CENTRO INTERNACIONAL DE ESTUDOS
DE DOUTORAMENTO E AVANZADOS
DA USC (CIEDUS)

TESIS DE DOCTORADO

MOTION PLANNING UNDER UNCERTAINTY FOR AUTONOMOUS NAVIGATION OF MOBILE ROBOTS AND UAVS

Presentada por:

Adrián González Sieira

Dirigida por:

Manuel Mucientes Molina
Alberto J. Bugarín Diz

ESCUELA DE DOCTORADO INTERNACIONAL

PROGRAMA DE DOCTORADO EN INVESTIGACIÓN EN TECNOLOGÍAS DE LA INFORMACIÓN

Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS)

SANTIAGO DE COMPOSTELA
Septiembre de 2019

DECLARACIÓN DEL AUTOR DE LA TESIS

Motion Planning under Uncertainty for Autonomous Navigation of Mobile Robots and UAVs

Adrián González Sieira

Presento mi tesis, siguiendo el procedimiento adecuado al Reglamento, y declaro que:

1. *La tesis abarca los resultados de la elaboración de mi trabajo.*
2. *En su caso, en la tesis se hace referencia a las colaboraciones que tuvo este trabajo.*
3. *La tesis es la versión definitiva presentada para su defensa y coincide con la versión enviada en formato electrónico.*
4. *Confirmo que la tesis no incurre en ningún tipo de plagio de otros autores ni de trabajos presentados por mí para la obtención de otros títulos.*

En Santiago de Compostela, Septiembre de 2019

Fdo. Adrián González Sieira

AUTORIZACIÓN DEL DIRECTOR/TUTOR DE LA TESIS

Motion Planning under Uncertainty for Autonomous Navigation of Mobile Robots and UAVs

Dr. Manuel Mucientes Molina, Profesor Contratado Doctor del Área de Ciencia de la Computación e Inteligencia Artificial de la Universidade de Santiago de Compostela

Dr. Alberto J. Bugarín Diz, Catedrático de Universidad del Área de Ciencia de la Computación e Inteligencia Artificial de la Universidade de Santiago de Compostela

INFORMAN:

Que la presente tesis, corresponde con el trabajo realizado por ***Adrián González Sieira*** *bajo nuestra dirección, y autorizamos su presentación, considerando que reúne los requisitos exigidos en el Reglamento de Estudios de Doctorado de la USC, y que como directores de ésta no incurre en las causas de abstención establecidas en Ley 40/2015.*

En Santiago de Compostela, Septiembre de 2019

Fdo. Manuel Mucientes Molina
Director de la tesis

Fdo. Alberto J. Bugarín Diz
Director de la tesis

There's nothing quite as frightening as someone who knows they are right.

Michael Faraday

Agradecimientos

Me gustaría empezar con el mayor de los agradecimientos a mis directores, Manuel Mucientes y Alberto Bugarín, tanto por todas las horas de dedicación, de esfuerzo y de implicación en este proyecto, como por la confianza que han depositado en mí al ofrecerme la oportunidad de iniciarme en la investigación. Sin ellos esta tesis no habría sido posible.

Quisiera agradecer también al Departamento de Electrónica y Computación, así como al Centro de Investigación en Tecnoloxías Intelixentes (CiTIUS), de la Universidade de Santiago de Compostela, por el excelente entorno de trabajo, así como por todos los recursos que han hecho posible el desarrollo de esta tesis. Gracias también al director adjunto del CiTIUS, Paulo Félix, por sus consejos y palabras de aliento, así como a las unidades de apoyo del centro por su constante esfuerzo. También me gustaría mencionar al Prof. Dr. Dirk Heylen, a la Prof. Dr. Vanessa Evers, y a todo el grupo del HMI (Human Media Interaction) de la Universidad de Twente por hacerme un hueco entre ellos durante mi estancia. Del mismo modo, quiero agradecerle a Alejandro Catalá el haber facilitado el primer contacto con ellos, así como su implicación y su ayuda.

No podía faltar un agradecimiento especial a todos los amigos y compañeros del CiTIUS, sobre todo a Teffa, Arturo, Pablo, Jorge, Tomás, Tino, Ángel, Álvaro, Ismael, Borja, David, Jano, Víctor, Chapela, Andrea, Yago, Lorenzo y Efrén, por todas las risas y decibelios que hacen del LS2 un sitio único. Gracias también a Ricardo Alonso y —una vez más— a Jorge por todo el trabajo que hicieron para preparar las pruebas con el UAV, así como a Dani Cores por su gran ayuda durante los experimentos de vuelo ~~y por no dejarse llevar por el pánico en los momentos de tensión~~.

Por supuesto, me gustaría agradecer también a todas las entidades que han hecho posible el desarrollo de esta tesis con su financiación. Así, me gustaría mencionar en primer lugar al programa de Formación de Profesorado Universitario (FPU) del Ministerio de Educación,

Cultura y Deporte (ref. AP2012-5712), así como al Ministerio de Ciencia, Innovación y Universidades (ref. TIN2017-84796-C2-1-R). También a la Consellería de Educación, Universidades y Formación Profesional de la Xunta de Galicia (ref. ED431C 2018/29 y "acreditación 2016-2019, ED431G/08", co-financiadas por el Fondo Europeo de Desarrollo Regional del programa ERDF/FEDER).

En lo personal, me gustaría también agradecer a mi familia por el infinito apoyo que me han dado durante todos estos años, en especial a mi madre y a mis hermanas. Nunca he podido quitaros del todo la idea de que pasaba los días construyendo robots, pero no hay duda de que sin vosotras no habría llegado hasta aquí.

Gracias también a mis amigos Elena, Ricardo y Luis, por todas las horas que hemos dedicado a repasar las aventuras y desventuras de este proyecto, así como por las risas que me ayudaron tanto en los días buenos como en los no tan buenos.

Para terminar, y aunque las palabras puedan quedarse cortas, me gustaría agradecerle de corazón a Álex todo el cariño y la confianza que ha depositado en mí durante todo este tiempo, y por animarme a dar siempre lo mejor de mí mismo.

Septiembre de 2019

Resumen

Uno de los objetivos de la robótica es el desarrollo de sistemas autónomos, cuyo interés para ciertas aplicaciones prácticas es cada vez mayor. Son muchas las tareas que se podrían beneficiar de algún tipo de automatización, entre las que se encuentran la exploración de entornos, algunos procesos de manufactura, la vigilancia o la reconstrucción de escenas, por citar algunos ejemplos. La navegación es una funcionalidad básica de cualquier robot autónomo y forma parte de estas y muchas otras aplicaciones prácticas. Por ello, la planificación de rutas para guiar a un robot móvil evitando obstáculos es un problema que ha recibido una atención considerable por parte de la comunidad científica [111, 115]. En particular, se ha considerado especialmente relevante el problema de la obtención de caminos adaptados al modelo de movimiento del robot para cumplir con sus restricciones cinemáticas y, al mismo tiempo, aprovechar sus prestaciones de forma adecuada.

Aunque la formulación inicial del problema de planificación se planteó de una forma muy simple, han sido muchos los factores que posteriormente se han identificado como elementos clave a tener en cuenta para resolverlo. Algunos de dichos factores contribuyen a aumentar la complejidad computacional de forma significativa, como las restricciones de movimiento del robot, las diferentes fuentes de incertidumbre o el conocimiento parcial del entorno. Sin embargo, aunque tenerlos en cuenta es necesario para asegurar la validez de las rutas, no todas las aproximaciones gestionan estos elementos de la misma forma.

Por ejemplo, la inclusión de las restricciones cinemáticas en la planificación es un problema que presenta gran dificultad [30, 115], y su relevancia es indiscutible por el hecho de que cada tipo de robot tiene las suyas propias. Sin embargo, para reducir la complejidad computacional y obtener rutas de forma eficiente, muchas propuestas se han centrado en una versión simplificada del problema que no tiene en cuenta dichas restricciones. Así, estas aproximaciones o bien asumen que la navegación se realizará a velocidades muy bajas, de tal forma

que los efectos de la inercia sobre el robot pueden ignorarse, o bien confían en que el controlador solucione las cuestiones relacionadas con la viabilidad de las rutas y su seguimiento de forma precisa. Sin embargo, en muchos casos dichas asunciones no son razonables, y aplicar este tipo de estrategias puede limitar de manera significativa el rendimiento del sistema. Los vehículos cuyo radio mínimo de giro es alto, como los coches autónomos [123, 124], son un claro ejemplo de este problema. También lo son los sistemas con modelos dinámicos complejos, como aquellos que navegan a velocidades altas [33], los vehículos aéreos [58] o los robots marinos [157], ya que además de las restricciones cinemáticas es necesario tener en cuenta los efectos de unas inercias significativas. En estos casos, simplificar el problema afectaría no sólo al rendimiento del robot, al obtener soluciones que deben ser necesariamente conservadoras para que puedan ejecutarse, sino también a su fiabilidad, debido a la alta tasa de fallos causada por los intentos del controlador de ejecutar rutas que en la práctica no son viables.

El espacio de estados de un problema de planificación representa todas las situaciones en las que el robot puede encontrarse. Dicha representación debe incluir toda la información relevante sobre los distintos aspectos que puedan afectar a los movimientos inmediatos y futuros del robot. Esto incluye no sólo el conjunto de poses en las que el robot pueda estar, sino también otros elementos clave como sus velocidades lineal y angular. Todo esto incrementa de forma inevitable la dimensionalidad del problema y, por ende, su complejidad.

En la bibliografía se han descrito distintas aproximaciones capaces de abordar con éxito la planificación de movimiento en problemas de alta dimensionalidad. Dichas aproximaciones utilizan técnicas basadas en muestrear el espacio de estados, disminuyendo así la complejidad computacional necesaria para representarlo, y permitiendo la obtención de soluciones en un tiempo razonable. Según el tipo de muestreo, dichos algoritmos se pueden clasificar en estocásticos [90, 92, 118] y deterministas [21, 158]. Entre estos últimos, las retículas de estados [158] destacan por discretizar el espacio de estados siguiendo un patrón regular, una estrategia que ha demostrado dar buenos resultados en dos métricas ampliamente utilizadas para evaluar la calidad del muestreo: dispersión y discrepancia.

Las retículas de estados también permiten reducir la computación que se debe realizar durante la planificación. Una de las tareas más costosas, en especial para sistemas con modelos dinámicos complejos, es la generación de trayectorias para conectar los estados discretos. Debido a la distribución regular de dichos estados en la retícula, se puede obtener a partir del modelo cinemático del robot un conjunto canónico de movimientos, o acciones primitivas, que

represente las distintas maniobras que el robot puede ejecutar. Cada una de dichas acciones encapsula los comandos de velocidad que deben ejecutarse para mover al robot entre dos estados, de tal forma que una misma acción puede reutilizarse para conectar cualquier otro par de estados distribuidos de la misma manera, independientemente de su posición. Así, la conectividad de toda la retícula queda representada a través de un conjunto finito de acciones primitivas, que además se pueden obtener *a priori*. Esta forma de representar el espacio de estados y el espacio de acciones es muy eficiente, y su estructura se corresponde con un grafo cuyos vértices son los estados discretos, y cuyos arcos son las acciones que los conectan.

Al ser obtenidas a partir del modelo de movimiento del robot, las acciones primitivas ya codifican sus restricciones cinemáticas, por lo que está garantizado que su ejecución es viable. Así, este tipo de planificación se puede formular de tal forma que se puede resolver con un algoritmo de búsqueda sobre grafos [66, 125, 188], lo que permite obtener la mejor ruta entre cualquier par de estados de la retícula de forma muy eficiente. Un punto a tener en cuenta en relación con esto es que la discretización del espacio de estados puede comprometer la optimalidad de las soluciones. Sin embargo, en ciertas condiciones se puede conservar su optimalidad asintótica [84, 90].

Para muchas aplicaciones prácticas, desde la navegación autónoma a la manipulación de objetos, la seguridad y precisión de las soluciones obtenidas es de vital importancia. En estos casos, el criterio para seleccionar cuál es el mejor camino no sólo depende de medidas como su longitud o tiempo de ejecución, sino en maximizar la probabilidad de que la misión se lleve a cabo con éxito. Muchos robots se ven afectados por la incertidumbre en el movimiento [112], que tiene su origen tanto en las imprecisiones que el modelo cinemático tiene con respecto al comportamiento real, como en la información ruidosa o parcial que los sensores proporcionan sobre el estado del sistema. Así, la incertidumbre de cada ruta puede ser diferente, ya que depende de los movimientos ejecutados y de las medidas recibidas por los sensores. En este contexto, es particularmente interesante estimar la incertidumbre *a priori*, de tal forma que el mejor camino pueda obtenerse con anterioridad a que el robot comience a moverse. Sin embargo, hay dos cuestiones que continúan siendo un reto: estimar la incertidumbre durante la planificación, de modo que el problema continúe siendo abordable, y determinar de forma fiable cuál es la idoneidad de cada camino a partir de dicha incertidumbre.

Las distintas propuestas existentes en la bibliografía se han centrado en aspectos como: la planificación de caminos que maximicen la información adquirida por los sensores [26, 163], la obtención de políticas de control enfocadas en la seguridad de la navegación [199], o la

generación de rutas con la mínima probabilidad de colisión [6]. Aunque todas estas estrategias son válidas, su idoneidad dependerá del tipo de aplicación a la que estén destinadas, así como del comportamiento que se espera del sistema autónomo. En general, para la navegación de robots resulta particularmente interesante la obtención de rutas con la mínima probabilidad de colisión.

En contraste con aquellas aproximaciones cuyo objetivo es minimizar la incertidumbre o el tiempo de ejecución de las rutas, el hecho de centrarse en la probabilidad de colisión podría generar un gran número de soluciones candidatas adecuadas para la navegación. Por ello, es necesario definir de forma más precisa la salida que se espera del planificador, algo que se puede conseguir utilizando criterios adicionales para evaluar los planes. Más concretamente, para el propósito de navegar de forma autónoma sería adecuado obtener soluciones que minimicen la probabilidad de colisión, pero también su tiempo de ejecución.

