Graph** y en su variante **MatlabBGL** de Gleich (2008), y la búsqueda del camino más corto se realiza con el Algoritmo A*. Se han diseñado varios ejemplos que ilustran las principales bondades del enfoque propuesto y se detallan en las siguientes secciones.

7.2.1. Búsqueda del primer camino válido

Dentro del área de la planificación de movimientos existen un conjunto de problemas que son considerados difíciles debido a la complejidad de ubicar muestras que permitan dar una solución rápida. Los problemas de pasillos estrechos y con forma de S, son dos de los más utilizados para probar las bondades de los métodos de planficación y usualmente han sido resueltos usando muestreo sesgado hacia las regiones de $\mathcal{C}$ que revisten especial interés, como los planteados con el muestreo Gausiano (*Gaussian Sampling*) o en el método de prueba de puente (*Bridge Test*).

Cuando el algoritmo que se presenta en este trabajo se usa sin un conocimiento previo del espacio de configuración (ver Sección 4.5), busca un camino libre de colisión $\mathbf{P}^q$ con un comportamiento iterativo, y evaluando de forma tardía la completa validez del camino. Aquí se muestra la evolución que presenta el algoritmo para solucionar el problema del pasillo estrecho y el del camino en S de la Figura 7.3.



(a) Canal estrecho. (b) Canal en S.

Figura 7.3: Problemas propuestos para ilustrar el proceso iterativo que sigue el planificador durante la construcción del conocimiento del espacio de configuraciones.

	Filtrado	*PRM* básico
Muestras generadas	115.3	189.6
Muestras conectadas	115.3	153.8
Validaciones (distancia/colisión)	480.7	615.2

Tabla 7.3: Comparativa con un planificador básico de tipo *PRM*. Las dos primeras filas corresponden al número de muestras y la última al número de validaciones de distancia realizadas. Estas validaciones de distancia se asimilan a las de colisión realizadas por el PRM básico.

En las Figuras 7.4 y 7.5 se muestran algunas imágenes que ilustran la evolución de G_T, G_W y el camino candidato $\mathbf{P}_i^q$ durante la ejecución del ciclo *while* del algoritmo FindPath (Algoritmo 5) aplicado a los problemas planteados. La imagen final corresponde al camino válido $\mathbf{P}_d^q$ final.

El problema que reviste mayor complejidad de estos dos, es el del canal en forma de S. Con el objetivo evaluar su desempeño se ha comparado con un planificador básico *PRM* que utiliza como generador de muestras la secuencia Halton (mismo generador que usa la propuesta de este trabajo) y se han obtenido los siguientes resultados que se detallan en la Tabla 7.3.

Estos datos muestran que para el caso de los obstáculos estáticos el algoritmo se desempeña muy bien ofreciendo resultados comparables y mejorados con respecto a la versión básica del *PRM*.

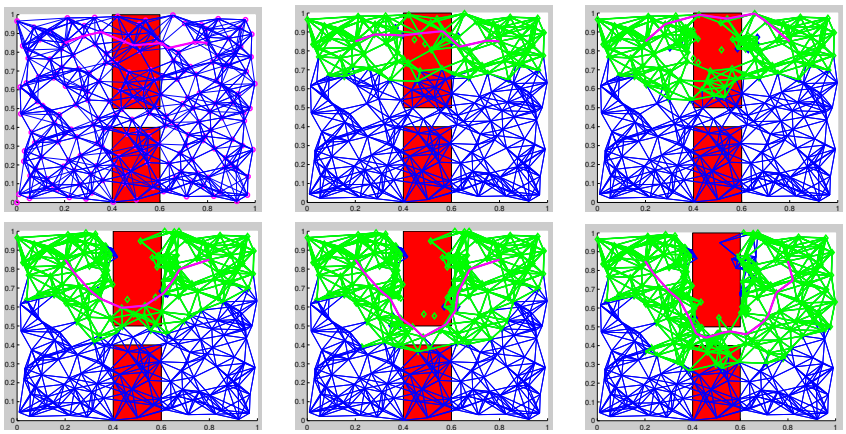


Figura 7.4: De izquierda a derecha y de arriba a abajo se ven las capturas de pasos sucesivos (no consecutivos) del algoritmo FindPath aplicado al problema de pasillo estrecho. El grafo G_T aparece en azul y el subgrafo G_W en verde.

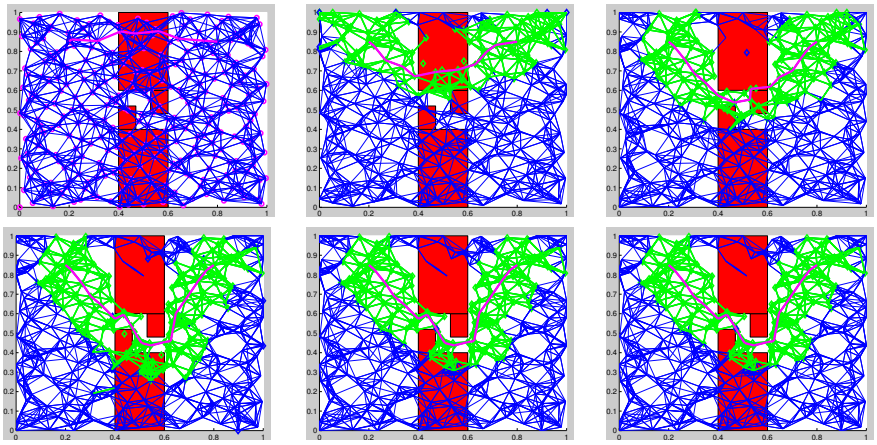


Figura 7.5: Capturas de pasos sucesivos (no consecutivos) del algoritmo FindPath aplicado al problema de canal en forma de S. El grafo G_T aparece en azul y el subgrafo G_W en verde.

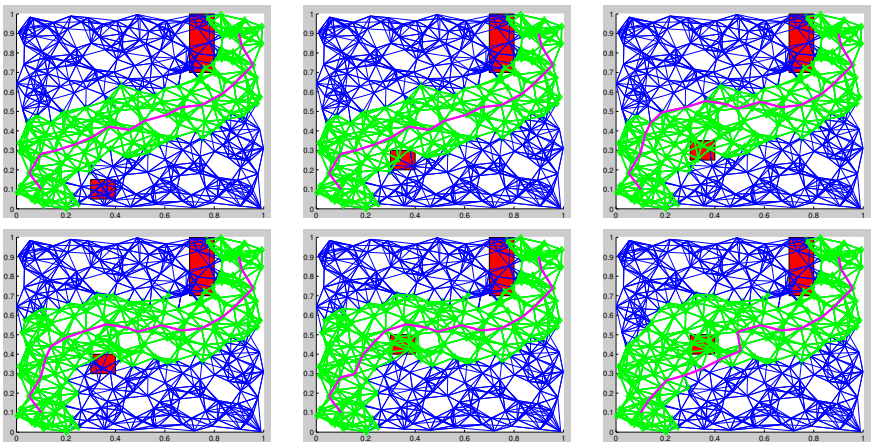


Figura 7.6: Planificación en entornos dinámicos: el objeto móvil (rectángulo) se mueve hacia arriba mientras el planificador se mantiene evaluando el valor H de los vértices del camino. Cuando un cambio es detectado, una nueva búsqueda es realizada, dando la posibilidad de encontrar un nuevo camino que evite las colisiones con él.

7.2.2. (Re)Planificación en presencia de obstáculos móviles

El algoritmo de planificación confiere mayor relevancia a la información relativa a los obstáculos dinámicos cuando se realiza una replanificación, es decir cuando se tiene un $\mathbf{P}_d^q$ obtenido de una iteración previa. En el ejemplo siguiente, se puede observar la capacidad del planificador propuesto para recalcular el camino libre de colisiones mientras se mueven los obstáculos presentes en el espacio. Para ello, se obtiene un $\mathbf{P}_d^q$ inicial y luego se deja libre el obstáculo móvil para que se desplace por la escena, mientras el planificador continua en su tarea de validación del camino.

La Figura 7.6 muestra varias imágenes que ilustran la evolución de G_T, de G_W y del camino solución $\mathbf{P}_d^q$ durante la ejecución del ciclo *while* del algoritmo FindPath, que consecutivamente replanifica el camino dada la presencia del obstáculo rectangular que se mueve verticalmente de abajo hacia arriba. La primera imagen cuenta con un camino inicial válido $\mathbf{P}_d^q$ que es recalculado para evitar colisiones.

El ejemplo muestra cómo el algoritmo es capaz de preservar la mayor cantidad posible del camino inicial, que es conseguido con la reducción del costo de las aristas del último camino válido (ver Apartado 4.3.4). Además, se pueden observar las muestras adicionales generadas cerca de los

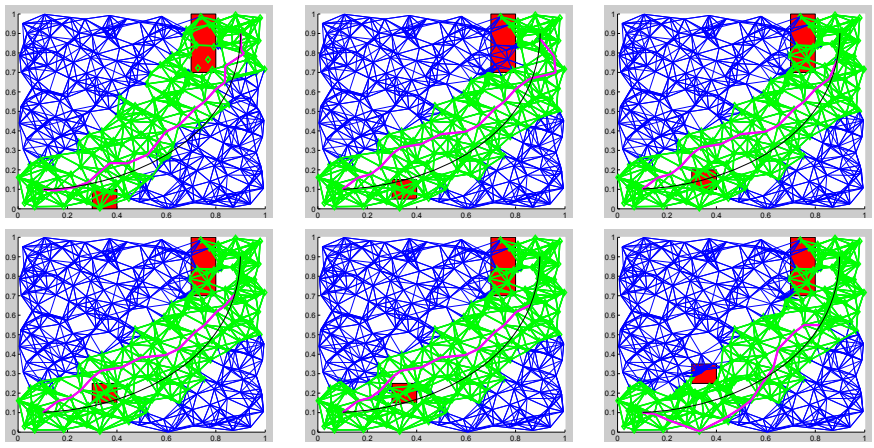


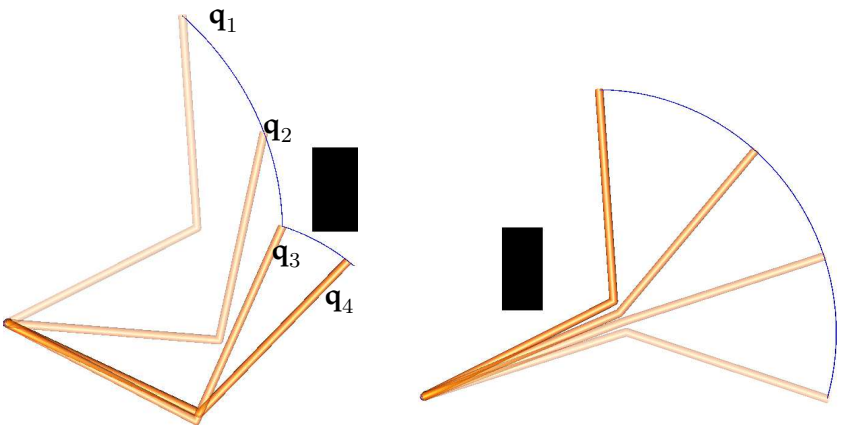
Figura 7.7: Replanificación por cambio de $\mathbf{q}_s$: Al mismo tiempo que el obstáculo se mueve hacia arriba, $\mathbf{q}_s$ se está moviendo hacia abajo siguiendo una trayectoria circular arbitraria. El nuevo camino encontrado mantiene la mayor cantidad posible del trazado original del camino $\mathbf{P}_d^q$ proveído.

obstáculos móviles. Estas muestras han aparecido porque eventualmente el camino candidato $\mathbf{P}_i^q$ cruza sobre el obstáculo y los procesos de validación de las aristas generan estas muestras extras. En contraste con el caso de los obstáculos estáticos, los nodos que colisionan con los obstáculos móviles no son removidos del grafo G_T ya que la oclusión se presentará por un tiempo finito.