En relación con la probabilidad de colisión, su estimación puede hacerse a partir de las distribuciones de probabilidad predichas para los distintos caminos explorados durante la planificación. Aunque se han propuesto distintas técnicas para conseguir esto, la mayor parte de ellas no tienen en cuenta las dimensiones reales del robot y, en su lugar, utilizan aproximaciones tales como robots puntuales [26, 130] o representaciones circulares [120, 197]. Aunque de esta forma se simplifica la planificación de forma significativa, esto puede en algunos casos comprometer la fiabilidad de las soluciones. Además, este tipo de aproximaciones hace que sea imposible tener en cuenta la incertidumbre en la orientación del robot y, aunque de forma sistemática esto no se ha tenido en cuenta en los trabajos previos, podría ser de gran relevancia para robots con formas irregulares o asimétricas, como algunas plataformas robóticas que se utilizan para el transporte de carga.

Otro aspecto importante a tener en cuenta es la garantía de completitud del planificador, esto es: la capacidad de devolver una solución válida, o bien de determinar si ésta no existe. De entre las distintas aproximaciones que tienen en cuenta las restricciones cinemáticas es difícil encontrar alguna que garantice totalmente la completitud, ya que la mayoría de ellas utilizan técnicas de discretización para reducir la complejidad computacional del problema. En su lugar, dichas aproximaciones ofrecen garantías más laxas. Así, por un lado hay métodos que son probabilísticamente completos [12, 94], algo que ocurre cuando la probabilidad de encontrar una solución válida aumenta con el número de muestras. Otros son completos en función de la resolución [21], de forma que su completitud es asintótica cuando la fidelidad de la representación del espacio de estados tiende a infinito. Las retículas de estados se

encuentran entre este segundo grupo de estrategias, por lo que su rendimiento se relaciona directamente con la fidelidad de los estados muestreados.

En relación con lo anterior, utilizar fidelidades altas incrementa tanto la potencia del planificador como la calidad de las soluciones, aunque a costa de su eficiencia. Por lo tanto, lo deseable es utilizar fidelidades lo más bajas posibles para poder obtener soluciones en un tiempo razonable, conservando las garantías de optimalidad y completitud del planificador. En este sentido, las retículas de estados con fidelidad graduada [159] pueden ser útiles para encontrar el punto de compromiso entre la potencia y la eficiencia. La idea es que no todas las regiones del entorno tienen la misma importancia a la hora de planificar, por lo que se pueden utilizar distintas resoluciones para discretizar el espacio de estados sin comprometer los resultados de forma significativa.

Algunos trabajos previos [123, 159] ya abordaron esta cuestión definiendo regiones de alta fidelidad alrededor del robot, y baja en el resto del entorno. Con esta estrategia la estructura de la retícula de estados cambia cuando el robot se desplaza, por lo que es necesario actualizar de forma constante la ruta planificada. Sin embargo, la efectividad de este método cuando se tiene en cuenta la incertidumbre durante la planificación es muy limitada. Esto es porque, con los cambios en la retícula, las incertidumbres y las probabilidades de colisión de todos los caminos afectados deben recalcularse, una operación que es computacionalmente costosa. Además, no hay garantías de que al actualizar la solución se obtendrá un camino similar al anterior ya que, a diferencia de otros criterios, el coste asociado a la incertidumbre no tiene por qué ser monótonamente decreciente hacia el objetivo.

A pesar de la relevancia de la representación del espacio de estados, éste no es el único elemento que juega un papel crucial en la eficiencia del planificador. El uso de heurísticas es clave para reducir el número de estados que los algoritmos de búsqueda necesitan explorar para obtener soluciones óptimas. Su eficacia está directamente relacionada con su valor informativo, por lo que una heurística que proporcione estimaciones precisas da lugar a tiempos de planificación menores. Sin embargo, la complejidad de obtener heurísticas capaces de incorporar las restricciones cinemáticas no es trivial.

Aunque hay distintas alternativas para estimar el coste entre el estado actual del robot y el objetivo, una estrategia que ha demostrado ser efectiva es la combinación de heurísticas que traten por separado el modelo cinemático del robot [100] y los obstáculos del entorno [124]. El primer tipo de heurística puede calcularse *a priori*, conocido dicho modelo, pero el segundo tipo ha de obtenerse en función de cada planificación, ya que depende del objetivo y del

entorno. Por lo tanto, su eficiencia resulta particularmente relevante para no comprometer los tiempos de obtención de las soluciones. Una estrategia eficaz es construir una malla de baja dimensionalidad para estimar el coste debido a los obstáculos del mapa, que puede obtenerse en tiempo real para la mayoría de problemas bidimensionales y entornos pequeños. Sin embargo, dicha estrategia sufre los mismos problemas de escalabilidad que el propio problema de planificación del movimiento y, por tanto, no puede aplicarse directamente a problemas de gran tamaño o de mayor dimensionalidad. A pesar de esto, la mejora de los tiempos de obtención de este tipo de heurísticas es algo que todavía no se ha abordado en la bibliografía.

El aumento de popularidad que muchos tipos de robots móviles han experimentado en los últimos años hace que cada vez sean más las aplicaciones prácticas de la planificación del movimiento y de los sistemas de navegación autónoma. Más recientemente, esto ha venido de la mano del significativo incremento en el uso de Vehículos Aéreos No Tripulados (UAVs). Aunque bajo esta denominación se pueden englobar muchos tipos de plataformas robóticas, esta tesis se centra en sistemas multi-rotor debido a su maniobrabilidad y al grado de precisión de sus movimientos. Además, dada su flexibilidad, pueden utilizarse en una gran variedad de entornos complejos, lo que abre la puerta a un número de aplicaciones aún mayor.

En algunos contextos es particularmente interesante automatizar ciertas tareas con el fin de evitar la intervención humana en la medida de lo posible. El beneficio es aún mayor en lo que respecta a trabajos repetitivos, con riesgos para la salud o peligrosos. Típicamente, los primeros pasos en pro de la automatización se consiguen introduciendo plataformas robóticas controladas manualmente. En el caso de los UAVs, lo habitual es que un operador humano esté a cargo tanto de la supervisión como del control del vuelo. Sin embargo, en algunos casos dicho control manual no es posible, o no resulta adecuado, ya sea por la distancia entre el operador y el aparato o la precisión que requieren algunas maniobras, que pueden estar fuera del alcance del piloto. En esos casos, es claramente beneficioso contar con un sistema de navegación autónoma que ayude al operador a cumplir algunas misiones.

Algunos ejemplos de tareas repetitivas son la exploración de entornos [172], vigilancia de tráfico [34, 164] o la reconstrucción de escenas 3D [146]. Normalmente, este tipo de misiones incluye la ejecución de distintas rutas y mantenerse en el aire durante la toma de datos, por lo que navegar de forma autónoma sería interesante para liberar al operador del control normal del UAV y dejarlo a cargo de analizar los datos recibidos y tomar el control sólo en situaciones que lo requieran. En cuanto a las tareas con riesgos para la salud, tenemos ejemplos como la detección y control de incendios forestales [209], la detección de escapes en tuberías de gas

[173, 186] o el pintado con técnicas de spray en superficies inaccesibles [200].

Entre las tareas peligrosas, tal vez la más notable sea la inspección de infraestructuras [31, 189, 204, 207], que normalmente involucra a trabajadores humanos, ha de ser realizada de forma periódica y requiere llegar a zonas de difícil acceso. Aquí, el beneficio de introducir un UAV es claro, no sólo para reducir el riesgo personal de dichas operaciones, sino también porque pueden recoger datos que permitan alertar de los problemas que se vayan encontrando. Debido a la variedad de sensores que pueden llevar a bordo este tipo de vehículos, la información que pueden recoger es muy variada y este tipo de inspecciones pueden, en muchos casos, mejorar los resultados de una inspección visual.

Otro contexto en el que la aplicación de UAVs se ha popularizado en los últimos años es el transporte de ciertas mercancías. Algunas compañías como Amazon, DHL, Google o Swiss Post han lanzado distintas iniciativas en este sentido [4, 57, 81, 82], algunas de las cuales se han centrado en el envío urgente de material médico [4, 57, 192]. Tal es el interés de este tipo de aplicaciones que algunas Administraciones Públicas han tenido sus propias ideas de aplicación en este ámbito. Por ejemplo, la Xunta de Galicia lanzará en el futuro un sistema que utilizará UAVs para la entrega rápida de desfribriladores con el fin de mejorar la atención temprana de paradas cardiorrespiratorias en el Camino de Santiago [35].

Dada la complejidad del modelo cinemático de los UAVs, es indispensable tenerlo en cuenta para obtener rutas realistas. Además, para muchas de las aplicaciones mencionadas previamente, garantizar la seguridad del vuelo es especialmente relevante. Todo esto debe hacerse sin comprometer la eficiencia del planificador, ya que de otro modo su integración en sistemas reales no sería posible.

Teniendo en cuenta este contexto, el principal objetivo de esta tesis es el desarrollo de una aproximación de planificación del movimiento que, de forma fiable y eficiente, permita la navegación autónoma de robots móviles y UAVs. Este objetivo general se concreta en los siguientes objetivos específicos:

- Desarrollar una aproximación capaz de obtener caminos óptimos en términos de seguridad y tiempo de ejecución, que incluya en tiempo de planificación las restricciones cinemáticas del robot y la incertidumbre asociada al movimiento y a las observaciones.

- Mejorar la eficiencia de los algoritmos de planificación del movimiento mediante la introducción de técnicas multi-resolución que permitan reducir la complejidad computacional del problema, así como analizar el impacto de dichas técnicas en los resultados.

- Integrar los algoritmos desarrollados en una arquitectura para la navegación autónoma de robots móviles y UAVs, y validar la propuesta con experimentos reales en distintos tipos de sistemas.

El trabajo que se ha desarrollado en esta tesis ha dado lugar a diversas publicaciones, entre las que destacan un artículo en la revista *Robotics and Autonomous Systems*, Q1 en el ranking SJR 2018, y un trabajo presentado en el congreso *International Conference on Robotics and Automation (ICRA)*, actualmente clasificado como Tipo A, Clase 2 en el ranking GII-GRIN-SCIE (GGS). En esta memoria se detallan las contribuciones del trabajo desarrollado y los resultados más relevantes de la experimentación.

En concreto, el Capítulo 2 analiza la investigación previa relacionada con la tesis. En primer lugar se detallan las aproximaciones clásicas que abordaron la planificación del movimiento teniendo en cuenta las restricciones cinemáticas, así como la generación de soluciones óptimas. A continuación, este capítulo detalla los trabajos centrados en planificación con incertidumbre, así como las distintas técnicas que hacen posible la obtención de soluciones adaptadas a la incertidumbre que se ha estimado *a priori*, o su probabilidad de colisión. Finalmente, se analizan las distintas propuestas para mejorar la eficiencia de la planificación, así como aquellos trabajos que se centraron en obtener rutas para UAVs. En esta revisión del estado del arte se ha identificado la necesidad de mejorar la eficiencia de los métodos existentes para planificar con incertidumbre, así como su fiabilidad. Esto proporciona el contexto en el que se encuadran las contribuciones de esta tesis, que se detallan en los dos capítulos siguientes.

El Capítulo 3 describe la estrategia de planificación desarrollada en esta tesis. Nuestra propuesta combina de forma novedosa distintos métodos del estado del arte, con el fin de obtener un planificador fiable, eficiente y escalable. En primer lugar, se ha utilizado una retícula de estados que representa el problema y permite tener en cuenta las restricciones cinemáticas del robot de forma eficiente. Dada la estructura de la retícula, se ha formulado el problema de la planificación como una búsqueda sobre grafos, para la que se ha utilizado el algoritmo Anytime Dynamic A* (AD*) [125]. Dicho algoritmo permite obtener soluciones óptimas respecto a una función de coste que utiliza distintos criterios con un orden de prioridad definido.

Así, primero se minimiza la probabilidad de colisión de las soluciones, luego el tiempo de ejecución de dichos caminos, y finalmente la incertidumbre final. De esta forma se obtienen caminos que son seguros y rápidos para navegar de forma autónoma. En segundo lugar, para estimar la probabilidad de colisión, nuestro método predice primero la incertidumbre tenien-

do en cuenta las imprecisiones tanto del movimiento como de las observaciones, así como el tipo de controlador. Por último, nuestra propuesta introduce una nueva aproximación que permite obtener una retícula de estados con fidelidad graduada, y también utiliza heurísticas que permiten obtener soluciones de una forma muy eficiente. Por un lado, la fidelidad graduada permite reducir el número de estados y de posibles caminos de la retícula, al tiempo que conserva la potencia del planificador y la calidad de las soluciones. Por otro lado, se han combinado dos heurísticas diferentes que estiman el coste a la meta teniendo en cuenta tanto las restricciones cinemáticas del robot como los obstáculos en el mapa.

Además de esta estrategia de planificación, el Capítulo 3 también detalla las principales contribuciones de esta tesis en el ámbito de la planificación del movimiento. En primer lugar, presentamos un método para estimar la probabilidad de colisión de las rutas de forma realista. Dicho método muestrea de forma determinista las distribuciones de probabilidad predichas, teniendo en cuenta tanto la incertidumbre en la orientación del robot como sus dimensiones reales. La validación se ha realizado en 15 experimentos diferentes, variando el entorno, el tipo de robot y las condiciones de incertidumbre. Los resultados muestran que la probabilidad de colisión se ha estimado de forma fiable en todos los casos, y los caminos planificados no registraron ninguna colisión durante la ejecución de las rutas planificadas.

En segundo lugar, este capítulo presenta una aproximación para la obtención de una retícula de estados con fidelidad graduada que tiene en cuenta los obstáculos del mapa, la maniobrabilidad del robot y la incertidumbre predicha. Así, las áreas del mapa en las que la planificación resulta sencilla, como los espacios abiertos, se representan en la retícula con un menor número de estados y acciones entre ellos. Dicho método se ha validado con varios modelos de movimiento y entornos con diferentes configuraciones de obstáculos, y los resultados muestran que el tiempo de planificación se ha reducido en promedio un 88,5 % con respecto a una retícula de estados clásica.

Finalmente, en este capítulo se detalla una heurística multi-resolución que estima el coste a la meta teniendo en cuenta los obstáculos en el mapa, para lo que se construye una malla que representa una versión simplificada del problema de planificación. La resolución de esta malla varía en función de los obstáculos, y los resultados muestran que su tiempo de cómputo es en promedio un 65,5 % mejor que con una malla de resolución fija, mientras que su precisión es un 4 % mayor.

En el capítulo 4 se detalla la aplicación del método de planificación propuesto para la navegación autónoma de UAVs. Esto incluye: i) los nuevos algoritmos que permiten tratar

con la mayor dimensionalidad de este problema; ii) su integración en una arquitectura flexible compuesta por el planificador, un controlador y un sistema para estimar el estado actual del robot; y iii) su validación en experimentos reales.