7.2.3. Replanificación por cambio de $\mathbf{q}_s$

Un rasgo de este algoritmo de planificación que es de gran importancia para cumplir con los objetivos de este trabajo, es su caracteristica de ejecutarse de forma paralela a la teleoperación. La forma de incluir este comportamiento dentro del planificador, consiste en brindar la facilidad de modificar la configuración inicial manteniendo como resultado la mayor similitud con el camino previo.

En el siguiente ejemplo se muestran los resultados del procedimiento de replanificación debido al cambio de configuración inicial $\mathbf{q}_s$. En las imágenes de la Figura 7.7, la configuración inicial de la esquina superior–derecha sigue una trayectoria de arco circular hacia la configuración objetivo ($\mathbf{q}_g$), localizada en la esquina inferior–izquierda, y cuyo centro se ubica en la esquina superior–izquierda. Esto quiere decir que la configuración inicial



(a) Sin cambio de configuración (b) Con cambio de configuración.

Figura 7.8: Dos ejemplos del problema 2D con robot RR donde la tarea a ser teleoperada requiere o no un cambio de configuración para evitar la colisión con el obstáculo negro.

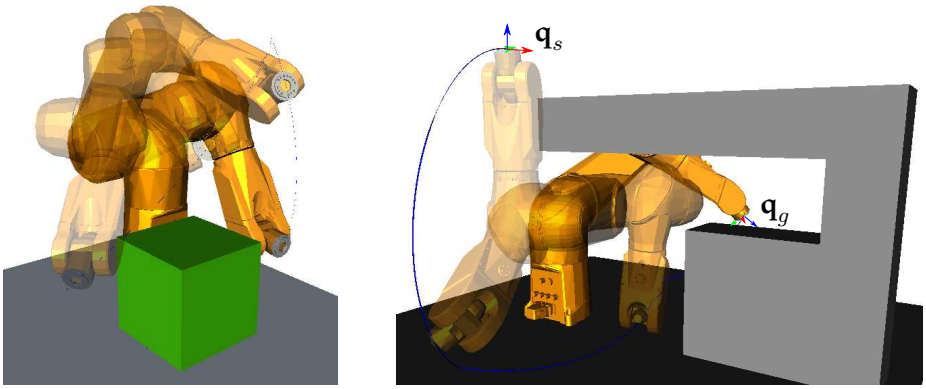
no sigue exactamente el camino propuesto inicialmente en la primera imagen, pero si de forma aproximada. De forma que si el robot se mueve, el algoritmo modifica el camino propuesto para ofrecer siempre una solución desde la posición actual. El obstáculo móvil de geometría rectangular del ejemplo anterior, está presente aquí también. Se puede observar cómo el camino propuesto comparte la mayoría de los vértices entre las diversas iteraciones del algoritmo, de forma que sólo cambia cuando es necesario.

7.3. Guiado háptico

7.3.1. Asistencia a la teleoperación

La asistencia de guiado, como se ha visto en el Capítulo 5, está basado en un camino generado con el planificador del Capítulo 4 que se ha validado en el apartado anterior. Las fuerzas de guiado que se generan a partir de allí sirven para atraer al operador al camino y si él lo desea en dirección a la configuración objetivo.

Con el fin de probar el desempeño del sistema de guiado, se han preparado dos ejemplos que ilustran de forma sencilla la utilidad de la ayuda propuesta para seguir un camino planificado, tanto cuando contiene cambios de configuración como cuando no. Estos ejemplos se llevan a cabo en dos escenarios distintos con diferente grado de dificultad, uno en 2D y el



(a) Sin cambio de configuración (b) Reproducción de la Figura 2.2

Figura 7.9: a)Algunas configuraciones siguiendo un camino sencillo que va a ser guiado y que no requiere ningún cambio de configuración. b) Este camino requiere un cambio entre codo arriba y codo abajo.

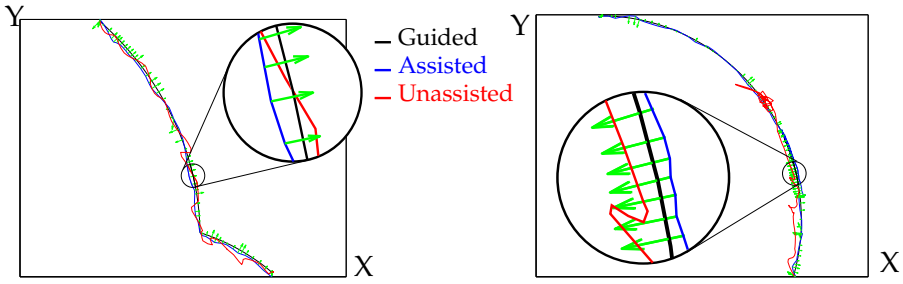
Task	ϵ_{t1}	ϵ_{t2}	ϵ_{t3}
2D RR	1	5.5	10
3D 6 *gdl*	4.78	26.29	47.8

Tabla 7.4: Valores de los umbrales utilizados en la evaluación (mm).

otro en 3D. El primer escenario involucra a un robot RR planar de 2 eslabones (Figura 7.8), y el segundo un robot industrial antropomórfico Stäubli TX90 de 6 *gdl* mostrado en las Figuras 7.9b y 7.9a.

Todas las tareas han sido teleoperadas primero sin hacer uso de la asistencia háptica de guiado y luego con ella. Esta ayuda de guiado ha sido restringida a realimentar únicamente fuerzas, es decir que los torques están desactivados (ver Apartado 7.1.2). Los valores de los umbrales utilizados en la definición de las fuerzas (ver Ec. (5.1)) se muestran en la Tabla 7.4.

La Figura 7.10a muestra los datos adquiridos mientras se teleoperaba la tarea de la Figura 7.8a. Se puede apreciar que el camino seguido con la ayuda de guiado es más suave y como se detalla en la Tabla 7.5, también se ejecuta de forma más rápida. Esta figura también muestra con flechas verdes cómo las fuerzas generadas conducen al operador a seguir el camino deseado. La Figura 7.11 ilustra para este ejemplo el valor calculado de B_r utilizando $B_r^{max} = 100$, $B_r^{min} = 10$, $d_{th} = 2$ y $d_{cov} = 50$, que corresponde a un aumento de la viscosidad proporcional a la distancia del obstáculo (ver Aparatado 5.2.3).



(a) Ejecución de la tarea de la Figura 7.8a (b) Ejecución de la tarea de la Figura 7.8b

Figura 7.10: Evaluación de la ayuda para el ejemplo 2D: Caminos ejecutado y guiado en el espacio cartesiano, con y sin la ayuda de guiado para el ejemplo de la Figura 7.8. Las fuerzas de guiado son mostradas con flechas verdes.

Task	Cambio de Conf.	Guiado		% Reducción.
		Sin	Con	
2D RR	Sin	19.1	13.9	27.2
	Con	–	39.1	–
3D 6 *gdl*	Sin	70.6	59.2	16.15
	Con	–	93.8	–

Tabla 7.5: Tiempos promedio para completar la tarea (para 5 operadores).

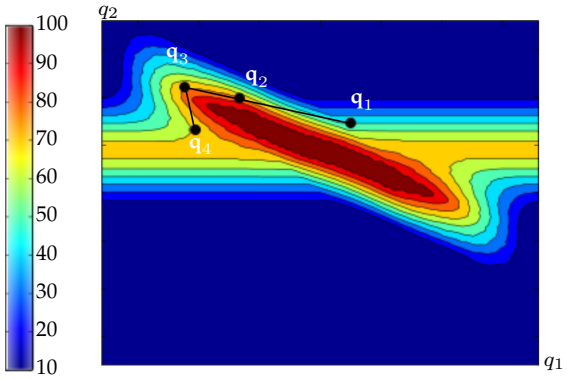
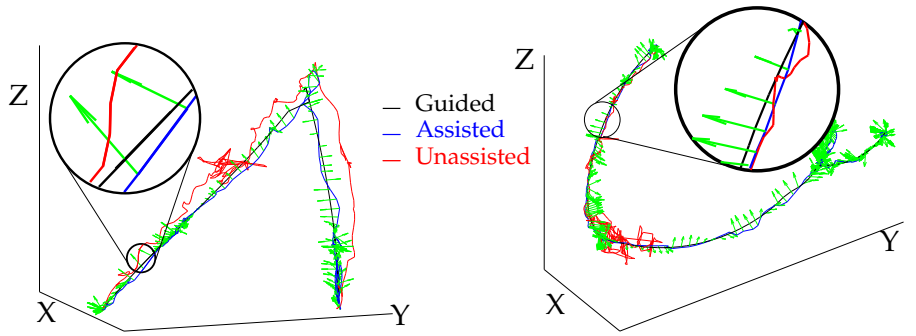


Figura 7.11: Valores de B_r calculados a partir de la Ec. (5.6) sobre el espacio de configuraciones correspondiente a la tarea mostrada en la Figura 7.8a. El camino guiado es mostrado en negro.