Concretamente, en este capítulo se describe una nueva heurística que permite operar en entornos 3D, cuya conectividad entre las distintas posiciones de la malla ha sido rediseñada para reducir su densidad y mejorar su eficiencia. También se introduce un nuevo método para estimar la probabilidad de colisión en el que las distribuciones de probabilidad predichas se muestrean evitando la obtención de poses poco representativas que puedan penalizar el rendimiento del planificador. Estos nuevos algoritmos se han validado en tres experimentos reales de navegación autónoma utilizando un UAV modelo AscTec Pelican[1], cuyos resultados muestran la fiabilidad y la eficiencia del sistema. Dichos experimentos se realizaron en entornos diferentes y bajo distintas restricciones de movimiento. En el último de ellos se ha utilizado el sistema de navegación autónoma propuesto para una aplicación práctica real, la reconstrucción de escenas 3D con una cámara RGB-D. Los resultados muestran que en dicho experimento se ha obtenido de forma autónoma un modelo en el que el error promedio de la reconstrucción 3D es de tan solo del 1,08 %.

Por último, el Capítulo 5 resume las conclusiones de la investigación realizada en esta tesis, además de discutir posibles líneas de trabajo futuro que permitirían mejorar los resultados de planificación en ciertos contextos. También se discute la posibilidad de extender los algoritmos de planificación propuestos para tener en cuenta las dinámicas de navegación en entornos sociales, lo que haría posible que los robots tuviesen un movimiento más natural, o "socialmente aceptable", en presencia de personas. Esto daría lugar a que el robot realizase movimientos más predecibles, lo que mejoraría la aceptación del comportamiento del robot por parte de los humanos.

La idea es que las rutas puedan obtenerse teniendo en cuenta no sólo la distancia entre el robot y las personas [109, 133, 190, 202, 203], sino también la incertidumbre en el movimiento, que afecta a la ejecución de las soluciones. En este sentido, hemos iniciado la extensión de nuestra técnica de muestreo que estima la probabilidad de colisión, de tal forma que pueda también medir la idoneidad de un determinado contexto persona-robot. Esta medida se obtiene a partir de las poses muestreadas utilizando la teoría de las distancias proxémicas [104, 170, 183], que estudia cómo se percibe una interacción según la distancia entre dos sujetos. Dicho criterio de coste puede integrarse de forma fácil en la estrategia de planificación

[1] `http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-pelican/`

propuesta, al ser un elemento más a minimizar por el algoritmo Anytime Dynamic A*.

Como líneas de trabajo futuro se podría extender el modelo para considerar las dimensiones del robot además de la incertidumbre, ya que algunos estudios afirman que puede afectar a cómo las personas perciben su movimiento [109, 190]. También se podría tener en cuenta la dirección en la que se mueve el robot respecto a la gente, ya que en la bibliografía [201] se indica que las personas reaccionan de manera distinta en función de dicha dirección, algo que podría utilizarse en beneficio de la navegación en entornos sociales [105].

El objetivo final de estas líneas de trabajo futuro sería la obtención de planes que, además de priorizar la seguridad del movimiento respecto a los obstáculos del entorno, también permitiesen al robot tener un movimiento que pudiese describirse como natural. Como resultado de planificar teniendo en cuenta la incertidumbre, la distancia persona-robot estaría adaptada a las circunstancias del entorno y a las condiciones de incertidumbre a lo largo de la ruta.

Contents

CHAPTER 1

INTRODUCTION

1.1 Motivation

Developing autonomous robots is one of the ultimate goals in robotics, which is of great interest for many practical applications, among which environment exploration, manufacturing, surveillance or scene reconstruction are only a few examples of the wide range of tasks that could benefit from some sort of automatization. In this regard, one of the basic features that anybody would expect from an autonomous robot is being able to successfully navigate in environments of different natures. Thus, the problem of obtaining a set of actions to drive a mobile robot while avoiding the obstacles has received significant attention from the research community [111, 115]. Obtaining plans that result in motions adapted to the robot kinematics is especially relevant to take full advantage of its capabilities and, at the same time, to deal with the limitations of each robotic platform.

The motion planning problem is known to be challenging due to the need of considering constraints of different kinds, such that the feasibility of the obtained paths is guaranteed. The classical *Piano Mover's Problem* [168] describes a simple version of motion planning in which, given the starting and goal poses, an algorithm must obtain the set of motions to move a piano through an indoor environment without colliding with the obstacles around it, as shown in Figure 1.1. While motion planning for mobile robots has been formulated in similar terms, there are many factors that contribute to increase the difficulty of the problem, such as the motion restrictions, the different sources of uncertainty or the partial knowledge about the environment. Notwithstanding, not all the existing approaches that address motion planning take into account or deal with these elements in the same way.

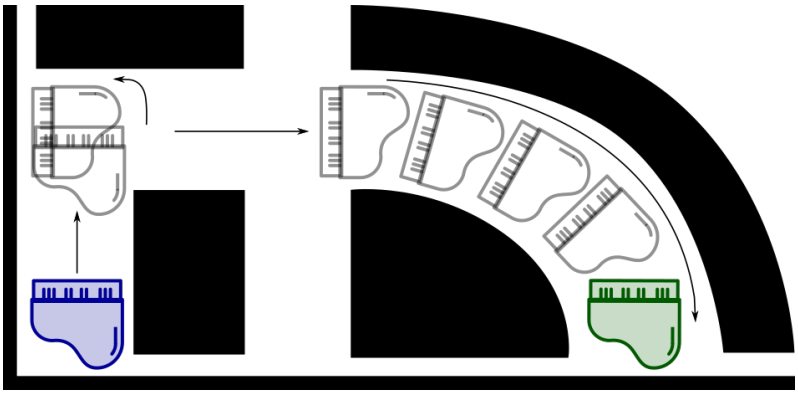


Figure 1.1: Example of the Piano Mover's problem. The obstacles are in black and the starting configuration is in blue, while the goal is in green. The arrows represent the direction of motion.

For example, motion planning taking into account the kinematic constraints is considered a hard problem itself [30, 115]. Moreover, due to the fact that such constraints are imposed by many robotic platforms, the relevance of dealing with them at planning time is undeniable. While some strategies attempted to alleviate the computational complexity issues by disregarding them, hence solving a simplified version of the problem, this is not suitable as a general strategy. This is because this simplification either assumes navigation at very low speeds, such that the dynamic effects on the robot due to the inertia are minimized, or leaves the issues caused by disregarding the kinematic constraints at planning time to be solved by a motion controller in execution time. However, in many cases these assumptions are not reasonable and disregarding the dynamic restrictions might significantly impact the performance of the autonomous system due to the severity of its mobility limitations.

Clear examples can be found in vehicles[2] with large minimum turning radius, like autonomous cars [123, 124] and composite vehicles [141]; or systems with complex dynamics, such as those that navigate at high speeds [33], Unmanned Aerial Vehicles (UAVs) [58] or marine robots [157], which in addition to the mobility restrictions have to deal with strong physical effects like the inertia. In these cases, planning ignoring the kinematic constraints is not reasonable due to the impact that it has in the practical results. While the overall performance of the autonomous robot would fall as a result of obtaining highly conservative

[2] The generic term "vehicle" will be occasionally used in this manuscript as synonim of "robot" or "robotic platform" to avoid repetitions.

solutions, the reliability of the system would also be affected due to the high probability of failure which arises from the controller attempting to execute solutions that are unfeasible in practice.

The state space is a key concept in the motion planning problem formulation. This representation extends the concept of robot configuration space [111] —the set of positions that the robot might have in the world—, to consider other key aspects of its status that might be relevant for planning. Thus, when coping with the kinematic restrictions it is necessary to take into account additional information that might influence the robot current and future motions, like the linear and angular speeds. Moreover, in some cases it might be interesting to consider specific variables that depend on the vehicle nature and which affect its maneuverability, like the steering angle in the case of autonomous cars.

Existing approaches successfully addressing motion planning in such high dimensional spaces rely on discrete representations in an attempt to reduce the computational complexity of the problem and obtain solutions in a reasonable time. Most planning algorithms of this kind can be classified as randomized [90, 92, 118] or deterministic [21, 158] strategies. Among the latter, state lattices [158] are noteworthy because they rely on obtaining a set of samples which are arranged following a regular pattern. This strategy demonstrated good sampling quality in commonly used measures like discrepancy [147] and dispersion [191].

State lattices also allow reducing the amount of computation that is performed online. One of the most challenging tasks, especially for systems with complex dynamics, is generating the trajectories that connect the sampled states. The regular arrangement of the lattice opens the door to the obtention of a set of motions that represents the maneuverability of the vehicle as well as possible given its kinematic model. Each one of these actions encapsulates the velocity commands that the robot has to execute to move between two states, in such a way that they can be used to connect every pair of states equally arranged, making them position independent. This way the connectivity of the entire lattice can be represented in terms of a finite set of actions, called motion primitives, leading to a very efficient representation of the robot state and action spaces. In addition to this, a state lattice can be seen like a graph whose vertices are the discrete states and the edges connecting them are actions which, since they were obtained from the motion model, are guaranteed to be feasible. These actions already encapsulate the vehicle kinematic restrictions, for which the constrained motion planning problem can be formulated as a search query that can be solved with an informed graph search algorithm [66, 125, 188], thus making it possible to efficiently find the best path in

the lattice between the initial state and the goal. Even though due to sampling the state space the optimality of the solutions is sacrificed, asymptotical optimality may be retained given the appropriate conditions [84, 90], including heuristic admissibility in the case of combining state lattices and search algorithms.

For many practical applications, from mobile robot autonomous navigation to manipulation, safety and accuracy are of critical importance, and the criteria for selecting the best path not only depends on classical measures like the path length or the traversal time, but on maximizing the probability of successfully completing the mission. In this regard, most real systems are affected by motion uncertainty [112], which might arise from the inaccuracies of the kinematic model with respect to the real behavior of the robot, and from the imperfect state information caused by receiving incomplete or noisy sensor measurements. The amount of uncertainty may be different for each path, since it depends on the executed motions and the quality of the observations in the current robot location. Moreover, unplanned collisions might occur due to the possible deviations that the robot may experience during the execution of the plans. This is shown in the example of Figure 1.2, which represents a couple trajectories with different uncertainties, one of which goes close to an obstacle.

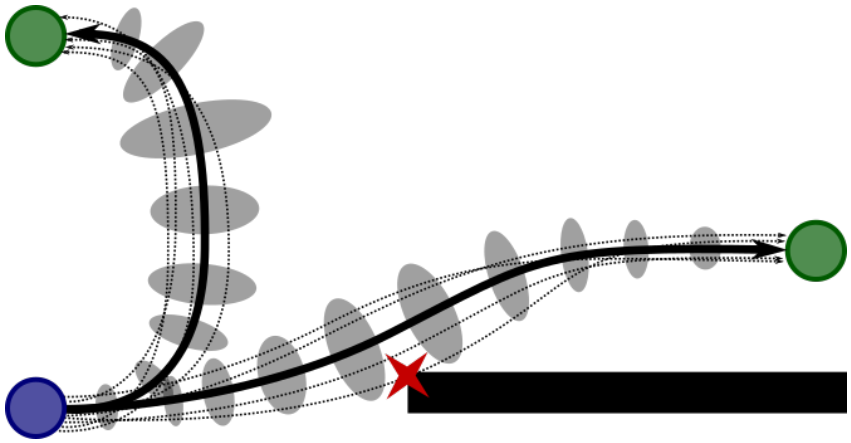


Figure 1.2: Effects of motion uncertainty in the reliability of the planned paths. The initial robot position is in blue, while the goals are in green. The probability distributions representing the robot state uncertainty are shown in gray, while the most probable trajectories are in solid black. The dotted lines represent different executions of the plans, for which the error in the trajectory tracking may be different in each case. The unwanted collisions that might occur due to disregarding the uncertainty at planning time are shown in red.

In this context, taking into account the uncertainty at planning time makes it possible to estimate the amount of motion and sensing uncertainty of each path before the robot starts moving, such that the best plan can be selected accordingly. However, obtaining precise estimations *a priori* in such a way that the problem remains computationally tractable, as well as using this information to determine the suitability of each path reliably, are a couple of issues that remain an open challenge.

Prior research in this field has focused on a variety of aspects: gathering as much information as possible from the sensor measurements during the execution of the paths [26, 163], computing safe control policies [199] or minimizing the probability of collision [6], are just a few examples of the different existing approaches in the literature. Despite the fact that all these are valid strategies, their suitability depends on the specific application and the desired behavior of the system. Thus, since the ultimate goal of an autonomous mobile robot is navigating avoiding obstacles as effectively as possible, explicitly addressing the obtention of paths minimizing the probability of collision in execution time is particularly interesting.

In contrast with those approaches that focus on minimizing the uncertainty, or the classical planners which optimize the traversal time or the total path length, there might be a great number of candidate paths that are suitable for navigation purposes in terms of probability of collision. In this regard, a more precise definition of the outcome that is expected from the motion planner can be achieved if additional criteria for evaluating the cost of the paths are considered. More concretely, the obtention of plans that minimize the probability of collision, as well as the traversal time, would suit well for the purpose of navigating autonomously. The integration of the multiple cost criteria with the strategy of using a search algorithm to find the optimal path in the state lattice can be achieved introducing an order of priority between all the different measures, such that the search algorithm can prioritize the different objectives. Thus, the motion planner can minimize firstly the probability of collision, which directly relates to a safe navigation, and secondly the traversal time, which is a commonly used measure for the optimality of the paths. Other criteria that might be interesting for some practical applications, and that can be used as a complement or in substitution of the mentioned above, are the energy consumption or the amount of uncertainty at the goal.

Regarding the probability of collision, it can be estimated from the Probability Density Functions (PDFs) that are predicted for each path as a previous step, and which represent the uncertainty of the true robot state in execution time. Although prior works have used different strategies to achieve this, most of them disregard the robot real dimensions on behalf

of approximations, like punctual [26, 130] or circular representations [120, 197]. By doing so, they avoid performing complex geometrical operations for the sake of the efficiency, although at the expense of the reliability of the planner. Moreover, using these approximations instead of the robot real dimensions also prevents from considering the effects of the uncertainty in heading. Despite the fact that this is also systematically disregarded in the state of the art, it might be of considerable relevance for robots with asymmetric or irregular shapes —e.g. large robotic platforms used for cargo transport. In this context, in this thesis we propose a method which estimates the probability of collision from the predicted PDFs in such a way that both the uncertainty in heading and the real dimensions of the robot are taken into account. Hence, the obtained solutions adapt to the particular shape that the different vehicles might have, improving the reliability of the results.

An important consideration regarding the performance of a motion planner is the guarantee of completeness, i.e., the capability of retrieving a valid solution, or being able to determine if it does not exist. Among the planning approaches which deal with the kinematic restrictions, completeness is rarely fully guaranteed due to the fact that most strategies rely on sampling the state space with the goal of reducing the computational complexity of the problem. Instead, these approaches offer relaxed completeness guarantees: probabilistic completeness [12, 94], whereby the probability of finding a valid solution increases with the number of samples; or resolution completeness [21], for those strategies which are asymptotically complete when the fidelity of the representation tends to infinite. Since the state lattices are among the second group of strategies, their performance is directly related to the fidelity, i.e., the resolution of the sampled states.

In connection with the foregoing, although increasing the fidelity results in better search capabilities and solutions of a higher quality, this comes at the expense of the planning efficiency. It has been studied that the computational complexity of deterministic sampling methods increases exponentially with the dimensionality of the problem [74, 85]. Thus, it is desirable to decrease the fidelity to the lowest possible to obtain reasonable planning times while retaining the optimality and completeness guarantees. In this regard, the state lattices with graduated fidelity [159] can help to balance the trade off between these two sides by varying the resolution in different areas of the state space, relying on the idea that not all of them have the same interest for solving a given planning query. Therefore, the state space representation could be optimized by decreasing the quality of the representation in selected areas. While some approaches addressed this using high fidelity in the vicinity of the robot

and low fidelity elsewhere [123, 159], thus updating the lattice structure and replanning in execution time, this strategy loses effectivity when dealing with the uncertainty at planning time. This is a direct consequence of having to re-compute the uncertainties and probabilities of collision of all the paths that are affected by the fidelity changes. Moreover, there is no guarantee that the solution will be similar between replannings due to the fact that, unlike other criteria like the path length, the cost due to the uncertainty does not have to monotonically decrease towards the goal.

We address these drawbacks introducing a novel approach for obtaining a graduated fidelity lattice which has a deterministic, consistent and predictable behavior. The number of candidate paths in the state lattice is reduced by grouping the similar motions and using the most suitable representative in each case in accordance with the obstacles in the map, the uncertainty conditions and the maneuverability of the vehicle. Thus, a more efficient representation for planning purposes can be obtained, while the quality of the solutions is retained. More importantly, with this strategy replanning is not required unless the map changes, for which it is suitable for planning under uncertainty.

The selection of the most suitable motion is effectively guided due to using a multi-resolution map as source of information about the obstacle cluttering in the different areas of the environment. This kind of representation is commonly used in many motion planning strategies because of its efficiency. More concretely, octree based maps [69, 134] can manage multiple resolutions in a hierarchical manner, and they are noteworthy for being able to represent large environments with very low computational resources as well. Thus, integrating them in the motion planning approach can help to retain a good scalability with regards to the size of the map.

Despite the relevance of the state space representation, it is not the unique element that plays a significant role in the planning efficiency. In this matter, heuristics are key for reducing the number of states that the search algorithm needs to explore to find the optimal solution. Their informative value is directly related to their performance, for which a heuristic that provides precise estimations leads to lower planning runtimes. However, good heuristics for planning under kinematic constraints are hard to compute and, although different alternatives exist, one approach that proved successful is coping with the vehicle dynamics [100] and the obstacles in the map separately, and then combining the individual estimations [124]. While strategies for pre-computing heuristics of the former kind exist, the latter have to be calculated for every planning query, for which their efficiency is of great relevance to obtain

low planning runtimes. An approach which is commonly used to estimate the cost due to the obstacles is to build a low-dimensional grid, which can be computed in real time for most small environments and bidimensional problems. However, the scalability issues of this operation are similar to those of the general motion planning problem. Hence, the efficiency might be compromised when coping with a high number of dimensions or large environments. Despite this, improving the obtention times of such heuristics is something that has not yet been addressed in the state of the art.

We try to fill this gap by proposing a multi-resolution heuristic that copes with the obstacles in the map and which, instead of a fixed resolution grid, it uses different resolutions throughout the environment in accordance with the information given by the multi-resolution map. In spite of the differences between both methods, this approach compares to the approach for the obtention of the graduated fidelity lattice. In addition to improving the overall planning efficiency, this heuristic also benefits the scalability in large environments.

As result of all these improvements, the efficiency of the motion planning approach we propose makes it possible its integration in real systems. In this context, its applicability to mobile robots of different kinds can be achieved almost effortlessly due to the fact that the only vehicle-specific component is the set of motion primitives used for generating the connectivity of the state lattice. Moreover, the motion planner prioritizes the minimization of the probability of collision, which is reliably estimated in such a way that takes into account the robot real dimensions and the predicted uncertainty, also in heading. All of the above makes this approach suitable for navigating autonomously in a safe manner, which inevitably forms part of many robotic tasks.

1.1.1 Unmanned Aerial Vehicles (UAVs)

In addition to the increasing attention that mobile robots have received in the last years, the raise in popularity of the Unmanned Aerial Vehicles (UAVs) has been particularly significant due to the hardware advances that have opened the door to more and more practical applications. Although under the denomination of UAVs different kinds of aerial vehicles are encompassed, this thesis focuses on multi-rotor systems like that shown in Figure 1.3. Their motion capabilities allow them to execute maneuvers that require a certain level of precision, and they have enough flexibility to operate in challenging environments.

The widely extended American neologism “the Three Ds” —dull, dirty and dangerous— accurately describes the kind of operations that are more interesting for having some degree

Figure 1.3: The AscTec Pelican, manufactured by Ascending Technologies, is an example of multi-rotor UAV, which is capable of executing a great variety of precise motions and was used in some practical experiments of this thesis.

of automatization with the goal of avoiding human intervention as much as possible. A first step in this direction can be achieved introducing manually controlled platforms that prevent human workers from doing certain kinds of tasks. Thus, in the case of UAVs a human operator is in charge of supervising and usually controlling the flight. However, the possibility or the suitability of manual control may be in some cases limited, for example due to the loss of communication with the aerial vehicle or the high precision that is required for executing some maneuvers, which might be outside the pilot capabilities —especially if having direct sight to the UAV is not possible. In these cases, the benefit of being able to navigate autonomously is clear when it comes to help the operator to accomplish certain missions.

Dull tasks are characterized for being repetitive and tedious. Some examples include exploration [172], surveillance [184], traffic monitoring [34, 164] or the reconstruction of 3D scenarios [146]. These kind of missions usually include executing several routes and hovering for a certain amount of time while gathering information with an on-board camera. In this context, navigating autonomously would be interesting because the operators would only be in charge of analyzing the received images and their intervention would be limited to taking control of the UAV to address situations that require any further actions, if necessary.

Any unsanitary or hazardous job is named as *dirty*. Examples of tasks of this kind would include operating the UAV in circumstances that might affect human health, like forest fire detection and tracking [209]. Similarly, addressing gas leaking detection in pipelines [173, 186], or spray painting in inaccessible 3D structures [200] would avoid the exposure to toxic substances.

The interest of being able to navigate autonomously is even clearer in other contexts in which human intervention is too hazardous. These are the so called *dangerous* tasks, among which infrastructure inspection is noteworthy because it has to be done in a routine manner either by human workers themselves, or helped by a variety of robotic systems [5]. In this regard, the benefit of introducing UAVs is clear because they prevent humans from attempting to access areas that might be unsafe. Moreover, they can be used to collect high fidelity visual and depth data and use this information to raise alerts when issues are identified. Due to the great variety of sensors that UAVs can carry, they can gather extensive information about the status of the infrastructure that might outperform visual inspection, like thermal or infrared imaging. In this context, UAVs have been applied to avoid human intervention as far as possible in bridge [31, 207] or wind turbine inspection [189, 204]. Post disaster assessment and management [40, 43] is another interesting scenario which poses particular challenges in regards to the safety of the UAV during the flight, especially in environments where the operator cannot easily have direct sight to the aerial vehicle.

Finally, another practical application which has received an increasing attention in the last years is the usage of UAVs to deliver goods, and many companies have launched different proposals in this sense, like Amazon [82], DHL [57], Google [81] or Swiss Post [4]. Some of these approaches have focused so far on the delivery of small medical supplies [4, 57, 192]. Such is the interest in this practical application that some administrations have their own proposals in this sense. For example, the Regional Government of Galicia will launch a system to deliver defibrillators with UAVs for the rapid response to cases of cardiorespiratory arrest in the Way of St. James [35].

UAVs are a clear example of system with complex dynamics, for which taking into account their kinematic constraints when planning routes for flying autonomously becomes especially important. More importantly, in all the contexts mentioned above it is particularly relevant to have a motion planner that is able to retrieve solutions as reliably as possible.

In addition to this, the efficiency of the approach is equally important to make it possible its integration in practical applications, like those mentioned before. In this regard, the motion

planning strategy we propose in this thesis is able to deal with the UAV severe kinematic constraints and produce feasible paths. The state lattice based planning strategy achieves this very efficiently, since the UAV dynamics are encoded in the motion primitives that connect the sampled states. Moreover, by taking into account the uncertainty at planning time, the inaccuracies that might occur as a result of the differences between the dynamics model and the UAV real behavior are taken into account. Despite the fact that this model can be obtained from real navigation data that accurately represents the UAV real behavior to most control commands, significant deviations might still occur due to unmodeled disturbances.

The method we propose also copes with the imprecisions or the temporary unavailability of the information about the UAV own state. For example, outdoor missions usually rely on satellite based localization systems, like the Global Positioning System (GPS), which may be not accessible in certain areas of the environment like in the proximity of buildings, below bridges, etc. Instead, in indoor environments other localization approaches are used, e.g., motion capture systems, or Simultaneous Localization and Mapping (SLAM) algorithms which use the information of an on-board camera to estimate the UAV motions [46, 121, 122], although these may suffer from similar coverage issues. In this connection, sensor fusion techniques can be used in combination with all the mentioned above methods for the sake of the precision and robustness of the results. Despite the differences between all these strategies, the motion planning approach we propose adapts to the uncertainty of each case, and it is able to cope with partial or missing observations as well.

Due to the high computational complexity of the problem, prior works in the field of motion planning for UAVs made simplifications of different natures —in most cases disregarding the uncertainty and sometimes the kinematic restrictions as well— to guarantee low planning runtimes. However, the efficiency improvements we propose in this thesis make it possible to obtain plans fast enough to allow operating the UAV without noticeable delays while dealing with the uncertainty conditions and kinematic restrictions described above.

1.2 Objectives

The main goal of this thesis is **to develop a reliable and efficient motion planning approach for the autonomous navigation of mobile robots in general and, more specifically, UAVs**. This general objective is divided into these specific ones:

O1. To develop a reliable motion planning approach capable of retrieving optimal paths in

terms of safety and traversal time and deal with the vehicle kinematic constraints and the motion and sensing uncertainty at planning time.

O2. To improve the efficiency of the motion planning algorithms by introducing multi-resolution techniques in order to reduce the computational complexity of the problem and analyze their impact in the planning results.

O3. To integrate the developed algorithms into an architecture for the autonomous navigation of mobile robots and UAVs and validate the proposed system with real experiments in different platforms.

1.3 Contributions

The research developed in this thesis has led to the following contributions:

C1. A deterministic method for estimating the probability of collision of the paths, given the robot state predicted uncertainty, that takes into account the vehicle real shape and also deals with the uncertainty in heading.

C2. A novel strategy for obtaining a graduated fidelity lattice, which adapts to the obstacles in the map and the robot maneuverability, that significantly improves the efficiency of the motion planner while retaining its performance.

C3. An efficient and scalable heuristic that relies on a multi-resolution, low dimensional grid to estimate the cost from the current robot state to the goal taking into account the obstacles.

C4. A motion planning approach adapted to the requirements of the autonomous navigation of UAVs which deals with the motion and sensing uncertainty and obtains optimal paths prioritizing their safety.

C5. The application of the developed motion planning algorithms for the obtention of plans for the autonomous reconstruction of 3D scenes with a RGB-D camera.

Publications

All the contributions detailed above are included in the following publications, which constitute the scientific output of the research developed during the development of this thesis:

Journals

- A. González-Sieira, M. Mucientes and A. Bugarín, *Motion Planning under Uncertainty in Graduated Fidelity Lattices*. Robotics and Autonomous Systems, vol. 109, pp. 168-182, 2018.

  Impact factor: 2.928 (JCR 2018); 0.83 (SJR 2018).

  Journal ranked Q1 in SJR 2018, in categories: Computer Science Applications (136/553), Control and Systems Engineering (59/222).

- A. González-Sieira, M. Mucientes and A. Bugarín, *Autonomous navigation for UAVs managing motion and sensing uncertainty*. Article submitted to Robotics And Autonomous Systems in September, 2019.

  Impact factor: 2.928 (JCR 2018); 0.83 (SJR 2018).

  Journal ranked Q1 in SJR 2018, in categories: Computer Science Applications (136/553), Control and Systems Engineering (59/222).

Conferences

- A. González-Sieira, M. Mucientes and A. Bugarín. *Graduated Fidelity Lattices for Motion Planning under Uncertainty*. In proceedings of the 2019 IEEE International Conference on Robotics and Automation (ICRA), pp. 5908-5914, Montreal (Canada), 2019.

  Conference ranking (GRIN-SCIE): A, Class 2.

- A. González-Sieira, M. Mucientes and A. Bugarín. *An Adaptive Multi-resolution State Lattice Approach for Motion Planning with Uncertainty*. In proceedings of the Second Iberian Robotics Conference (ROBOT2015), pp. 257-268, Lisbon (Portugal), 2015.

- P. Rodríguez-Mier, A. González-Sieira, M. Mucientes, M. Lama and A. Bugarín. *Hipster: An Open Source Java Library for Heuristic Search*. In proceedings of the 9th Iberian Conference on Information Systems and Technologies (CISTI2014), pp. 481-486, Barcelona (Spain), 2014.