La Figura 7.10b muestra los datos adquiridos mientras se teleopera la tarea de la Figura 7.8b. En este caso la tarea no se puede completar sin la ayuda del guiado ya que se requiere un cambio de configuración cinemática. Haciendo uso del guiado la tareas es fácilmente terminada, es decir que



(a) Ejecución de la tarea de la Figura 7.9a (b) Ejecución de la tarea de la Figura 7.9b

Figura 7.12: Evaluación de la ayuda para el ejemplo 3D: Caminos guiado y ejecutado en el espacio Cartesiano, con y sin la ayuda de guiado, para el ejemplo de las Figuras 7.9b y 7.9a. Las fuerzas de guiado son mostradas como flechas verdes.

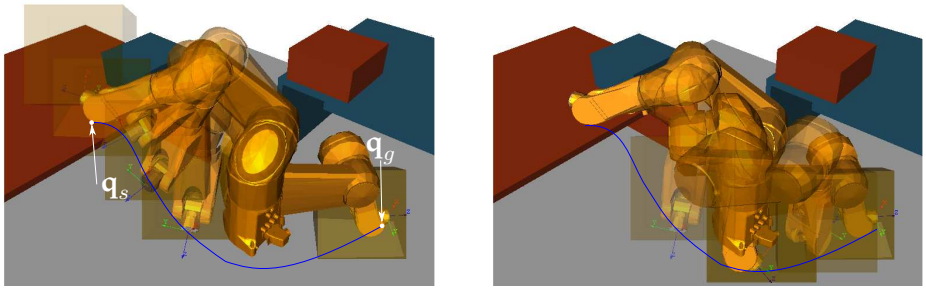
el operador es capaz de cambiar de configuración del robot simplemente pasando cerca de este punto crítico ya que el sistema de ayuda de cruce que puntos críticos el brinda el soporte para atravesarlo correctamente.

El problema 3D con el robot Staübli TX90 es mas difícil de comandar, porque el operador debe manejar correctamente la orientación del TCP, lo cual es difícil y necesita un poco de entrenamiento previo. Además, la limitada percepción de profundidad obstaculiza la teleoperación, como ha ocurrido en la tarea de la Figura7.9a, donde no se ha usado el sistema de guía como se ilustra en la Figura 7.12a. Esta tarea también se ilustra con el robot real en la Figura 7.16.

La Figura 7.12 muestra los datos adquiridos mientras se teleoperaba la tarea de la Figura 7.9b. En este caso la tarea no puede ser completada sin la ayuda de guiado porque es necesario realizar un cambio de configuración cinemática. En estos dos últimos ejemplos, se puede ver que usando la ayuda de guiado las tareas pueden ser terminadas de forma suave y más rápida, como se observa en la Tabla 7.5.

7.3.2. Asistencia a la resincronización

La ayuda de resincronización se aprecia cuando se deben realizar varios movimientos de sincronización para cubrir completamente un camino a teleoperar como se aprecia en la Figura 7.13. El experimento realizado para verificar el desempeño de los operadors con esta ayuda, consiste en una



(a) Fase inicial. (b) Fase final.

Figura 7.13: Evaluación de la resincronización: tarea utilizada para evaluar la ayuda a la resincronización donde se pueden observar varios de estos momentos. Dadas las dimensiones del espacio de trabajo avatar son necesarias varias resincronizaciones para cubrir todo el camino que se está siguiendo.

Ejecución		$\bar{x}$	σ
Sin ayuda	Tiempo	135.1	9.8
	Núm Resin.	18	1.6
Con ayuda	Tiempo	101.5	4.7
	Núm Resin.	11.8	1.0

Tabla 7.6: Tiempos (seg.)y número de resincronizaciones realizadas durante la prueba.

simple tarea de movimiento del efector final del robot Staübli TX90. En la Figura 7.13 se muestra la configuración inicial de la tarea y en la que el operador debe mover el efector final del robot hasta donde marca el final del camino que se marca en azul.

El operador es libre de cambiar de cámara, resincronizar el dispositivo háptico y el robot en cualquier momento, y nunca es forzado a seguir el camino propuesto durante la prueba. En esta prueba han colaborado seis operadors y cada uno de ellos ha realizado la prueba dos veces, una de ellas con la ayuda a la resincronización y la otra sin ella. La mitad de ellos han realizado primero la prueba sin el sistema de ayuda para después realizarla con la ayuda y la otra mitad lo han echo al revés. Se han registrado el tiempo de ejecución y el número de veces que se ha realizado una resincronización (Tabla 7.6). El uso de la ayuda a la resincronización ha servido para reducir no sólo el tiempo de ejecución ($20 - 35\,\%$ de reducción) sino el número de veces que es necesario realizarlas para realizar completamente

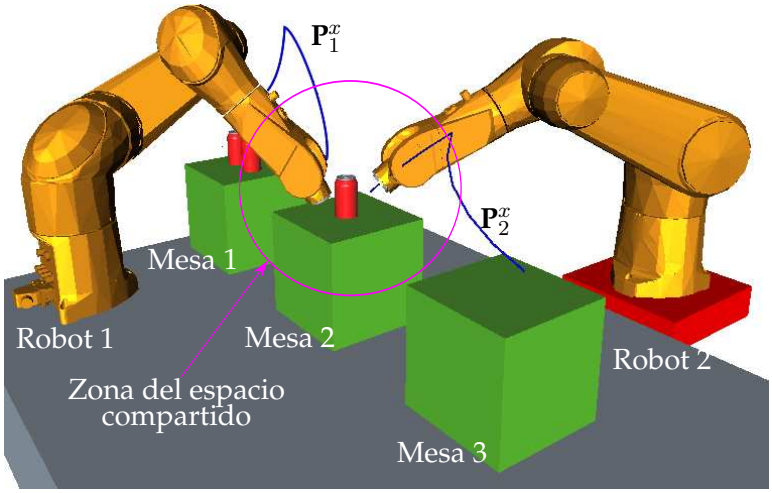


Figura 7.14: Celda remota del IOC donde dos robots comparten parte de su espacio de trabajo y operan de forma conjunta para complir el objetivo global.

la tarea (más de un $25\,\%$ de reducción). Resultados similares se han obtenido con otras tareas probadas.

7.3.3. Teleoperación multirobot

Para la validación de la teleoperación multirobot–multioperador se ha realizado un experimento que involucra dos nodos–mando y un nodo–operación con dos robots Staübli TX90 en modo simulación. Un computador central ubicado físicamente en la celda remota (ver Apartado 6.4.4), centraliza y distribuye la información relativa a las configuraciones comandadas por cada uno de los operadores que tienen como realimentación la simulación de la celda y en la cual se pueden apreciar los movimientos de los robots tal y como están siendo comandados. La tarea consiste en desplazar los envases cilíndricos desde la mesa 1 y hasta la mesa 3. Dada la distancia de separación entre las mesas, ninguno de los dos robots puede realizar la tarea independientemente de forma que deben colaborar. Un robot desplazará los envases de la mesa 1 a la mesa 2 y de allí los tomará el segundo robot para llevarlas hasta la mesa 3 como se observa de la Figura 7.14. En este caso no se considera activo el grado de libertad del carril que sostiene al robot 2. La entrada de los planificadores serán las configuraciónes que permitan a los robots ubicarse en el centro de las respectivas mesas.

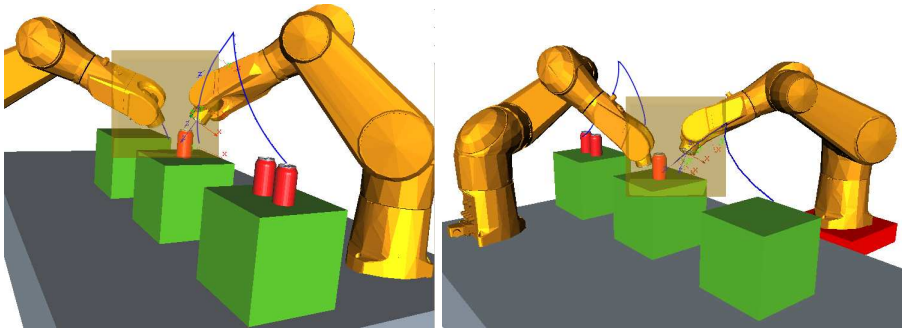


Figura 7.15: Imágenes realimentadas a cada uno de los operadores. En ellas se pueden observar los espacios de trabajo proyectado o avatar dentro de los que se mueven los TCP de los robots de acuerdo a los movimientos de cada uno de los dispositivos hápticos y el vector de fuerza que los atrae al camino.

La zona compartida de la teleoperación se encuentra alrededor de la mesa 2 cuando los dos operadores, siguiendo sus respectivos planes, acceden a esta zona de forma concurrente.

En la Figura 7.15 se pueden ver las imágenes de realimentación que observa cada uno de los teleoperadores durante el desarrollo de la tarea.

7.3.4. Entornos reales

Aunque la interacción directa con la celda remota real excede los alcances de este trabajo, se ha probado con éxito la generación de fuerzas de guiado durante la teleoperación de un robot. En la Figura 7.16 se puede observar la ejecución de la tarea de la Figura 7.9a, tanto en el simulador como la imagen realimentada desde la celda.

7.4. Conclusiones

En este capítulo se han presentado un resumen de los experimentos que se han realizado para la validación de los métodos y los algoritmos propuestos a lo largo de este trabajo. El método usado para definir la correspondencia entre los espacios de trabajo Cartesiano del dispositivo háptico y del robot, unido a la correlación con los movimientos dados por la coordinación mano–ojo son una ayuda útil frente al cambio de punto de vista durante la ejecución de la tarea teleoperada. El algoritmo de planificación

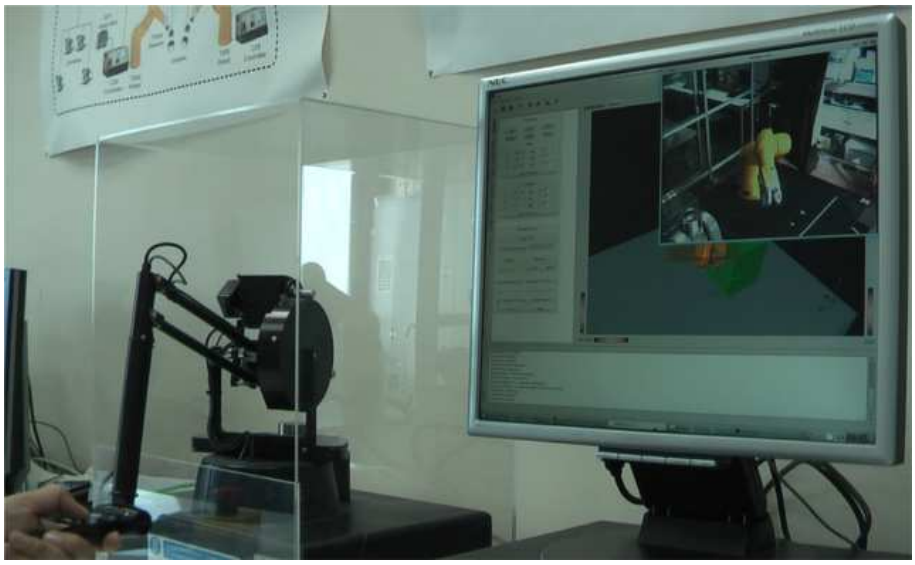
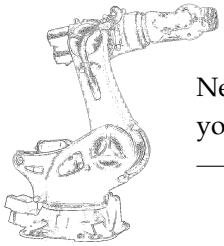


Figura 7.16: Ejecución de la tarea de la Figura 7.9a con el robot real y con realimentación directa de video. La ventana de video no se puede anclar dentro del simulador por lo que ambas se superponen cuando son ampliadas.

de caminos brinda la solución a la tarea y mantiene la evaluación de ésta durante la teleoperación para reaccionar frente a los cambios del entorno y a las necesidades del operador. Juntos son la base del sistema de generación de fuerzas que a través del dispositivo háptico entregan la información de soporte al operador y que le permite mejorar la forma de ejecución de la teleoperación.