- A. González-Sieira, M. Mucientes and A. Bugarín. *A State Lattice Approach for Motion Planning under Control and Sensor Uncertainty*. In proceedings of the First Iberian Robotics Conference (ROBOT2013), pp. 247-260, Madrid (Spain), 2013.

- A. González-Sieira, M. Mucientes and A. Bugarín. *Anytime Motion Replanning in State Lattices for Wheeled Robots*. In Proceedings of the XIII Workshop of Physical Agents 2012, pp. 217-224, Santiago de Compostela (Spain), 2012.

1.4 Dissertation structure

This dissertation is divided into five chapters that analyze the related prior work, describe in detail the contributions and experimental results, and present the conclusions of this research. More concretely, the document structure is as follows:

- First, in Chapter 2 the different approaches that addressed the motion planning problem are analyzed. This includes a review of the existing techniques for the generation of trajectories that observe the vehicle kinematic constraints, the algorithms that obtain feasible and optimal global plans, and the different methods that deal with motion planning under uncertainty. Prior research that focused on the autonomous navigation of UAVs is also analyzed.

- Chapter 3 describes the proposed motion planning approach for obtaining optimal paths in terms of probability of collision and traversal time, detailing the contributions regarding the reliable estimation of the safety of the paths —contribution C1— and the efficiency improvements —contributions C2 and C3— that result in reasonable planning times, making it possible the integration of the planner in real systems.

- Chapter 4 details the new algorithms for the obtention of plans for the autonomous navigation of UAVs —contribution C4. Furthermore, results for the experiments conducted on a real platform are reported. This includes the generation of trajectories for a real application: the autonomous reconstruction of 3D scenes —contribution C5.

- Finally, in Chapter 5 the contributions of the research developed in this thesis are summarized. Moreover, this chapter discusses future lines of work that could allow improving the results, and also the possibility of extending the developed motion planning algorithms to take into account the restrictions due to navigating in social environments, making it possible for the robots to have a more "human-like" behavior when navigating near people.

Chapter 2

Related work

The problem of finding optimal and collision-free paths for mobile robots has received significant attention from the research community [112, 114, 119]. Constant developments have been achieved in the motion planning field since the first solutions to the problem were proposed in the late 1960's [182], especially after the notable contributions that arised in the 1980's [78]. As a result of the increasing interest in autonomous vehicles [59, 153], subsequent advances in this field have focused on planning taking into account the kinematic restrictions, which opened the door to the obtention of solutions adapted to the motion capabilities of the different robotic platforms. Thus, nowadays mobile robots can navigate dealing with obstacle avoidance in a great variety of situations. However, in many real world applications safety and accuracy are critical, and motion uncertainty influences on which path is best. This raised new challenges and opened new lines of research, among which dealing with the uncertainty at planning time stands out for its relevance.

The most recent advances in this field take into account the uncertainty due to the inaccuracies in the controls and the noisy or missing measurements from the sensors, which makes it possible to obtain solutions that minimize the probability of collision of the robot during the execution of the plans. In this sense, estimating the safety of the paths in a realistic manner, taking into account as much information as possible about the robot and its predicted uncertainty, is key for achieving a reliable motion planning approach. However, the runtimes of these strategies are significantly higher due to the increased overhead which is caused by having to estimate the probabilities of collision of all the candidate paths, which is a major concern that needs to be addressed for integrating them in real systems. Despite the constant

hardware improvements that have allowed a significant increase of the computing power in the last years, the efficiency of the motion planning algorithms is essential for the obtention of reasonable runtimes that make it possible to react to changes in the environment and, additionally, to increase the amount of resources that are left available to other components of the robotic system. Part of the research conducted in this direction was related to the heuristics that are used for estimating the cost of the paths, while others focused on obtaining more lightweight representations of the robot state and action spaces.

The autonomous navigation of mobile robots is a challenging task due to a number of factors which include mapping, localization and path planning for avoiding obstacles in complex environments. The safety requirements of this task are high and demand reliable motion plans which, at the same time, have to be efficiently obtained to avoid noticeable waiting times before the robot can start moving. In this sense, managing the trade off between planning efficiency and performance is key for satisfying the restrictions on these both ends. The autonomous navigation of UAVs presents a number of additional challenges due to the impact that the dynamics model and the motion uncertainty have in the accuracy of the executions of the obtained plans. Moreover, the increased problem dimensionality due to planning in 3D space increments the computational complexity in a significant manner.

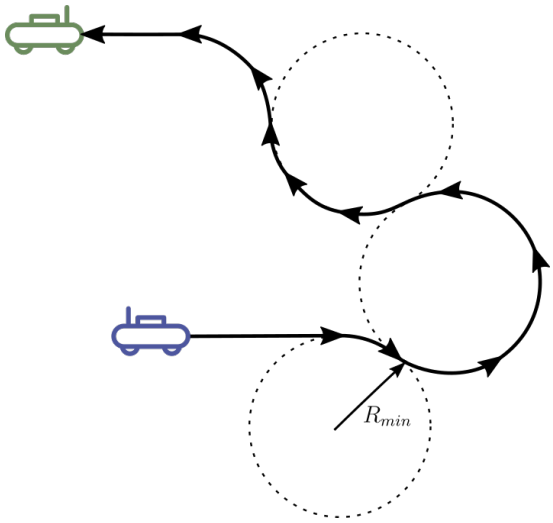
This chapter reviews in detail prior work related to the topics of this thesis, and it is organized in six sections. In Section 2.1 the different methods for generating trajectories for mobile robots observing the vehicle kinematic constraints are analyzed. Section 2.2 reviews the existing approaches that addressed motion planning, focusing on the state space representation, their main features and limitations. The obtention of optimal solutions for the motion planning problem is addressed in Section 2.3, where the different strategies to obtain paths with the lowest cost are detailed. Section 2.4 reviews the approaches for planning taking into account the motion and sensing uncertainty, focusing on their suitability for dealing with the different kinds of inaccuracies. The different methods for addressing the high computational complexity of planning in high dimensional spaces and the increased overhead of dealing with complex systems are described in Section 2.5. Finally, Section 2.6 focuses on prior work that addressed the autonomous navigation of UAVs, a motion planning problem which is challenging due to the complex dynamics of this kind of systems.

2.1 Planning with kinematic constraints

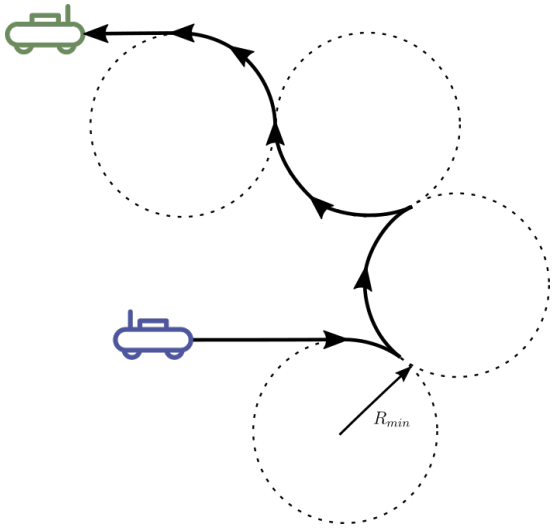
The generation of trajectories that accurately represent the motion of a robot taking into account its kinematic constraints is a necessary step to obtain suitable solutions for motion planning. In this sense, a significant amount of research has been done to efficiently find trajectories that observe these restrictions. The problem of obtaining a collision free path between two points in the free space is especially challenging for nonholonomic vehicles, whose trajectories were first analyzed by Dubins [39] —which described a model for a car that only moves forward— and Reeds and Shepp [167] —which addressed forward and backwards motions. These models limited the vehicle steering angle, resulting in constant curvature motions, as shown in Figure 2.1. However, their research opened the door to further developments in the characterization of nonholonomic trajectories, which allowed the precision of the models used as basis for trajectory generation to improve.

In this sense, a more flexible representation for these kind of paths was introduced in [179] for the Dubins car, and in [48] for the Reeds and Shepp case. Instead of considering curvature constrained paths, their models combined straight lines, arcs and clothoids —segments with curvatures which are function of their length— to obtain trajectories with continuous curvature profiles. These improvements in the representation of the vehicle motions leaded to the obtention of smoother trajectories and to a lower error in tracking. At the same time, the issue of needing to stop the robot for redirecting the wheels with every change in the curvature of the path was overcome. However, these advantages came at the expense of the efficiency due to the lack of a closed form solution for the clothoids. For this reason, further research developed in this direction [49, 110, 180] focused on improving the flexibility and the efficiency of these representations.

In addition to coping with the robot kinematic restrictions, taking into account the obstacles in the environment is essential for obtaining suitable paths for the autonomous navigation of mobile robots. In this sense, despite being local planning solutions, some mentioned above methods were combined with higher level strategies [92, 113, 132] with the goal of obtaining paths that were suitable for navigating autonomously in environments with complex obstacle configurations. Most of the resulting approaches relied on building a roadmap, using the local planner to connect the points that were distributed throughout the free space. In this regard, in [179] the authors proposed first obtaining a set of simple candidate paths, and then selecting as local planning solution the shortest collision free path. Conversely, in [48, 49, 180] the collisions were directly checked when the roadmap was built. However, since clothoids are not



(a) Dubins model.



(b) Reeds and Shepp model.

Figure 2.1: Examples of trajectories for a nonholonomic system. The initial position is in blue, while the goal is in green. The path for moving between them is highlighted in black. The dotted circles represent the minimum curvature allowed (R_{min}).

analytic curves, they approximated the area swept by the robot using polygons whose edges aligned with the initial and final poses of each motion. Instead of using a roadmap, the approach of [110] obtained a collision free path disregarding the vehicle kinematic constraints, and then calculated the feasible path which better approximated that solution. Despite this, their trajectory generation method can be combined with the roadmap approach as well.

In some cases trajectories might be computed analytically [9, 27], e.g. for systems with linear dynamics and quadratic cost [50]. The latter approach represented the robot dynamics with differential equations, thus decoupling the path planning and the low-level control. Moreover, this formulation also opened the door to the obtention of trajectories which minimize other criteria than the traversal time, such as the path length or the energy required by the controls. In other cases, although closed-form solutions are not available, nearly optimal ones can be obtained using numerical optimization, given the problem constraints [7]: the vehicle kinematics, the initial and final configuration, the velocity restrictions, and the cost function. Following this direction of research, in [70, 96] they described an algorithm for the generation of trajectories with a high level of efficiency and generality. The efficiency of this strategy relies on using a fast numerical optimization technique, the Newton's method, while the generality arises from parametrizing the vehicle controls, linearizing and inverting the different models that represent the vehicle behavior. Thus, this method can accommodate complex effects such as inertias or wheel-terrain interaction while maintaining very low computation times.

In order to generate trajectories as realistically as possible, the robot behavior with respect to the different controls should be considered. However, the classical motion equations do not take into account some physical effects that might be crucial for the obtention of paths for mobile robots with complex dynamics. In this sense, [1, 2] described an algorithm for the obtention of a six degrees of freedom model from real navigation data. Their model is based on parametrizing the vehicle accelerations and it focuses not only on one-step transitions —which would only explain the inmediate robot response to the controls—, but also on minimizing the long-term prediction error. Unlike other approaches, e.g. the industry standard CIFER® [136, 137, 193], the acceleration based model is able to capture the nonlinear effects which depend on both the linear and angular rates, such as the inertias. This is particularly relevant for mobile robots with complex dynamics, e.g. UAVs, in which important deviations from the planned trajectories might occur if the robot real response to the controls is disregarded, like in other prior works [10, 99, 138]. Although all the mentioned approaches

focused on single-rotor UAVs, they can be generally applied to other kinds of small scale rotorcraft UAVs [28] —e.g. multi-rotor aerial vehicles.

All these approaches addressed the generation of trajectories for different kinds of mobile robots considering their kinematic constraints. This problem is of great relevance for guaranteeing that the obtained plans are feasible and can be tracked with a certain degree of accuracy. Moreover, the development of these techniques has made it possible to cope with increasingly complex dynamics models, as well as to consider other physical effects like the vehicle inertia. Despite the good results obtained by these techniques, they focused on local planning, and therefore they cannot deal with complex obstacle configurations. However, their integration with other approaches that will be analyzed later in this chapter makes it possible to obtain global solutions retaining these properties. In this regard, their efficiency improvements have been essential for achieving reasonable runtimes.

2.2 Approaches to motion planning

The motion planning problem consists in finding the appropriate controls to drive the robot from its current state to a desired goal. It was formulated in 1979 [168] and shown to be a PSPACE-hard problem. That formulation was restricted to polyhedral obstacles and a finite number of robot bodies which were attached by spherical joints. Later in [30] it was demonstrated that the problem formulation lied in PSPACE. Thus, the general motion planning problem is PSPACE-complete [115]. Generally speaking, in order to find suitable plans it is essential to take into account both the robot dynamics —e.g. the kinematic and holonomic restrictions— and the obstacles in the environment. This has received an important research effort and has been addressed in the literature following approaches of different kinds.

The methods based on potential fields [13, 79, 97, 98] rely on building a collision free trajectory by calculating the negative gradient of a potential function which simulates repulsive forces originated in the obstacles to push the robot away from them and attractive forces to drive it to the goal. Instead of computing the entire trajectory to the goal, the potential field based methods act as a feedback control policy and obtain the immediate control from the current robot state. While the efficiency of this kind of methods makes them suitable for working in real time, their main drawback is their lack of robustness to local minima [102]. Some efforts in extending the applicability of these methods was made introducing randomization [14] or artificial potential fields [169], which act as navigation functions. The applicability

of the latter is limited, since computing this function for the general case is as computationally expensive as solving the motion planning problem considering all the constraints [64]. Notwithstanding, due to explicitly representing the occupancy information, this kind of methods scale poorly with the dimensionality of the problem or the number of obstacles.

Another family of approaches relies on the Fast Marching (FM) method [185], which finds an approximate solution to the equations that describe wave propagation. On the one hand, there is a wavefront arrival time function for every point of the map and, on the other hand, another function that describes the wave propagation velocity. A reasonably good solution is then found combining both functions with a gradient descent method. Despite the efficiency of the original FM method, the robot kinematic restrictions and the obtained paths are non smooth. Moreover, the solutions run too close to the obstacles, making them impractical for most applications.

Different improvements have been proposed to address these issues, among which applying a repulsive potential to the obstacles [97], following maximum clearance maps [55, 56], or the Fast Marching Square (FM^2) algorithm [54] —which consists in applying the FM method twice to obtain smooth paths—, stand out for their relevance. However, due to disregarding the kinematic restrictions the feasibility of the obtained paths is not guaranteed, it depends on how the algorithm parameters are tuned. Moreover, further research is still needed to deal with the unseen areas of the map or the presence of small obstacles which might affect the suitability of the obtained paths.

A global motion planning strategy retrieves the entire path connecting the beginning and goal configurations. In this regard, an approach is considered complete when it only fails retrieving a solution if such path does not exist. While there were efforts in obtaining complete planning algorithms [30, 129, 181], due to the computational complexity of the problem itself these kind of approaches are not suitable in practice. Instead, other methods that relaxed the completeness requirements to achieve more reasonable planning times were developed.

One family of approaches which demonstrated their performance obtaining reasonable planning times are the resolution complete approaches. They provide guarantees of retrieving a valid solution if the resolution parameters of the planning algorithms are properly set. A motion planning approach which is a good representative of this family of methods is based on cell decomposition [24, 25]. It relies on partitioning the robot state space in a finite number of regions in which collision free paths can be easily obtained, and then obtaining planning solutions as sequences of adjacent cells. Despite the good performance and reasonable plan-

ning times of this method, even in complex environments, its suitability for high dimensional planning spaces is limited because of its scalability issues.

As mentioned in Section 2.1, despite being local solutions, some of the analyzed methods for mobile robot trajectory generation reported global planning results integrating their approaches with higher level planning techniques. In this regard, randomized sampling methods for motion planning —introduced in the 1990s— have proven successful in dealing with robots with many degrees of freedom. The Probabilistic Roadmap (PRM) framework, a two-phase approach that allows solving multi-query motion planning problems, was introduced in [91, 152], being later refined in [72, 92, 95, 114]. In a first stage a roadmap which represents the different trajectories that the robot can execute in the environment is built, as shown in Figure 2.2. This is achieved sampling the free space in a stochastic manner, connecting each new sample with the already existing ones —within a distance threshold— using a local motion planner and checking collisions with the obstacles. In a second phase, this roadmap is queried and feasible paths between any two configurations belonging to it are retrieved very fast. Additional plans may be required to connect the beginning and the goal points to the nearest nodes in the roadmap. In [73] they presented a new incremental algorithm for building a roadmap which guaranteed a better performance than the classical PRM method. However, the authors pointed out some difficulties due to the fact that the performance of the method relied on the uniform sampling of the free space, while the existing implementations had relied on sampling the robot control inputs uniformly, which does not guarantee a uniform sampling of the state space, as noted in [51].

Although it is valuable for highly structured environments, e.g. factories, for many other practical applications building a roadmap for the entire map is not so interesting. The cost of this operation might be significant, especially for large environments, and it can vary depending on the efficiency and performance of the local planning method. For this reason, another kind of approaches that focused on single-query problems arised, avoiding the high cost of computing the plans for connecting all configurations distributed throughout the free space.

In this sense, in [71, 75, 76] the authors presented a bidirectional randomized planner which relied on growing two search trees simultaneously, one starting in the beginning and another starting in the goal, until the explored regions intersected with each other. But perhaps the most noteworthy approach of this family of algorithms is the Rapidly Exploring Random Tree —RRT—, introduced in [86, 116, 118, 119]. That approach relied on building a structure comprised by feasible trajectories that connected randomly generated configurations, an idea

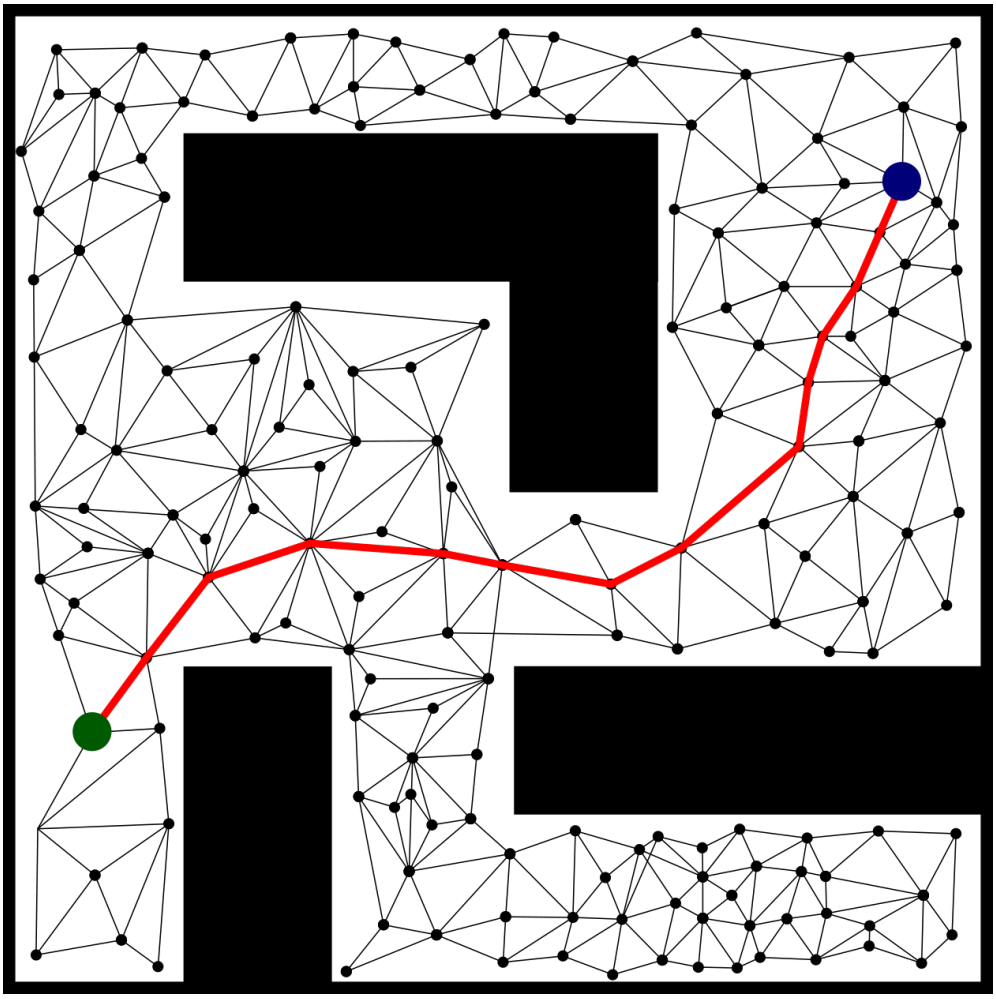


Figure 2.2: Structure of a Probabilistic Roadmap (PRM). The sampled configurations are represented as circles, while the lines are the paths connecting them. The start is shown in green, while the goal is in blue. The solution for the planning query is highlighted in red.

which is similar to the PRM approach. However, both algorithms construct the graph representing the robot trajectories in a different manner. RRT is a single-query approach which, unlike PRM, avoids exhaustively exploring the robot state space and focuses only on computing the possible trajectories from a single point, as shown in Figure 2.3. When a new configuration is added to the graph, RRT connects it only with its nearest neighbor in the search tree. The cause beyond the rapid exploration of RRT is its Voronoi bias [127]. Due to the way in which the state space is explored, the probability that a node is selected goes in accordance with the volume of its Voronoi region. Thus, the search is biased towards regions of the state space which have lower levels of exploration, and strategies to take advantage of that bias and improve the planning efficiency were developed [128, 206].

The efficiency of randomized planners like RRT and PRM relies on avoiding to explicitly represent the robot state space as such. Despite not being complete approaches, they have proven to be probabilistically complete [12, 14, 75, 94, 108]. This concept describes those planners whose probability of failing in retrieving a valid solution if it exists decays to zero when the number of samples tends to infinity. But more importantly, the decay of the probability of failure is exponential under the assumption of good visibility properties of the environment free space [12]. In that work these visibility properties were characterized under the concept of expansiveness, that is tied to the rate at which the set of samples successfully connected to the roadmap grows with its size. It also measures the difficulty of sampling in multi-dimensional narrow passages —see [76]— and also captures the issue of using randomized planners under these conditions, already studied in [72]. The study conducted in [74] revealed that this assumption was reasonable for most practical applications, and tied this to the empirical success of randomized algorithms.

Due to the increasing popularity of randomized motion planners, the research community also focused on studying the different available techniques for sampling the state space. In this regard, two concepts that are commonly used for evaluating such techniques are the discrepancy and dispersion. Generally speaking, the lower the discrepancy of a sequence, the more proportional the number of samples that fall into an arbitrary volume to its size. On the other hand, the lower the dispersion of the samples, the smaller the maximum area which is left uncovered by them. Prior research in [21] showed that similar or improved performance can be achieved with quasi-random sampling techniques due to the more uniform sampling that is attributed to this kind of methods. Moreover, they introduced a couple of sampling techniques which were used for planning purposes: one based on low discrepancy sequences,

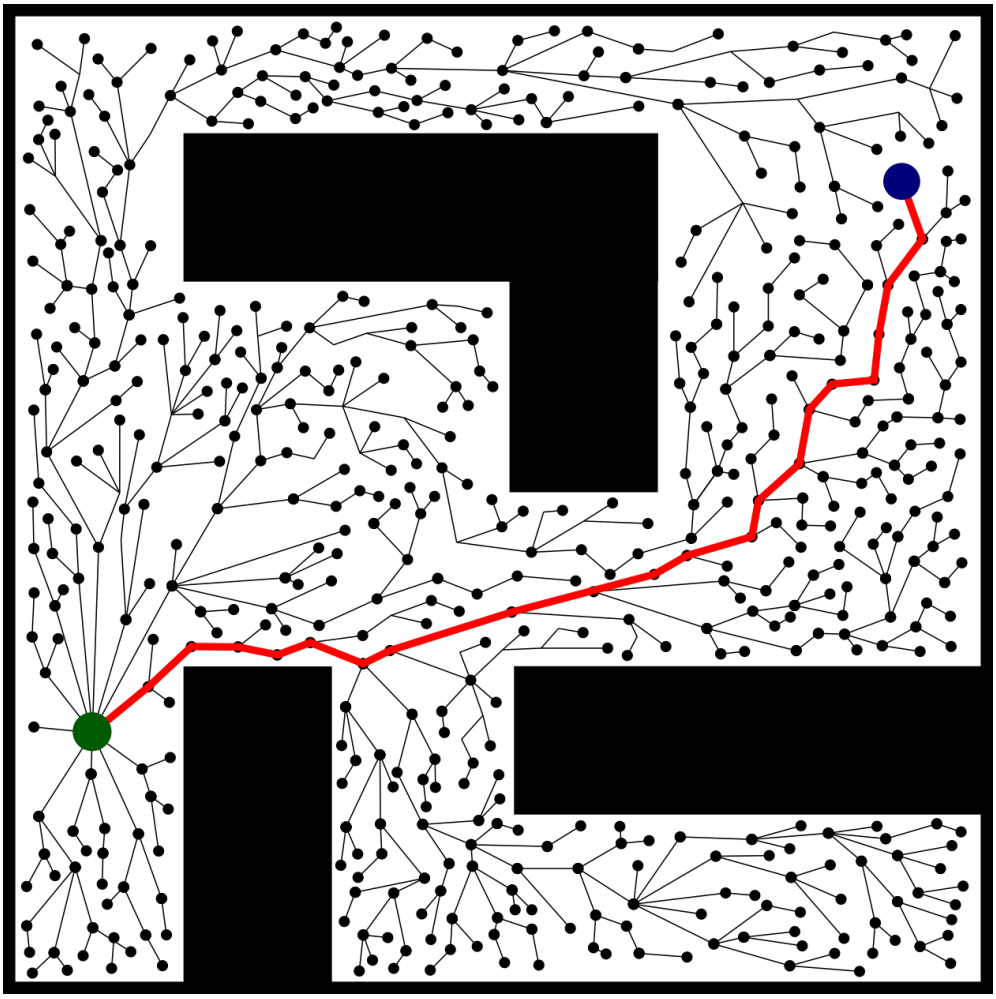


Figure 2.3: Structure of a Rapidly-Exploring Random Tree (RRT). The sampled configurations are represented as circles, while the lines are the paths connecting them. Unlike PRM, RRT computes only the paths from the starting point, shown in green. The goal is in blue, and the path connecting them is highlighted in red.

named Quasi-PRM, and another based on regular lattices, the Lattice Roadmap (LRM). These two methods are resolution complete, and the main advantages of LRM are its properties of near-optimal discrepancy and optimal dispersion. More recent studies have shown that the efficiency of randomized planning approaches like PRM and RRT is not due to the stochasticity itself [117]. In fact, in that work the authors pointed out that the obtention of stochastic samples via random number generators has a certain degree of determinism.

Another line of research focused on proposing lazy variants of the existing planning algorithms based on roadmaps which avoided checking collisions during the construction phase [18, 19, 177, 178]. This way the collision check operations were left to the query phase and the obtained roadmaps could be re-used with other configurations, such as robots with different dimensions or changes in the obstacle locations. In this regard, [21] suggested that having an initial representation of the possible robot motions —e.g. a roadmap, a tree or a graph— was not even necessary since its construction could be done during the search.

All the planning techniques we analyzed in this section addressed the problem of finding a global solution to drive the robot from an initial state to another one. However, there are notable differences in the way they achieve that. The high complexity of the problem has motivated the development of sampling based methods in order to efficiently deal with high dimensional search spaces. In this regard, the existing techniques can be divided into those which compute a roadmap for the whole environment, allowing to solve multi-query problems, and those which build a search tree and solve a single query instead. Another way of classifying prior works could be by the method used for obtaining the sampled states. While randomized techniques have proven their suitability for dealing with problems of different natures, when it comes to comparing the sampling properties the techniques which are quasi-random or deterministic have better dispersion and discrepancy.

2.3 Optimal motion planning

In most robotic applications, obtaining a good solution in terms of a measurable cost —e.g. the length of the path or the traversal time— is highly relevant. However, computing optimal motion plans is challenging even for the most basic problems [30], and despite the fact that some authors pointed out the relevance of this topic [119], the most noteworthy research efforts in this direction started in the mid-2000s. Although sampling the state space comes at the expense of sacrificing the optimality of the planned paths, asymptotic optimality may be

retained under certain conditions [84, 90]. In this regard, in the literature there are approaches both based on randomized algorithms and deterministic sampling in combination with graph search algorithms.