8

Conclusiones Generales



Never memorize something that you can look up.

Albert Einstein

La ejecución de tareas teleoperadas con robots puede ser mejorada con sistemas de guiado de forma que el usuario reciba la ayuda necesaria para incrementar no sólo la velocidad sino la seguridad con la que las realiza. Este sistema asume que la tarea a ser teleoperada es conocida, así como el entorno de trabajo donde se desempeña el robot, que es una asunción común y razonable para muchas de las tareas teleoperadas. Desarrollado sobre un sistema de teleoperación bilateral donde un robot industrial es comandado por un dispositivo háptico de escritorio, el sistema de guiado propuesto se basa en la generación de fuerzas que atraen al usuario a un camino libre de colisiones previamente planificado y lo empujan a lo largo de él. Estas fuerzas de guiado son ligeras, de forma que el usuario no queda restringido a seguir exactamente el camino planificado y además en cualquier momento él puede solicitar que este camino sea recalculado desde la configuración en la que se encuentre hasta la configuración objetivo.

8.1. Resumen

Se ha propuesto un marco de trabajo para teleoperación de robots industriales antropomórficos con un novedoso modelo de correspondencia entre el espacio de trabajo del dispositivo háptico con el que va a ser co-

mandado y el del robot. Esta correspondencia tiene en cuenta la posición y orientación de la cámara activa que proporciona el video desde el nodo remoto. Este sistema cuenta con elementos de ayuda al operador en forma de guiado basado en técnicas de planificación de caminos (*path planning*) que facilitan la labor al teleoperador.

Las principales características de este sistema son:

1. Proporciona una correlación sencilla e intuitiva entre los movimientos aplicados al dispositivo háptico en el nodo–mando y los movimientos realizados por el robot y que son visualizados a través de las imágenes de video que transmite la cámara activa. Esto lo consigue al relacionar el dispositivo háptico con el sistema coordenado de la cámara en lugar del sistema coordenado de la base del robot o con el global de la escena.

2. Durante la teleoperación, el operador puede cambiar de cámara activa para incrementar la visibilidad o cambiar las escalas que relacionan los espacios de trabajo para ajustar la precisión.

3. Durante la teleoperación, el operador experimenta ligeras fuerzas que lo atraen hacia el camino y, si el operador lo solicita, también lo empujan a lo largo del mismo hacia la configuración objetivo.

4. Cuando el operador lo solicita, el sistema calcula la nueva posición del espacio de trabajo virtual del dispositivo háptico proyectado dentro de la escena, desde donde pueda retomar y efectuar la teleoperación tanto tiempo como sea posible y lo guía hasta esta posición y orientación a través de la realimentación de fuerza que el dispositivo provee.

5. Proporciona ayudas al teleoperador cuando la tarea incluye cambios de configuración o se debe asegurar el paso del robot por algún punto crítico durante su ejecución, que de otra forma no se podrían asegurar dado que el terreno de la teleoperación es el espacio cartesiano.

Desde el punto de vista de la planificación de movimientos, se incorpora un novedoso algoritmo que es capaz de encontrar caminos realizables tanto entre obstáculos estáticos como entre móviles o la mezcla de unos y otros.

Las principales características del planificador propuesto en el presente trabajo son:

1. Está basado en un *PRM* con evaluación tardía de las colisiones del camino encontrado y el proceso de validación lo realiza en dos etapas.

2. La estrategia utilizada, se basa en la construcción inicial de un gran grafo no evaluado cuyos nodos son muestras generadas con una secuencia de baja discrepancia y la posterior obtención de un subgrafo donde la planificación es realmente llevada a cabo y que está centrado en el camino anteriormente planificado.

3. Cada camino planificado es evaluado y progresivamente mejorado en las subsiguientes iteraciones.

4. El proceso de evaluación resulta, cuando la validación de alguna arista falla, en un nuevo conjunto de muestras cerca de los obstáculos y que resulta muy útil para las siguientes iteraciones.

5. Este procedimiento es directamente utilizado para replanificar cuando están presentes obstáculos móviles, o cuando la configuración inicial cambia y el resultado de esta replanificación siempre es muy cercano al anterior camino válido.

Aunque aquí se ha utilizado el Algoritmo A* para resolver las búsquedas del camino más corto, otros algoritmos de búsqueda en grafos también pueden ser utilizados. Algún algoritmo de búsqueda dinámica, como el *Dynamic A** o el D*, pueden ser incorporados en este mismo marco mejorando la efectividad de las búsquedas. Además, se puede cambiar la definición de la función H para incluir otras características de la teleoperación como la manipulabilidad, adicionalmente a la holgura a los obstáculos.

Siguiendo el camino sugerido, el operador es liberado de prestar atención a las potenciales colisiones de cualquier parte del robot con el entorno, porque las técnicas de planificacion de trayectorias utilizadas garantizan que todo el robot se mueve por un camino libre de colisión, y no solamente el efector final del robot (TCP) como se hace en otros enfoques.

El esquema de control proporcional más amortiguamiento (**P+d**), en el que se basa el sistema de teleoperación bilateral, garantiza la estabilidad a

pesar de los retardos en el canal de comunicaciones si sus ganancias satisfacen una condición dada. Adicionalmente, se hace uso del factor de amortiguamiento del nodo–operación como un factor de seguridad frente a potenciales colisiones, es decir que la teleoperación se ralentiza con el incremento de este factor (a partir de un valor mínimo que satisface el criterio de estabilidad) cuando se advierte una potencial colisión contra un obstáculo estático o dinámico. Este comportamiento reactivo no interfiere con el guiado.

El sistema ha sido validado con varios operadors, tanto en simulación como con el robot real, mostrando que esta asistencia brindada efectivamente incrementa la velocidad y la seguridad en la ejecución de la tarea y aligera la carga de la teleoperación.

8.2. Aportaciones

1. Se ha implementado un marco para la teleoperación bilateral entre dispositivos con diferente cinemática y con espacios de trabajo de tamaño dispar, realizada en el espacio Cartesiano y usando realimentación visual de diferentes cámaras. Este marco permite una teleoperación intuitiva y fácil, complementada mediante una ayuda háptica para minimizar el número de resincronizaciones requerido para realizar una tarea teleoperada, teniendo en cuenta la transformación entre espacios.

2. Se ha propuesto un algoritmo de planificación para entornos dinámicos adaptado a las necesidades de guiado háptico en tareas de teleoperación en escenarios multioperador–multirobot (adaptación a los cambios de la configuración inicial del robot producidos por las consignas del operador, así como a los cambios de configuración de los otros robots, que son conocidos). La propuesta es computacionalmente eficiente al estar basada en evaluaciones tardías e información de distancia.

3. Se ha propuesto un sistema de guiado háptico para la teleoperación bilateral entre dispositivos con diferente cinemática que, basándose en la planificación de caminos y mediante la realimentación de fuerzas, sugiere los movimientos al teleoperador para la ejecución de la

tarea evitando colisiones del robot con el entorno y con los otros robots, y facilitando los posibles cambios de configuración cinemática del robot que la tarea requiera. El sistema se completa con un comportamiento reactivo, que no interfiere con las fuerzas de guiado, que aumenta el amortiguamiento del algoritmo de control en el nodo–operación para prevenir situaciones de inminente colisión.

4. Se ha desarrollado una herramienta de software para la planificación y la teleoperación asistida en entornos virtuales y reales (particularizada al hardware del IOC). La herramienta es multiplataforma, de código abierto, y sigue los estándares actuales de este tipo de desarrollos. Su uso se ha extendido a tareas docentes de planificación de movimientos y a tareas afines de investigación.

8.3. Publicaciones

A lo largo del desarrollo de este trabajo se han realizado aportes que han sido publicados en medios académicos de reconocimiento ya sean dentro del ámbito Español o Internacional. Algunos de ellos se han presentado en congresos y posteriormente han sido extendidos para su publicación en revistas.

Adicionalmente, se ha desarrollado toda una infraestructura de *software* que permite no sólo teleoperar los robots con el enfoque aquí expuesto, sino que brinda un conjunto de herramientas de planificación y simulación que ha permitido el desarrollo de algoritmos para la planificación de los movimientos de robots que van desde cuerpos libres en el espacio (*free-flying*) hasta robots antropomórficos industriales provistos de manos mecánicas.

8.3.1. Artículos derivados

El contenido del trabajo expuesto en el capítulo de teleoperación (Capítulo 3), ha sido publicado en el siguiente artículo, así:

- A. Pérez y J. Rosell, "An assisted re-synchronization method for robotic teleoperated tasks," *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 886 – 891, 2011.
  En este artículo está plasmado el trabajo sobre la correspondencia en-

tre los espacios de trabajo del dispositivo háptico y el el robot tratado en el Capítulo 3.

Adicionalmente, se están terminando dos artículos basado en el contenido de los capítulos de planificación y en el de las ayudas (Capítulos 4 y 5 respectivamente) con la finalidad de enviarlos próximamente a revisión:

- A. Pérez y J. Rosell, "A lazy-evaluated dynamic local roadmap for path planning between moveable obstacles".
  En este artículo se encuentran las bases del planificador basado en *PRM* que se ha desarrollado en el presente trabajo y que está descrito en el Capítulo 4.

- A. Pérez y J. Rosell, "Haptic Aids for Bilateral Teleoperators".
  En este artículo, se han formalizado las diferentes herramientas que constituyen la ayuda háptica dentro del marco de teleoperación bilateral sobre el que se ha desarrollado el presente trabajo y que está tratado en el Capítulo 5.

De la misma manera, durante el proceso de cimentación de las bases en el área de la planificación de movimientos se ha participado en la publicación de los siguientes artículos:

- J. Rosell, C. Vazquez, y A. Pérez, "C-space decomposition using deterministic sampling and distances," *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems.*, pp. 15–20, Nov. 2 2007.

- J. Rosell, M. Roa, A. Pérez, y F. García, "A General Deterministic Sequence for Sampling d-dimensional Configuration Spaces," *Journal of Intelligent and Robotic Systems*, vol. 50, no. 4, pp. 361 – 373, Dic. 2007.

- J. Rosell, C. Vázquez, A. Pérez, y P. Iñiguez, "Motion Planning for Haptic Guidance," *Journal of Intelligent and Robotic Systems*, vol. 53, no. 3, pp. 223–245, May. 2008.

- A. Pérez, A. Rodríguez, J. Rosell, y L. Basañez, "A constraint-based Probabilistic Roadmap Planner," *40th International Symposium on Robotics, ISR'09*, pp. 297–302, 2009.

- A. Rodríguez, A. Pérez, J. Rosell, y L. Basañez, "Sampling-based path planning for geometrically-constrained objects," *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 2074–2079, 2009.

- J. Rosell, L. Cruz, R. Suárez, y A. Pérez, "Importance sampling based on adaptive principal component analysis," *Proc. of the IEEE International Symposium on Assembly and Manufacturing (ISAM)*, pp. 1–6, 2011.

Los aportes hechos en estos artículos también están incorporados en mayor o menor escala en el desarrollo de este trabajo en el capítulo de planificación (Capítulo 4) y en particular en: *"A General Deterministic Sequence for Sampling d-dimensional Configuration Spaces,"* se describe la secuencia de muestreo $S_d(k)$ utilizada para muestrear de forma uniforme el espacio de configuraciones, de características análogas a la Halton; y de los artículos: *"Sampling-based path planning for geometrically-constrained objects,"* y *"Importance sampling based on adaptive principal component analysis,"* se ha extractado la esencia del sistema de generación de muestras en las inmediaciones de los obstáculos y siguiendo las pautas de las restricciones causadas por la cinemática y la geometría.

Así mismo, ya que el presente trabajo tiene un gran componente de herramientas computacionales que hicieron posible no sólo el desarrollo sino la validación de todos los conceptos presentados (Capítulo 6), también se han realizado aportes en esta línea y sobre todo enfocado al público que llega al área de planificación de movimiento y teleoperación desde ámbitos donde los conceptos de abstracción y programación no son muy profundos, de forma que presenta el desarrollo de los modelos, las herramientas utilizadas y los principios necesarios para hacer frente a la solución de este típo de problemas:

- A. Pérez y J. Rosell, "Planificación de movimientos en robótica: curso práctico para implementar un PRM," *Actas de las XXIX Jornadas de Automática (CEA-IFAC)*, 2008.

- A. Pérez y J. Rosell, "A Roadmap to Robot Motion Planning Software Development," *Computer Applications in Engineering Education*, vol. 18(4), pp. 651–660, Dic. 2010.

Adicionalmente, se tiene preparado un artículo sobre *The Kautham Project*, con la finalidad de difundir sus fortalezas y potencialidades en el área de la planificación y la teleoperación, que se espera enviar próximamente a revisión.

- A. Pérez, J. Rosell, y A. F. Montaño, "The Kautham Project: A robot simulation toolkit for motion planning and teleoperation guiding".

De forma paralela, se han publicado artículos que no están directamente relacionadas con el tema de la investigación, pero que han brindado aportes interesantes para la extensión del sistema de planificación y simulación de robots en especial los de tipo árbol, aplicado principalmente en las tareas de planificación de agarre de objetos con manos antropomórficas y minoritariamente en la planficación para la broncospía virtual:

- R. Suárez, J. Rosell, A. Pérez, y C. Rosales, "Efficient search of obstacle-free paths for anthropomorphic hands," *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and System*, pp. 1773–1778, 2009.
- J. Rosell, R. Suárez, C. Rosales, J. A. García, y A. Pérez, "Motion planning for high DOF anthropomorphic hands," *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 4025–4030, 2009.
- P. Cabras, J. Rosell, A. Pérez, W. G. Aguilar, y A. Rosell, "Haptic-based navigation for the virtual bronchoscopy," *18th World Congress of the International Federation of Automatic Control (IFAC)*, vol. 18(1), pp. 9638–9643, 2011.
- J. Rosell, R. Suárez, A. Pérez, y C. Rosales, "Including virtual constraints in motion planning for anthropomorphic hands," *Proc. of the IEEE International Symposium on Assembly and Manufacturing (ISAM)*, pp. 1–6, 2011.
- J. Rosell, R. Suárez, C. Rosales, y A. Pérez, "Autonomous motion planning of a hand-arm robotic system based on captured human-like hand postures," *Autonomous Robots*,vol. 31(1), pp. 87–102, 2011.

8.3.2. *Software*

Más de 12.000 líneas de código componen el sistema de planificación, simulación y teleoperación denominado *The Kautham Project* desarrollado a lo largo del periódo de ejecución de la presente tesis doctoral. Al inicio de este trabajo, no estaba disponible un entorno de simulación y prueba de algorítmos de planificación utilizable ya que cada grupo de investigación al rededor del mundo utilizaba sus propias aplicaciones y, aunque

algunas veces este código estaba disponible como en el caso de la MSL (http://msl.cs.uiuc.edu/msl) y la MPK (http://robotics.stanford.edu/mitul/mpk), su uso práctico estaba restringido ya fuera por la claridad del código como por la insuficiente documentación que aportaban.

Siendo este el punto de partida, *The Kautham Project* permite simular problemas de planificación en espacios *SE(2)*, *SE(3)*, $\mathbb{R}^n$, *SE(3)*$\times\mathbb{R}^n$ y en espacios multirobot donde cada robot puede tener uno de estos espacios de configuraciones. Ha permitido el desarrollo y prueba de diversos tipos de planificador de caminos tanto para robots *free-flying* como para robots más complejos y se ha extendido para ser usado en el marco de la teloperación de los robots de la celda remota. De esta forma se ha conseguido que en una sóla aplicación y de forma organizada y accesible se haya unificado las labores de implementación de las propuestas teóricas.

Sin embargo, con el correr de los años y teniendo en cuenta las limitaciones propias, tanto personales como organizacionales, han ido apareciendo paulatinamente herramientas que con el mismo horizonte y la misma filosofía han logrado congregar masa crítica que ha permitido su rápida evolución e incorporación en el mundo de la robótica. Es así como han encontrado cabida herramientas como la OMPL (http://ompl.kavrakilab.org), la OpenRAVE (http://openrave.programmingvision.com) y más recientemente y con mayor impacto global las herramientas OROCOS (www.orocos.org) y ROS (www.ros.org).

Con el proyecto *The Kautham Project* se han ido incorporando los sistemas de comunicaciones de ROS y se ha ido modularizando la aplicación para acercarla aún más al paradigma de la programación por componentes, lo que ha permitido extender su utilización en las tareas cotidianas del IOC y la ha convertido en un referente interno para el desarrollo de tareas de planificación, simulación y posteriór ejecución de los resultados obtenidos. Se ha utilizado en el desarrollo de cuatro tesis de master y se prevee extender su utilidad con la incorporación de modelos dinámicos de ODE a través de un proyecto de final de carrera.

Es por estas razones que se ha considerado el proyecto *The Kautham Project* (sir.upc.es/kautham) como uno más de los aportes, aunque no teóricos, del desarrollo de esta tesis.

8.4. Trabajos futuros

Para darle continuidad al trabajo presentado aquí, se proponen cuatro líneas de actuación en el marco de los aportes teóricos:

1. Incluir herramientas de realidad aumentada en el video realimentado desde la escena remota con la finalidad de agregar el espacio de trabajo virtual del dispositivo háptico y mejorar así la percepción de los movimientos por parte del operador.

2. Mejorar los sistemas de sensores que permiten actualizar el modelo del entorno de trabajo de los robots de forma que se puedan tener en cuenta otros objetos móviles adicionalmente a los robots.

3. Extender la ayuda de cruce de puntos críticos con el fin de ser utilizada en las inmediaciones y en las mismas singularidades mecánicas del robot.

4. Estudiar la forma de compartir los planes calculados entre los nodos–mando como forma de prever los movimientos de cada uno de los operadores incrementando así la ventana de tiempo en la que se pueden asumir estáticos todos los obstáculos.

De la misma forma se propone continuar con la modularización del proyecto *The Kautham Project* a la vez que se continue incorporando más herramientas de las provistas tanto en OROCOS como en ROS para establecer el marco de pruebas interno de los algoritmos que se desarrollan en el IOC.

A

Definición del problema

En este apéndice se incluye un ejemplo del archivo XML que describe un problema de planificación dentro del ambiente de *The Kautham Project*. El archivo XML incluye cuatro secciones principales: El robot activo, los otros robots, los obstáculos fijos y los parámetros del planificador y la consulta. Las secciones mínimas son las que denotan el robot activo y los obstáculos fijos o escenario. Aquí se incluye uno como ejemplo.

```
<?xml version="1.0"?>
<Problem name="IOC_Remote_Cell.xml">
        <Robot robot="robots/mobile_TX90.rob" scale="1.0">
                <Limits name="Y" min="0.0" max="2500.0" />
                <Home X="0.0" Y="0.0" Z="0.0" WX="0.0" WY="0.0" WZ="1.0" TH="0.0" />
        </Robot>
        <Robot robot="robots/TX90.rob" scale="1.0">
                <Home X="1484.0" Y="50.0" Z="0.0" WX="0.0" WY="0.0" WZ="1.0" TH="180.0" />
                <InvKinematic name="TX90" />
        </Robot>
        <Scene scene="scenes/IOC.iv" scale="1.0">
                <Location X="0.0" Y="0.0" Z="0.0" WX="0.0" WY="0.0" WZ="1.0" TH="0.0" />
        </Scene>
        <Planner>
                <Parameters>
                        <Name>PRM</Name>
                        <LocalPlanner stepSize="0.05">Linear</LocalPlanner>
                        <Parameter name="Drawn␣Path␣Link">-1</Parameter>
                        <Parameter name="Max.␣Neighs">10</Parameter>
                        <Parameter name="Max.␣Samples">300</Parameter>
                        <Parameter name="Neigh␣Thresshold">1.5</Parameter>
                        <Parameter name="Speed␣Factor">1</Parameter>
                        <Parameter name="Step␣Size">0.05000000075</Parameter>
                </Parameters>
                <Queries>
                        <Query>
                                <Init dim="13">0.454 0.5 0.5 0.5 0.5 0.5 0.5 0.353 //
                                0.61 0.664 0.542 0.407 0.5</Init>
                                <Goal dim="13">0.454 0.5 0.5 0.5 0.5 0.5 0.5 0.264 //
                                0.725 0.722 0.563 0.329 0.5</Goal>
                        </Query>
                </Queries>
        </Planner>
</Problem>
```

B

Definición de un robot

En este apéndice se incluye un ejemplo del archivo XML que describe un robot dentro del ambiente de *The Kautham Project*.