One of the first randomized approaches is the Heuristically-guided RRT (hRRT) [196], which addressed the generation of low cost paths focusing on biasing the RRT growth using heuristics. Such behavior was achieved weighting the probability density function used for obtaining the random samples in accordance with the value of the heuristic. Thus, the trees obtained by hRRT are biased towards the low cost paths. In their experiments they also compared the performance of different heuristics in terms of the cost of the obtained solutions. In [45] running RRT several times for the same planning problem was proposed. They showed that, despite not having optimality convergence guarantees, this method allowed the cost of the solutions to decrease with every new execution. In this regard, [205] studied the performance of RRT in problems of different nature and analyzed the convenience of restarting the algorithm for achieving a reduction in the cost of the solutions. Although it is not applicable to all kind of problems, their results showed the tendency for restarts to be more useful in highly difficult planning problems.

The different approach of T-RRT, Transition-based RRT [83], combined the rapid exploration properties of RRT with stochastic optimization methods, which use transition tests to determine the acceptance or not of a new potential state. The difficulty of these tests was adapted on-line depending on the failure rate during the search process in order control its exploratory behavior. However, since the exploration of T-RRT only relied on sampling the state space, it had difficulties to deal with cost functions characterized by having narrow low cost areas surrounded by others with higher costs. This was addressed in [16], which combined T-RRT with a gradient function to bias the exploration towards the lower cost regions of the state space. However, despite the good results obtained, the applicability of these two methods is limited to planning problems with continuous cost functions.

Perhaps one of the most important contributions in the obtention of optimal paths using randomized algorithms is the research presented in [90]. The authors first provided an extensive analysis of the optimality and complexity of the different variants of PRM and RRT, showing that of all existing algorithms of this kind, only the simplified version of PRM — Simple PRM (sPRM) [93, 94]— offers asymptotically optimal guarantees, but at the expense of a high computational complexity. Apart from their valuable analysis, in their research the authors also proposed several algorithms: the Optimal Probabilistic Roadmap (PRM*), the

Rapidly Exploring Random Graph (RRG) and the Optimal Rapidly Exploring Random Tree (RRT*); which they proven to be asymptotically optimal, probabilistic complete and efficient in terms of computational complexity.

PRM* is a variant of PRM which uses a distance threshold that depends on the dimensionality of the problem and decays with the number of samples to generate an efficient representation of the robot possible motions in the free space. RRG incrementally builds a roadmap in a similar manner to RRT, with the main difference that RRT only attempts to connect the new samples to the nearest existing ones, and RRG also attempts to connect other nodes within a distance threshold if the first attempt was successful. Thus, it is clear that RRG is a version of RRT with an improved connectivity, since for the same sampling sequence the result of the RRT graph is a subset of the one obtained by RRG. Moreover, while the first is a directed tree, the second is an undirected graph that might contain cycles.

Finally, RRT* is similar to RRG but it maintains the structure of a tree, which results in improved memory requirements. This is achieved limiting the increased connectivity of RRG by removing the cycles caused by redundant edges —those that do not belong to the shortest path between each vertex and the root of the tree. This representation is more suitable for some practical applications and it is more easily extensible to problems with differential constraints. The trees obtained by RRT* share with those of RRT the same structure of vertices, although they have different edge sets. In the former it is guaranteed that each node is connected to the root of the tree by a minimum cost path.

2.3.1 State lattices

State lattices, introduced by [158], are noteworthy among the approaches based on deterministic sampling for their efficient representation of the state space due to using a regular discretization scheme. Their regularity comes with substantial benefits like optimal dispersion and low discrepancy [115]. Like LRM [21], state lattices are a resolution complete approach, since increasing the fidelity —the resolution of the sampled states— makes the number of samples to increase as well, such that for sufficiently high values the state space representation approaches the continuum. In fact, state lattices can be seen as an extension of LRM which also allows coping with the robot motion constraints. This is because the state lattice resembles a graph in which the sampled states are connected by feasible motions of the robot, so that these constraints are by construction encoded in its structure.

While building the state lattice might seem computationally expensive, the regular sam-

pling strategy has the benefit of providing translational invariance, as shown in Figure 2.4. Thus, the same motion can be used to connect every pair of states equally arranged. This is extensible to the low level controls that allow executing each motion, which are the same for all instances regardless their initial position. This way the connectivity of the state lattice can be pre-computed offline and stored in terms of a canonical set of motions —the canonical control set, or motion primitives—, which constitutes one of the main advantages of this representation. These actions are obtained from the vehicle motion model, e.g. using an inverse trajectory generator —see Section 2.1. Moreover, unlike in randomized approaches, the unfeasible motions are already discarded when the motion primitives are built. For all these

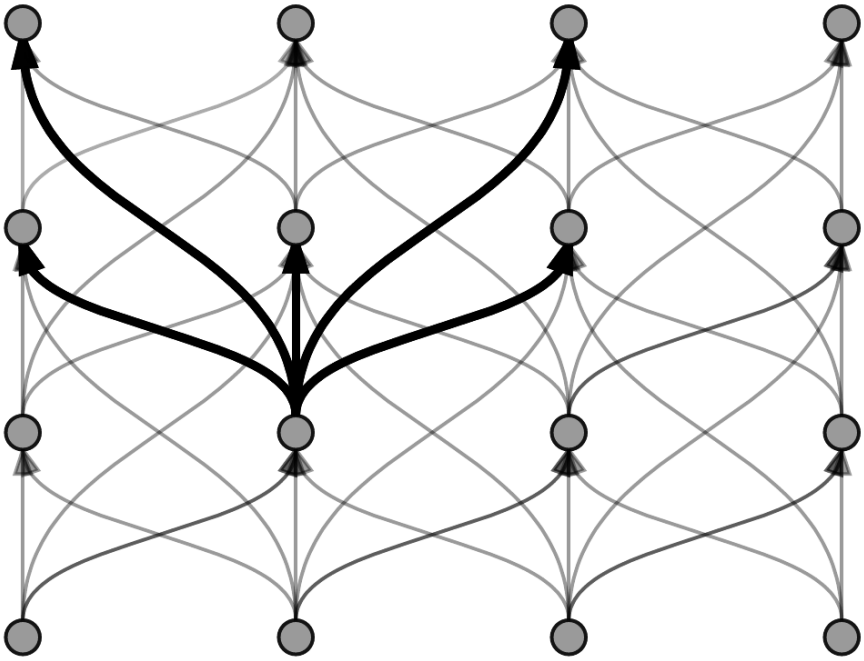


Figure 2.4: Motion planning based on state lattices. Due to the regular sampling, the motion primitives can be obtained offline and replicated to connect the sampled states. Then, an informed search algorithm can be used to find optimal paths in the lattice.

reasons, state lattices are a very efficient representation of the robot state and action spaces.

The efficiency and performance of state lattice based motion planners highly rely on the set of motion primitives used for search. Prior works [42, 161] have analyzed the relevance of having a good quality representation of the robot action space, while some authors also noted its relevance in reactive obstacle avoidance [20, 62]. In this regard, the problem of building a set of motion primitives properly representing the vehicle motion constraints while leading to an efficient search was addressed in [160], which introduced rules for designing a good state space discretization scheme and for generating a representative set of canonical actions. In that work a method for removing redundant actions for an improved search efficiency was also introduced.

The obtention of optimal paths in state lattice based representations has been addressed using graph search algorithms. The planned paths are optimal for the discretization scheme used to represent the state space —e.g. resolution of the sampled states, arrangement, etc. In this regard, different algorithms have been described in the literature, but perhaps the most commonly used for robotic applications are: A* [66, 67], D* [187] and its variants [101, 188].

More recent research in the field of graph search algorithms proposed alternative solutions to deal with complex motion planning problems. In this regard, in [125] they presented Anytime Dynamic A* (AD*), a search algorithm featuring replanning and anytime search —the capacity of obtaining sub-optimal solutions which are incrementally refined reusing the information that has already been computed.

As shown in this section, many approaches have focused on the obtention of optimal plans. Many of these proposals are modified versions of prior algorithms that have been improved to explore and represent the search space in such a way that optimality guarantees are provided. Similarly as in Section 2.2, the analyzed proposals can be classified in randomized and deterministic techniques. Among the latter, state lattices stand out for their efficient representation of the robot state space. This is due to obtaining offline a set of canonical actions which can be replicated to generate the connectivity between all the sampled states that are equally arranged. By doing so, the vehicle kinematic restrictions can be managed almost effortlessly. State lattices also stand out for their good properties in terms of discrepancy and dispersion of the sampled states, as well as their resolution completeness guarantees. Moreover, when used in combination with a graph search algorithm, optimal solutions between any two lattice states can be obtained. In this regard, different alternative algorithms exist, although AD* has desirable features to deal with complex motion planning problems.

2.4 Motion planning under uncertainty

For many robotic applications, safety and accuracy are of critical importance, and in these cases the most suitable path is not always the one with the lowest traversal time or the minimal length. Instead, it might be more interesting to prioritize other criteria, like the safety of the plan or the accuracy in its execution. Measuring these features, so that the best path can be selected accordingly, requires managing the motion uncertainty at planning time. However, classical motion planning approaches assume systems with deterministic dynamics and full knowledge about the robot state, leaving the issues regarding the uncertainty to the managed by a feedback controller [210]. To overcome this, the approaches that explicitly take into account the uncertainty have received an increasing attention in recent years, and extensive research that addressed this problem from different points of view has been conducted. Since motion uncertainty arises from different sources, like the inaccuracies in the executed controls and noisy or partial measurements, it evolves differently for each path. This makes planning under uncertainty a challenging problem due to its high computational complexity.

One family of methods addresses motion motion planning under uncertainty using the theoretical framework provided by the Markov Decision Processes (MDPs) [15]. MDPs allow modeling problems for which the stochasticity in the controls is addressed, but the robot state is assumed to be fully observable. In this regard, in [6] they presented a motion planning algorithm that combined a PRM with the MDP theory to maximize the probability of collision avoidance and successfully reaching the goal. The authors reported results for a variety of practical applications for which their approach, the Stochastic Motion Roadmap (SMRM), was suitable. This included the steering of medical needles, and motion planning for a nonholonomic robot. Their method, however, is limited to applications for which the robot state can be precisely estimated from the sensor measurements, since MDPs only cope with the uncertainty in the controls.

For problems for which the robot state is not fully observable due to the stochasticity in the measurements, the use of Partially Observable MDPs (POMDPs) [8] becomes necessary to also deal with the sensing noise. However, solving POMDPs in an exact manner is known to be of extreme computational complexity [154], and they can only be directly applied to problems with low-dimensional state spaces and limited size [89, 174]. Instead, in recent years approximate-based solutions, which rely on discretizing the state and action spaces, have been developed [11, 107, 151]. However, while these approaches partially addressed the scalability issues of solving POMDPs, further research is still needed to make them applicable

to high dimensional problems.

The later proposals of [23, 29, 68] were able to handle continuous state and action spaces in reasonable time, although they maintained a discrete representation of the reward function in belief space. This issue was overcome in [199], which presented a continuous approach that approximated the reward function in parametric form only in the areas of the environment which were relevant for planning —due to the proximity to the obstacles—, achieving polynomial runtimes in the dimensionality of the state. However, despite the good results obtained by their approach, further research is still needed to improve the computational complexity of the algorithm and make it suitable for high-dimensional systems with complex dynamics. In this regard, later research conducted by the authors suggested that it was possible to reduce the computational complexity even more [198]. Besides that, their method requires the dynamics, observation and cost functions to be smooth due to the use of gradients to compute a solution. Thus, in its current state their approach is not suitable for domains in which this formulation is not possible, for example in systems equipped with sensors with abrupt boundaries, e.g. cameras, GPS, etc.

Extending POMDPs for systems with mixed observability, in which the state can be measured only in part, is possible by factoring the model into the components which can be fully and partially measured [151]. Thus, Mixed Observable MDPs (MOMDPs) models allow improving the POMDP planning efficiency when using approximate-based algorithms, although like in prior approaches of this family of methods, scaling it up to complex robotic systems remains an open challenge. There are approaches similar to [199] that also focused on obtaining locally optimal control policies [41, 87, 195], although unlike the former they assumed maximum likelihood observations to predict the robot uncertainty in a deterministic manner. Due to this, the resulting probability density functions (PDFs) measure the capability of inferring the robot state from the observations, instead of representing the true robot state during the execution of the paths. Moreover, a common drawback of POMPD based approaches is that they focus on computing locally optimal control policies. Obtaining globally optimal solutions would be possible, but this would scale very poorly with the complexity of the environment, increasing the computational complexity of a family of solutions that already suffers from issues in this sense even more.

Another family of methods that allows planning under uncertainty is based on combining classical planning solutions, already analyzed in Sections 2.2 and 2.3, with the prediction of the uncertainty along the different candidate paths. This allows selecting the best solution

according to different criteria, but also the uncertainty is managed differently in each proposal.

The Belief Roadmap planner (BRM) [163] addressed planning in belief space focusing on systems with linear dynamics and Gaussian noise. With that method an initial uncertainty was propagated throughout all the candidate paths in a roadmap, which had been built *a priori*, and the path which maximized the information gain about the robot state was obtained. This was achieved using the Extended Kalman Filter (EKF) theory and simulating the measurements that the robot would receive in each point of the roadmap. While this was computationally expensive, they proposed a method that factorized the covariance matrix and combined the multiple EKF update steps into a single one. That way, computing the uncertainties throughout the entire roadmap became a substantially faster operation and reasonable planning times on large-scale environments were achieved. However, BRM disregards the vehicle motion constraints and the uncertainty due to the controls, assuming that the controller is able to accurately track the planned path, which is not reasonable for most robotic applications.

In [77] a trajectory optimization method [194] was combined with the outcome of the BRM algorithm. That method produced a control policy which satisfied the information constraints of the planned path at the same time that its feasibility was ensured. However, the solutions were not guaranteed to be optimal due to using PRM as basis for planning. Moreover, these approaches propagated the uncertainty assuming maximum likelihood observations, for which the predicted PDFs do not represent the true robot state.

This issue was overcome by the algorithm proposed in [197], named Linear Quadratic Gaussian Motion Planning (LQG-MP). They relied on RRT to find the path which minimized the predicted probability of collision, estimated from the PDFs taking into account the uncertainty due to the controls and due to the measurements. Their method achieved that at the same time that it avoided to simulate maximum-likelihood observations during the propagation of the uncertainties along the candidate paths that were generated by the search algorithm. Moreover, under the assumption that the solutions would be executed with a Linear Quadratic Gaussian (LQG) controller [17], their method used this prior knowledge to accurately predict the distributions of the robot true state in a realistic manner. In this sense, good results were reported for a variety of robotic platforms.