En particular los robots Staübli TX90 tienen la siguiente geometría:

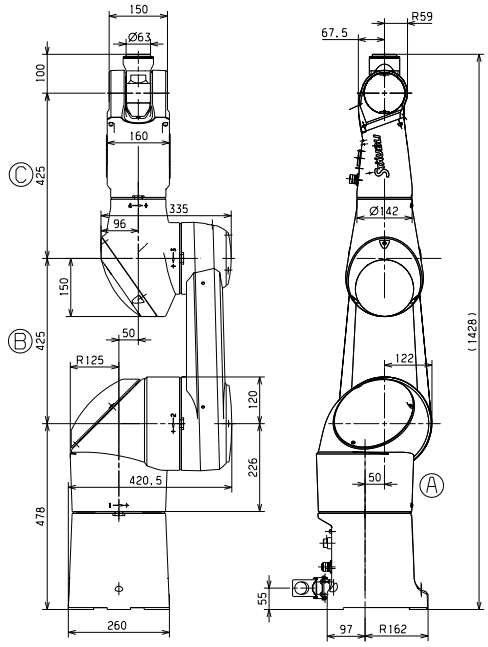


Figura B.1: Dimensiones de un robot Staübli TX90.

Motion Range	Axis 1 (A)	$\pm$ 180
	Axis 2 (B)	+147,5/-130
	Axis 3 (C)	$\pm$ 145
	Axis 4 (D)	$\pm$ 270
	Axis 5 (E)	+140/-115
	Axis 6 (F)	$\pm$ 270

Tabla B.1: Límites de las articulaciones en grados.

A partir de esta información geométrica se puede describir completamente el robot a través de sus parámetros D–H como se aprecia en la Tabla B.1. Teniendo en cuenta la cantidad de grados de libertad que tendrá este robot y de acuerdo a la descripción hecha de los controles en el Capítulo 6, se construye el archivo Rob. En este archivo se describe cómo se construye el robot usando cada uno de los eslabones y en cual orden han de usarse. La única diferencia en la descripción de un robot cadena cinemática abierta y una tipo árbol, es que en esta última un eslabón puede tener más de un hijo.

```
<?xml version="1.0" encoding="UTF-8"?>
<Robot  name="TX90" DHType="Modified" robType="Chain">

  <WeightSE3 rho_r="1.0"></WeightSE3>
  <Joints size="7">
    <Joint name="Base" ivFile="rtx90/base.wrl">
      <DHPars alpha="0.0" a="0.0" theta="0.0" d="0.0"></DHPars>
      <Description rotational="false" movable="false" ></Description>
      <Limits Hi="0" Low="0"></Limits>
      <Weight weight="1.0"></Weight>
      <Parent name=""></Parent>
    </Joint>

    <Joint name="Shoulder" ivFile="rtx90/shoulder.wrl">
      <DHPars alpha="0.0" a="0.0" theta="0.0" d="478.0" ></DHPars>
      <Description rotational="true" movable="true" ></Description>
      <Limits Hi="180.0" Low="-180.0"></Limits>
      <Weight weight="1.0"></Weight>
      <Parent name="Base"></Parent>
    </Joint>

    <Joint name="ForeArm" ivFile="rtx90/forearm.wrl">
      <DHPars alpha="-90.0" a="50.0" theta="-90.0" d="50.0"></DHPars>
      <Description rotational="true" movable="true" ></Description>
      <Limits Hi="147.5" Low="-130.0"></Limits>
      <Weight weight="1.0"></Weight>
      <Parent name="Shoulder"></Parent>
    </Joint>

    <Joint name="Elbow" ivFile="rtx90/elbow.wrl">
      <DHPars alpha="0.0" a="425.0" theta="90.0" d="0.0"></DHPars>
      <Description rotational="true" movable="true" ></Description>
      <Limits Hi="145.0" Low="-145.0"></Limits>
      <Weight weight="1.0"></Weight>
      <Parent name="ForeArm"></Parent>
    </Joint>

    <Joint name="Arm" ivFile="rtx90/arm.wrl">
      <DHPars alpha="90.0" a="0.0" theta="0.0" d="425.0"></DHPars>
      <Description rotational="true" movable="true" ></Description>
      <Limits Hi="270.0" Low="-270.0"></Limits>
      <Weight weight="1.0"></Weight>
```

```
    <Parent name="Elbow"></Parent>
  </Joint>

  <Joint name="Wrist" ivFile="rtx90/wrist.wrl">
    <DHPars  alpha="-90.0" a="0.0" theta="0.0" d="0.0"></DHPars>
    <Description rotational="true" movable="true"   ></Description>
    <Limits Hi="140.0" Low="-115.0"></Limits>
    <Weight weight="1.0"></Weight>
    <Parent name="Arm"></Parent>
  </Joint>

  <Joint name="Tcp" ivFile="rtx90/tcp.wrl">
    <DHPars  alpha="90.0" a="0.0" theta="0.0" d="100.0"></DHPars>
    <Description rotational="true" movable="true"   ></Description>
    <Limits Hi="270.0" Low="-270.0"></Limits>
    <Weight weight="1.0"></Weight>
    <Parent name="Wrist"></Parent>
  </Joint>
      </Joints>

<ControlSet size="6">
      This control set is thinkig to be as EigenVector collection with their own
      EigenValue used to obtain the respective DOF vector of the robot. All the
      robots need to be positioned in any location in SE3 WorkSpace as well it need
      6 values (X, Y, Z, X1, X2, X3) to reach it. The X1, X2, X3 values will be mapped to
      axis-angle notation (Wx, Wy, Wz, angle) and the other DOF are respectively the
      articular values in Chain or Tree robot types.

      <Offset>
  This vector can be added in order to provide a offset or centre values to adjust
  the control values and the DOF values.

            <DOF name="Shoulder" value="0.5"></DOF>
            <DOF name="ForeArm" value="0.5"></DOF>
            <DOF name="Elbow" value="0.5"></DOF>
            <DOF name="Arm" value="0.5"></DOF>
            <DOF name="Wrist" value="0.5"></DOF>
            <DOF name="Tcp" value="0.5"></DOF>

      </Offset>

      At less, the robot needs to be controled through a 6 values corresponding to a
      SE3 configuration.

      <Control name="Shoulder" eigValue="1.0">
            <DOF name="Shoulder" value="1.0"></DOF>
      </Control>

      <Control name="ForeArm" eigValue="1.0">
            <DOF name="ForeArm" value="1.0"></DOF>
      </Control>

      <Control name="Elbow" eigValue="1.0">
            <DOF name="Elbow" value="1.0"></DOF>
      </Control>

      <Control name="Arm" eigValue="1.0">
            <DOF name="Arm" value="1.0"></DOF>
      </Control>

      <Control name="Wrist" eigValue="1.0">
            <DOF name="Wrist" value="1.0"></DOF>
      </Control>

      <Control name="Tcp" eigValue="1.0">
            <DOF name="Tcp" value="1.0"></DOF>
      </Control>
  </ControlSet>
</Robot>
```

Bibliografía

Agha-mohammadi, A., Chakravorty, S., y Amato, N. M., "Firm: Feedback controller-based information-state roadmap - a framework for motion planning under uncertainty -," en *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2011, pp. 4284–4291.

Akgun, B. y Stilman, M., "Sampling heuristics for optimal motion planning in high dimensions," en *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2011, pp. 2640–2645.

Alterovitz, R., Patil, S., y Derbakova, A., "Rapidly-exploring roadmaps: Weighing exploration vs. refinement in optimal motion planning," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, 2011, pp. 3706–3712.

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., y Wu, A. Y., "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *Journal ACM*, vol. 45, no. 6, pp. 891–923, 1998.

Azorin, J. M., J. M. Sabater, Paya, L., y Garcia, N., "Kinematics correspondence & scaling issues in virtual telerobotics systems," en *Proc. of the World Automation Congress*, 2004, pp. 383 – 388.

Basañez, L., Rosell, J., Palomo, L., Nuño, E., y Portilla, H., "A framework for robotized teleoperated tasks," en *Proc. of ROBOT 2011 Robótica Experimental*, 2011, pp. 573–580.

Basañez, L. y Suárez, R., "Teleoperation," en *Handbook of Automation*, 2009, pp. 449–468.

Bekris, K. E., "Avoiding inevitable collision states: Safety and computational efficiency in replanning with sampling-based algorithms," en *Workshop on "Guaranteeing Safe Navigation in Dynamic Environments", Int. Conf. on Robotics and Automation, ICRA*, Anchorage, AK, May 2010, pp. 1–8.

Belghith, K., Kabanza, F., Hartman, L., y Nkambou, R., "Anytime dynamic path-planning with flexible probabilistic roadmaps," en *Proc. of IEEE Int Conf. on Robotics and Automation, ICRA'06.*, 2006, pp. 2372–2377.

Bernabeu, E. J., "Generation of homotopic paths for a size-changing sphere," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Roma,Italy, 2007, pp. 717–723.

Bohlin, R. y Kavraki, L., "Path planning using lazy PRM," en *Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA*, vol. 1, 2000, pp. 521–528.

Brooks, T. y Ince, I., "Operator vision aids for telerobotic assembly and servicing in space," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, vol. 1, 1992, pp. 886 – 891.

Bruyninckx, H., Soetens, P., y Koninckx, B., "The real-time motion control core of the Orocos project," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, 2003, pp. 2766–2771.

Caccavale, F., Chiacchio, P., De Santis, A., Marino, A., y Villani, L., "An experimental investigation on impedance control for dual-arm cooperative systems," en *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Sept. 2007, pp. 1–6.

Carignan, C. y Olsson, P., "Cooperative control of virtual objects over the Internet using force-reflecting master arms," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, vol. 2, Abril 2004, pp. 1221–1226.

Chotiprayanakul, P. y Liu, D., "Workspace mapping and force control for small haptic device based robot teleoperation," en *Proc. of the IEEE Int. Conf. on Information and Automation, ICRA*, 2009, pp. 1613–1618.

Conti, F. y Khatib, O., "Spanning large workspaces using small haptic devices," en *Proc. of the First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2005, pp. 183 – 188.

Cortés, J., Siméon, T., y Laumond, J.-P., "A random loop generator for planning the motions of closed kinematic chains using prm methods," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, vol. 2, 2002, pp. 2141– 2146.

Costa, R. y Basañez, L., "Planificación de Movimientos y Control de Fuerza en Entornos Multi-Robot," *Revista Iberoamericana de Automática e Informática industrial (RIAI)*, vol. 1, no. 3, pp. 29–42, Octubre 2004.

Şucan, I. A. y Kavraki, L. E., "On the implementation of single-query sampling-based motion planners," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Anchorage, Alaska, 2010, pp. 2005–2011.

Denny, J. y Amato, N. M., "Toggle prm: Simultaneous mapping of c-free and c-obstacle - a study in 2d -," en *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2011, pp. 2632–2639.

Diankov, R., "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, Robotics Institute, August 2010.

Dijkstra, E. W., "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

Diolaiti, N. y Melchiorri, C., "Obstacle avoidance for teleoperated mobile robots by means of haptic feedback," en *Proc. of the IEEE 1st Int. Workshop on Advances in Service Robotics*, 2003, pp. 1–7.

Dominjon, L., Lécuyer, A., Burkhardt, J.-M., y Richir, S., "Development of a telerobotic system for exploration of hazardous environments," en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2004, pp. 2532–2537.

Dominjon, L., Lécuyer, A., Burkhardt, J.-M., Andrade-Barroso, G., y Richir, S., "The "bubble"technique: Interacting with large virtual environments using haptic devices with limited workspace," en *Proc. of the First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2005, pp. 639–640.

Elhajj, I., Xi, N., Fung, W., Liu, Y., Hasegawa, Y., y Fukuda, T., "Modeling and Control of Internet Based Cooperative Teleoperation," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, vol. 1, Mayo 2001, pp. 662–667.

Eric Larsen, M. C. L. Stefan Gottschalk y Manocha, D., "Fast proximity queries with swept sphere volumes," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, 2000, pp. 3719–3726.

Ferguson, D. y Stentz, A., "Anytime, dynamic planning in high-dimensional search spaces," en *Proc. of IEEE Int Conf. on Robotics and Automation, ICRA'07*. IEEE, 2007, pp. 1310–1315.

Folgheraiter, M., Bongardt, B., Albiez, J., y Kirchner, F., "A bio-inspired haptic interface for tele-robotics applications," en *Proc. of the IEEE Int. Conf. on Robotics and Biomimetics*, 2009, pp. 560–565.

Gaztanaga, I. (2011) The Boost Interprocess library. [Online]. Available: http://www.boost.org/doc/libs/1_49_0/doc/html/interprocess.html

Geeks, D. T.; Henrich, "Path planning and execution in fast-changing environments with known and unknown obstacles," en *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'07*, 2007, pp. 21 – 26.

Geraerts, R. y Overmars, M. H., "A comparative study of probabilistic roadmap planners," *Proc. of the Workshop on Algorithmic Foundations of Robotics*, vol. 7, no. 2, pp. 43 – 58, 2002.

Gleich, D., "The matlabBGL: A Matlab Graph Library based on Boost Graph Library. Release 4.0." October 2008. [Online]. Available: http://www.cs.purdue.edu/homes/dgleich/packages/matlab_bgl/

Gottschalk, S., Lin, M. C., y Manocha, D., "OBBTree: A hierarchical structure for rapid interference detection," *Computer Graphics*, vol. 30, pp. 171–180, 1996.

Gropp, W., Lusk, E., y Thakur, R., *Using MPI-2: Advanced Features of the Message-Passing Interface*. MIT Press, 1999.

Halton, J., "On the effciency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," *Numer. Math.*, vol. 2, pp. 84–90, 1960.

Halton, J. H., "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," *Numerische Mathematik*, vol. 2, no. 1, pp. 84 – 90, December 1960.

Hart, P., Nilsson, N., y Raphael, B., "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Trans. on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100 –107, 1968.

Hauser, K. K. y Latombe, J.-C., "Multi-modal motion planning in non-expansive spaces," *Int. J. of Robotics Res.*, vol. 29, no. 7, pp. 897–915, 2010.

Hirai, T., Ikuta, T., y Noborio, H., "A teleoperation system based on generation of artificial forces and sensor-based motion-planning," en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, vol. 2, 2000, pp. 1179 – 1186.

Horan, B. y Nahavandi, S., "Intuitive haptic control surface for mobile robot motion control," en *Proc. of the IEEE Int. Workshop on Safety, Security and Rescue Robotics*, 2008, pp. 121–127.

Hsu, D., Latombe, J.-C., y Kurniawati, H., "On the probabilistic foundations of probabilistic roadmap planning," en *Int. Symp. on Robotics Research*, 2005, pp. 83–97.

Ivanisevic, I. y Lumelsky, V., "Configuration space as a means for augmenting human performance in teleoperation tasks," *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 30, no. 3, pp. 471 – 484, 2000.

Jaillet, L., Hoffman, J., van den Berg, J., Abbeel, P., Porta, J. M., y Goldberg, K. Y., "Eg-rrt: Environment-guided random trees for kinodynamic motion planning with uncertainty and obstacles," en *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2011, pp. 2646–2652.

Jarvis, R., "Terrain-aware path guided mobile robot teleoperation in virtual and real space," en *Proc. of the 3rd Int. Conf. on Advances in Computer-Human Interactions*, 2010, pp. 56 – 65.

Jihong, Y., Yanhe, Z., Jie, Z., y Hegao, C., "Task Planner Design Based on Petri Net for Multi-robot Teleoperation over Internet," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Octubre 2006, pp. 5220–5225.

Jiménez, P., Thomas, F., y Torras, C., "Collision Detection Algorithms for Motion Planning," en *Robot Motion Planning and Control*, Laumond, J.-P., Ed. Springer, 1998, ch. 6, pp. 305 – 342.

Karaman, S. y Frazzoli, E., "Sampling-based algorithms for optimal motion planning," *Int. J. Robotics Res.*, vol. 30, no. 7, pp. 846–894, Jun. 2011.

Kavraki, L. E., Svestka, P., Latombe, J.-C., y Overmars, M., "Probabilistic roadmaps for path planning in high dimensional configuration spaces," *IEEE Trans. on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

Kavraki, L. E., Kolountzakis, M. N., y Latombe, J.-C., "Analysis of probabilistic roadmaps for path planning," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 1, pp. 166 –171, Feb. 1998.

Khatib, O., Yokoi, K., Chang, K., Ruspini, D. C., Holmberg, R., y Casal, A., "Coordination and decentralized cooperation of multiple mobile manipulators," *J. of Robotic Systems*, vol. 13, no. 11, pp. 755–764, 1996.

Kikuchi, J., Takeo, K., y Kosuge, K., "Teleoperation System via Computer Network for Dynamic Environment," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, vol. 4, Mayo 1998, pp. 3534–3539.

Koenig, S. y Likhachev, M., "D*lite," en *Proc. of Eighteenth National Conf. on Artificial Intelligence*. Menlo Park, CA, USA: American Association for Artificial Intelligence, 2002, pp. 476–483.

Kuffner, J. J., "Effective sampling and distance metrics for 3d rigid body path planning," en *Proc. of the IEEE Int. Conf. on Robotics and Automation ICRA*, vol. 4, 2004, pp. 3993–3998.

Kuffner, J. J. y LaValle, S. M., "Space-filling trees: A new perspective on incremental search for motion planning," en *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2011, pp. 2199–2206.

Kumar, V., "50 years of robotics [from the guest editors]," *IEEE Robotics Automation Magazine*, vol. 17, no. 3, p. 8, 2010.

Kunz, T., Reiser, U., Stilman, M., y Verl, A., "Real-time path planning for a robot arm in changing environments," en *Proc. of IEEE/RSJ Int Conf. on Intelligent Robots and Systems, IROS'10.*, 2010, pp. 5906 –5911.

Larsen, E., Gottschalk, S., Lin, M. C., y Manocha, D., "Fast Proximity Queries with Swept Sphere Volumes," Department of Computer Science, University of N. Carolina, Chapel Hill., Tech. Rep. TR99-018, 1999.

LaValle, S. M., *Planning Algorithms*. Cambridge University Press, 2006.

LaValle, S. M., Branicky, M. S., y Lindemann, S. R., "On the Relationship between Classical Grid Search and Probabilistic Roadmaps," *Int. J. of Robotics Res.*, vol. 23, no. 7-8, pp. 673–692, 2004.

Lee, D. y Spong, M., "Passive bilateral teleoperation with constant time delays," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Mayo 2006, pp. 2902–2907.

——, "Bilateral Teleoperation of Multiple Cooperative Robots over Delayed Communication Networks: Theory," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Abril 2005, pp. 360–365.

Lee, D., Martinez-Palafox, O., y Spong, M., "Bilateral Teleoperation of Multiple Cooperative Robots over Delayed Communication Networks: Application," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Abril 2005, pp. 366–371.

——, "Bilateral Teleoperation of a Wheeled Mobile Robot Over Delayed Communication Network," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Mayo 2006, pp. 3298–3303.

Li, Y. y Gupta, K., "Motion Planning of Multiple Agents in Virtual Environments on Parallel Architectures," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Abril 2007, pp. 1009–1014.

——, "A Hybrid Two-layered Approach to Real-Time Motion Planning of Multiple Agents in Virtual Environments," en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Octubre 2006, pp. 4362–4367.

Likhachev, M., Ferguson, D., Gordon, G., Stentz, A., y Thrun, S., "Anytime dynamic A*: An anytime, replanning algorithm," en *Proc. of Int Conf. on Automated Planning and Scheduling, ICAPS'05*, 2005, pp. 262–271.

Lindemann, S. R., Yershova, A., y LaValle, S. M., "Incremental grid sampling strategies in robotics," en *Proc. of the Sixth Int. Workshop on the Algorithmic Foundations of Robotics*, 2004, pp. 297 – 312.

Lindemann, S. R. y LaValle, S. M., *Current Issues in Samplig-Based Motion Planning*, ser. Tracts in Advanced Robotics. Springer Berlin / Heidelberg, 2005, vol. 15, pp. 36 – 54.

Liu, J., Sun, L., Chen, T., Huang, X., y Zhao, C., "Competitive Multi-robot Teleoperation," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Abril 2005, pp. 75–80.

Lo, W.-T., Liu, Y., Elhajj, I., Xi, N., Wang, Y., y Fukuda, T., "Cooperative Teleoperation of a Multirobot System With Force Reflection via Internet," *IEEE/ASME Trans. on Mechatronics*, vol. 9, no. 4, pp. 661–670, Diciembre 2004.

Lozano-Pérez, T., "Spatial planning: A configuration space approach," *IEEE Trans. on Computers*, vol. C-32, no. 2, pp. 108–120, Feb. 1983.

——, "A Simple Motion Planning Algorithm for General Robot Manipulators." *IEEE J. of Robotics and Automation*, vol. RA-3, no. 3, pp. 224 – 238, Junio 1987.

Lozano-Pérez, T. y Wesley, M. A., "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles." *Communications of ACM*, vol. 22, no. 10, pp. 560–570, 1979.

Lumelsky, V. y Cheung, E., "Towards Safe Real-Time Robot Teleoperation: Automatic Whole-Sensitive Arm Collision Avoidance Frees the Operator for Global Control*," *Proc. of IEEE Int. Conf. on Robotics and Automation, ICRA*, vol. 1, pp. 797–802, 1991.

Lumelsky, V. J. y Cheung, E., "Real-Time Collision Avoidance in Teleoperated Whole-Sensitive Robot Arm Manipulators," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 23, no. 1, pp. 194–203, 1993.

Maneewarn, T. y Hannaford, B., "Augmented haptics of manipulator kinematic condition," en *Proc. SPIE of Telemanipulator and Telepresence Technologies VI.*, vol. 3840, no. 54-64, 1999.

Martinez-Palafox, O., Lee, D., Spong, M. W., Lopez, I., y Abdallah, C., "Bilateral Teleoperation of Mobile Robot over Delayed Communication Network: Implementation." en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Octubre 2006, pp. 4193–4198.

Mellado, M., Correcher, C., Catret, J. V., y Puig, D., "Virtualrobot: An open general-purpose simulation tool for robotics," en *European Simulation and Modelling Conference (ESM2003), EUROSIS*, 2003.

Morales, M., Pearce, R., y Amato, N. M., "Analysis of the evolution of c-space models built through incremental exploration," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Roma, Italy, 2007, pp. 1029–1034.

Morishige, K. y Noborio, H., "A teleoperation support system with the help of dual views of cartesian and configuration space," en *Proc. of the 9th IEEE Int. Workshop on Robot and Human Interactive Communication*, 2000, pp. 382 – 387.

Nuño, E., Basañez, L., y Prada, M., "Asymptotic stability of teleoperators with variable time-delays," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA.*, 2009, pp. 4332 –4337.

Nuño, E. y Basañez, L., "Haptic guidance with force feedback to assist teleoperation systems via high speed networks," en *37th International Symposium on Robotics*, Düsseldorf, 2006.

Nuño, E., Ortega, R., y Basañez, L., "An adaptive controller for nonlinear teleoperators," *Automatica*, vol. 46, no. 1, pp. 155 – 159, 2010.

Park, H., Lim, Y.-A., Pervez, A., Lee, B.-C., Lee, S.-G., y Ryu, J., "Teleoperation of a multi-purpose robot over the internet using augmented reality," en *Proc. of the Int. Conf. on Control, Automation and Systems*, 2007, pp. 2456 – 2461.

Payandeh, S. y Stanisic, Z., "On application of virtual fixtures as an aid for telemanipulation and training," en *Proc. of the 10th Symp. on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2002, pp. 18 – 23.

Peer, A., Stanczyk, B., y Buss, M., "Haptic Telemanipulation with Dissimilar Kinematics," en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems.*, 2005, pp. 3493 – 3498.

Pérez, A. y Rosell, J., "A Roadmap to Robot Motion Planning Software Development," *Computer Applications in Engineering Education*, vol. 18(4), pp. 651–660, December 2010.

Plaku, E., Bekris, K. E., y Kavraki, L. E., "OOPS for Motion Planning: An Online Open-source Programming System," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Rome, Italy, 2007, pp. 3711–3716.

Portilla, H. y Basañez, L., "Augmented reality tools for enhanced robotics teleoperation systems," en *3DTV Conference*, May 2007, pp. 1–4.

Quigley, M., Gekey, B., Cnley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., y Ng, A., "Ros: an open-source robot operating system," *Workshop on Open Source Robotics in IEEE Int. Conf. on Robotics and Automation, ICRA*, pp. 1–8, May 2009.

Rabin, S., *AI Game Programming Wisdom 2*. Charles River, 2003.

Rantanen, M. T., "A connectivity-based method for enhancing sampling in probabilistic roadmap planners," *J. of Intelligent and Robotic Systems*, vol. 64, no. 2, pp. 161–178, 2011.

Rodríguez, A., Nuño, E., Palomo, L., y Basañez, L., "Nonlinear control and geometric constraint enforcement for teleoperated task execution," en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2010, pp. 5251 – 5257.

Rosell, J. y Vázquez, I., "Haptic rendering of compliant motions using contact tracking in C-space," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, 2005, pp. 4223–4228.

Rosell, J., Roa, M., Pérez, A., y García, F., "A general deterministic sequence for sampling d-dimensional configuration spaces," *J. of Intelligent and Robotic Systems*, vol. 50, no. 4, pp. 361–374, 2007.

Rosell, J., Vázquez, C., Pérez, A., y Iñiguez, P., "Motion planning for haptic guidance," *J. of Intelligent and Robotic Systems*, vol. 53, no. 3, pp. 223–245, 2008.

Rosell, J., Vázquez, C., y Pérez, A., "C-space decomposition using deterministic sampling and distances," en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, San Diego, California, EE.UU., 2007, pp. 15–20.

Rosell, J., Vázquez, C., Pérez, A., y Iñiguez, P., "Motion Planning for Haptic Guidance," *J. of Intelligent and Robotic Systems*, pp. 223–245, Mayo 2008.

Safaric, R., Parkin, R. M., Czarnecki, C. A., y Calkin, D. W., "Virtual environment for telerobotics," *Integrated Computer-Aided Engineering*, vol. 8, no. 2, pp. 95–104, 2001.

Safaric, R., Sinjur, S., Zalik, B., y Parkin, R., "Control of Robot Arm with Virtual Environment via the Internet," *Proceedings of the IEEE*, vol. 91, no. 3, pp. 422–429, Marzo 2003.

Salisbury, K., Conti, F., y Barbagli, F., "Haptic rendering: Introductory concepts," *IEEE Computer Graphics and Applications*, vol. 24, no. 2, pp. 24–32, 2004.

Schinstock, D., "Approximate solutions to unreachable commands in teleoperation of a robot," *Robotics and Computer-Integrated Manufacturing*, vol. 14, no. 3, pp. 219 – 227, 1998.

Siciliano, B. y Khatib, O., *Springer Handbook of Robotics*, Siciliano, B. y Khatib, O., Eds. Springer, 2008.

Sirouspour, S. y Setoodeh, P., "Multi-operator/Multi-robot Teleoperation: An Adaptive Nonlinear Control Approach," en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, Agosto 2005, pp. 1576–1581.

Stentz, A., "Optimal and efficient path planning for partially-known environments," en *Proc. of IEEE Int Conf. on Robotics and Automation, ICRA'94*, 1994, pp. 3310 –3317 vol.4.

Stilman, M., "Task Constrained Motion Planning in Robot Joint Space," en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robot and Systems*, San Diego, CA, USA, October 2007, pp. 3074 – 3081.

Svenstrup, M., Bak, T., y Andersen, H. J., "Minimising computational complexity of the rrt algorithm a practical approach," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA, ICRA'11*, 2011, pp. 5602–5607.

Svestka, P. y Overmars, M., "Probabilistic Path Planning," en *Robot Motion Planning and Control*, Laumond, J.-P., Ed. Springer, 1998, ch. 5, pp. 255 – 304.

Todt, E., Raush, G., y Suárez, R., "Analysis and classification of multiple robot coordination methods," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, 2000, pp. 3158–3163.

Trovato, K., "Differential A*: an adaptive search method illustrated with robot path planning for moving obstacles and goals, and an uncertain environment," en *Proc. of IEEE Int Workshop on Tools for Artificial Intelligence: Architectures, Languages and Algorithms.*, 1989, pp. 624 –639.

Trovato, K. y Dorst, L., "Differential A*," *IEEE Trans. on Knowledge and Data Engineering*, vol. 14, no. 6, pp. 1218 – 1229, 2002.

van den Berg, J., Stillman, M., Kuffner, J., Lin, M. C., y Manocha, D., "Path planning among movable obstacles: A probabilistically complete approach," en *Algorithmic Foundation of Robotics VIII: Selected Contributions of the Eighth Inter. Workshop on the Algorithmic Foundations of Robotics (WAFR), Springer Tracts in Advanced Robotics (STAR)*, vol. 57, 2009, pp. 599–614.

van den Bergen, G., "Efficient collision detection of complex deformable models using AABB trees," *J. Graph. Tools*, vol. 2, no. 4, pp. 1–13, 1997.

Vazquez, C., Rosell, J., Chirinos, L., y Domínguez, O., "Haptic primitives guidance based on the kautham path planner," en *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2010, pp. 4686 – 4691.

Verner, L. N. y Okamura, A. M., "Force & torque feedback vs force only feedback," en *Proc. of the World Haptic Conference*, 2009, pp. 406–410.

Vázquez, C. y Rosell, J., "Haptic guidance based on harmonic functions for the execution of teleoperated assembly tasks," en *2007 IFAC Workshop on Intelligent Assembly and Disassembly IAD'07.*, Alicante, Spain, Mayo 2007, pp. 88–93.

Wang, Y., Sillitoe, I., y Mulvaney, D., "Mobile Robot Path Planning in Dynamic Environments," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Abril 2007, pp. 71–76.

Yao, Z. y Gupta, K., "Path planning with general end-effector constraints," *Robotics and Autonomous Systems*, vol. 55, pp. 316–327, 2007.

Yershov, D. S. y LaValle, S. M., "Simplicial dijkstra and a* algorithms for optimal feedback planning," en *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2011, pp. 3862–3867.

Yershova, A. y LaValle, S., "Improving Motion-Planning Algorithms by Efficient Nearest-Neighbor Searching," *IEEE Trans. on Robotics*, vol. 23, no. 1, pp. 151–157, February 2007.

Zhang, L., Kim, Y. J., y Manocha, D., "A hybrid approach for complete motion planning," en *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS*, 2007, pp. 7–14.

Zucker, M., Kuffner, J., y Branicky, M., "Multipartite RRTs for Rapid Replanning in Dynamic Environments," en *Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, Abril 2007, pp. 1603–1609.