The main drawback of their strategy is that the uncertainty was only predicted for a set of candidate paths which were obtained from running RRT several times. Then, among those with a bounded probability of collision the one which minimized the cost criterion was chosen. However, it had been demonstrated that RRT was not asymptotically optimal [90], so that

strategy could not guarantee that good paths in terms of cost or predicted uncertainty would be found. In addition to this, since the paths obtained by RRT may be non smooth, the authors proposed post-processing them, an idea which had been already applied to classical planning approaches. However, when the uncertainty is managed at planning time, post-processing the solutions might affect their predicted distributions, that might be different from those that were obtained for the original plans. Therefore, if these distributions are not updated, the probabilities of collision that are estimated for the post-processed paths might be unreliable.

The authors also outlined some results for the obtention of paths using a search algorithm over pre-built roadmaps, for which the uncertainty was propagated entirely with their method. That approach outperformed the similar proposal of [163], since LQG-MP could cope with both the uncertainty due to the controls and the observations, while with BRM only the latter could be managed. However, the reported results were for simple search problems for which the roadmaps were made by hand.

The approach of [26] addressed the limitations of prior research and made it possible to obtain globally optimal solutions entirely planning in belief space. They combined a variant of LQG-MP with the RRT* algorithm, thus retaining the assumption that the obtained plans would be executed with a LQG controller. Their method relied on building a search tree which incorporated the predicted distributions throughout all the possible paths contained in it. This allowed them to evaluate all possible alternatives in terms of predicted uncertainty and total cost and, due to the asymptotically optimal guarantees of RRT*, obtaining the best solution for a given planning query.

Similarly to some other prior approaches in the literature, that method focused on minimizing the amount of uncertainty during the execution of the plans, although their cost was minimized as a secondary objective. Despite the good results reported, the authors identified an issue whereby in some circumstances might get stuck due to the uncertainty monotonically decreasing at the same time that the cost of the path increases —e.g. a robot cycling in some rich information area of the environment. This led the authors to propose a pruning method to avoid obtaining unnaturally long paths which was based on a parameter which had to be adjusted in a heuristic manner. As a final remark, the authors noted that further research was still needed to study the computational complexity of their method. After all, in their experiments a significant number of iterations of RRT* was required to find near-optimal solutions. This was due to the fact that, while the paths obtained by RRT* are significantly more smooth than those of RRT, this issue is not completely overcome when the number of samples is low.

The later approach of [130] proposed a motion planner, also based on RRT*, which considered the uncertainty due to the location of the obstacles in the map, as well as the other sources of uncertainty already considered by prior methods. Their motion planning strategy relied on a risk based cost function that managed the trade off between minimizing the duration of the path and a risk avoidance behavior. However, the authors reported results whereby the of the path, given a user specified risk threshold, did not monotonically decrease with the number of nodes in the search tree. Moreover these results were only for a holonomic robot, noting that further research was needed to extend it to systems with more complex dynamics.

Despite the drawbacks mentioned above, the proposal of [26] achieved good results in predicting the PDFs along the paths generated by RRT*. For this reason, in this thesis their method is used to manage the uncertainty due to the control and sensing inaccuracies, while novel methods that address some of the issues of prior research are proposed. In this regard, the lack of smoothness of the paths obtained by randomized sampling techniques is addressed combining the uncertainty prediction approach of [26] with a state lattice representation. A different strategy for evaluating and ordering the candidate paths by their cost is also introduced. Thus, instead of minimizing the amount of uncertainty during the execution of the paths, the proposed method focuses on minimizing, firstly, the predicted probability of collision, a measure which is directly related to the safety of the planned paths. Secondly, the traversal time of the path is minimized and, finally, the covariance of the robot state. This strategy allows addressing the issue regarding the loops that might appear as a result of navigating in the valley areas of the map with good sensing information, due to not focusing on optimizing the covariance of the true robot state.

An important drawback of prior approaches which focused on minimizing the probability of collision of the paths is that their estimation considered simplified versions of the robot shape. While this allows limiting the computational complexity of dealing with motion planning under uncertainty, it comes at the expense of some drawbacks. Firstly, due to disregarding the robot real dimensions, the reliability of the motion planner cannot be guaranteed under all circumstances if the robot is approximated by a circle [120, 197] or by a single point [26, 130]. Secondly, being too conservative and representing the robot by its outer circle might also affect the planning efficiency in some cases. In theory that would be a reasonable assumption, since the planned paths would keep an increased security distance to the obstacles. However, in practice it results in an unnecessary overestimation of the cost function which might accentuate the “narrow passage effect” [72, 76] that the sampling based algorithms ex-

perience in cluttered environments. Lastly, relying on these approximations does not allow considering the uncertainty in the robot heading. This is especially relevant for robots with asymmetries or very long shapes, for which small changes in heading might cause relevant differences in the estimated probability of collision.

This section has analyzed the relevant prior works that addressed the problem of planning under uncertainty, which is known for its high complexity. In an attempt to reduce the computational requirements, many existing approaches limited the kinds of uncertainty that they considered, or assumed certain approximations. Others have reported good results in predicting the probability distributions of the robot state, but faced different difficulties in their attempts to select a good path in terms of predicted uncertainty. However, several challenges remain.

While many authors focused on obtaining solutions that minimize the amount of uncertainty, the suitability of these paths for navigation purposes is limited due to the fact that their optimality in terms of cost is not guaranteed. In this sense, this thesis proposes minimizing the probability of collision in the first place, which is directly related to the safety of the paths, and then focusing on other criteria of the cost of the paths, like the traversal time and the uncertainty at the goal. Regarding the way in which this probability of collision is estimated, there are two aspects to consider. On the one hand, as part of the approximations mentioned before, the real dimensions of the robot are systematically disregarded in the state of the art. To address this, in this thesis a novel method to estimate the probability of collision taking into account this information is proposed. This method is based on sampling the predicted PDFs, following a regular pattern to maintain the deterministic nature of the proposed motion planning approach, and checking collisions with the obstacles in the map. On the other hand, prior research have managed the uncertainty limiting its extent to the robot position, not including the heading. However, this might affect the reliability of the planner for robots with irregular shapes. This is addressed by the method we propose in this thesis, which estimates the probability of collision by sampling the PDFs in such a way that the uncertainty in heading is taken into account. With this strategy, reliable collision free paths for all kinds of shapes are obtained.

2.5 Efficiency improvements

The scalability of state lattices is usually pointed out as a drawback in high dimensional planning problems due to the number of states that are needed to obtain a suitable representation of the state and action spaces. However, they have been successfully used for motion planning with different robotic platforms [37, 123]. Moreover, the regular arrangement of the sampled states and the possibility of computing the canonical actions offline opens the door to different efficiency improvements which allow for the planning times to be significantly lower.

In this regard, i) heuristics are key for reducing the number of states that are explored during the search of optimal paths, and ii) multi-resolution planning techniques allow managing the trade-off between the efficiency and the performance of the motion planning algorithms. The idea of improving the efficiency of the search while retaining optimality guarantees was introduced with the A* algorithm [66, 67]. Later proposals in the field of informed search algorithms introduced variations in the way that heuristics were managed, although the main concept remained unchanged. During the search, the heuristic function estimates the cost of the path between each state and the goal in order to reduce the number of states that need to be explored in order to compute the optimal solution. In this sense, although the connection between the accuracy of the heuristics and the efficiency of the search has long been known, the way in which the heuristics affect the computational complexity of search algorithms has been studied in detail [103]. Two desirable properties for a heuristic function [175] are admissibility —never overestimating the true cost to the goal— and consistency —the estimated cost between the current state and the goal is always equal or greater than the cost between the current node and a successor plus the estimated cost between the successor and the goal. While a consistent heuristic function is also admissible, the reverse might not be true. Admissibility is required to retain the optimality guarantees of algorithms of the family of A*, while consistency is desirable as it ensures that any explored state is processed only once.

At this moment there is no known closed form heuristic for state lattice based motion planners and, although different alternatives are available, finding suitable heuristics is challenging due to the fact that the motion planner takes into account the full state space and the vehicle motion constraints. For this reason, too simple estimation functions, like the Euclidean distance, might be adequate for some basic search problems, but in general their poor informative value due to disregarding the vehicle motion model results in an underestimation of the true cost of the paths, which in the end affects the efficiency of the search algorithms. In this sense, [100] described an admissible heuristic, named Free Space Heuristic (FSH), which

takes into account the robot kinematic constraints. This is achieved by using the motion planning algorithm itself to compute a Heuristic Look-Up Table (HLUT) of possible motions in free space. Such table is a kind of pattern database [36] which is computed offline and stored to be queried during the search. This heuristic provides precise estimations, especially in uncluttered environments and, since it considers free space, it is guaranteed that its estimations are optimistic with respect to the true cost of the paths. However, to make the problem of computing and storing the HLUT tractable the authors proposed using a second heuristic function, e.g. the euclidean distance, to select which queries are worthy to be included in the HLUT. Those for which both heuristics provide comparable estimations are trimmed due to their poor informative value and, in those cases, the backup heuristic is used during the search. Their research showed that this method produces look-up tables of reasonable sizes and that can be efficiently computed.

Although the heuristic described before works well for estimating the cost to the goal in sparse environments, its accuracy is affected in complex environments due to disregarding the obstacles. However, following the same strategy to design a heuristic which copes with the complexity of the map is impractical due to the infinite number of possible obstacle configurations. Instead, in [124] a low dimensional heuristic which takes into account this information, named H2D, was presented. This heuristic is based on solving online a simplified search problem disregarding the vehicle motion restrictions for the actual environment. More concretely, their approach is based on using the Dijkstra's algorithm over a grid with a fixed resolution which matches the maximum fidelity of the state lattice used to represent the main motion planning problem. Once the grid is completely explored, it stores the cost of all the shortest paths between the current robot location and all other positions belonging to the grid. Their research also detailed the strategies that make it possible to guarantee that the estimated costs are admissible and consistent, such that they can be used in A* based algorithms and retain their optimality guarantees. Notwithstanding, computing this heuristic can be costly and affect the planning efficiency, specially in large and uncluttered environments.

The behavior of these two heuristics is shown in Figure 2.5, which represents the different paths to the goal that they estimate. Both provide valuable information with complementary benefits. As mentioned before, FSH considers free space, and therefore its informative value decreases in the presence of obstacles. Conversely, this heuristic is more precise than H2D in uncluttered environments and also near to the goal, which would correspond to the area that goes after the obstacle in the example of Figure 2.5. Instead of selecting one of these heuris-

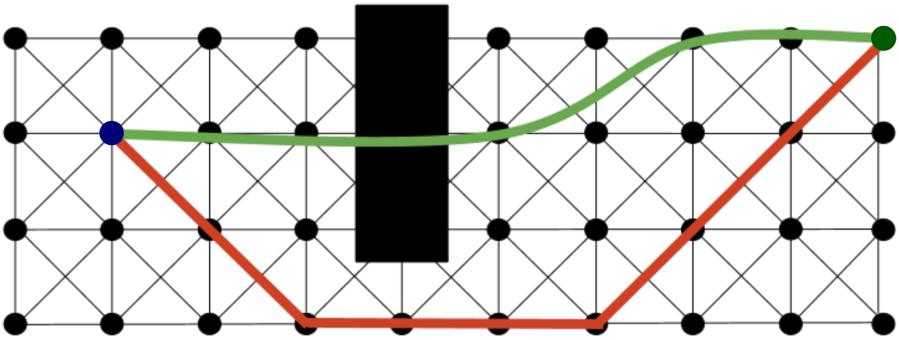


Figure 2.5: Comparison of the path estimated by the heuristic FSH, in green, which copes with the kinematic restrictions considering free space and the one estimated by the heuristic H2D, in red, which copes with the obstacles in the map but disregarding the vehicle motion model. The current state is in blue, while the green point is the goal.

tics, it is possible to improve the accuracy of the estimations by combining them, e.g. using the maximum of the individual estimates. In fact, in [124] these heuristics were used together in a state lattice based motion planner, showing that the combination of H2D and FSH can be an order of magnitude more effective than when they are used separately. Moreover, since both heuristics are admissible and consistent, these properties are retained when combining them [155].

While FSH can be computed offline, H2D has to be initialized for every motion planning query. Despite its good results, computing this heuristic can be costly and affect the planning efficiency in large environments. To address this drawback, in this thesis a novel heuristic based on H2D is presented. Instead of using a fixed resolution grid, dependent on the sampling configuration of the state lattice, it uses the information of the map to compute a multi-resolution grid which improves the efficiency of the low dimensional planning problem representation. This allows to substantially improve the computing time of this heuristic and its scalability in large environments, which makes the motion planner more efficient.

The efficiency and performance of state lattice based motion planners are also affected by the fidelity used to represent the state space —the resolution of the sampled states. While increasing the fidelity improves the performance of the motion planner and the quality of the obtained solutions, decreasing it reduces the computational complexity of the problem and

therefore the runtime of the motion planning algorithm. Using different fidelities in arbitrary regions of the environment would be particularly interesting in order to manage the trade off between planning efficiency and performance. In this regard, state lattices with graduated fidelity were introduced in [159] as part of a hierarchical planning concept in which search spaces of different fidelities were combined and unified under the same graph representation. More specifically, their approach relied on using high fidelity only in the immediate vicinity of the robot, while elsewhere a low dimensional representation which disregarded the robot dynamics model was used instead. The main drawback on that method is that it constantly updates the solution as the robot moves along the planned path. This is because the high fidelity areas change with the robot position, causing modifications in the state space representation which have to be managed online. Moreover, with that approach the robot kinematic constraints are considered only locally around the robot, so the feasibility and the optimality guarantees of the global plan cannot be retained.

These drawbacks were addressed in [123], where the authors presented a similar approach which, instead of using a lower dimensional representation outside the high fidelity areas, relied on using a subset of the motion primitives to reduce the connectivity between the sampled states. With this method fully feasible solutions were obtained, although as the strategy of using high fidelity in the vicinity of the robot and low fidelity elsewhere was maintained, replanning is still needed to manage the modifications in the connectivity of the lattice that are produced by the robot position changes.

Although some approaches mentioned above reported good results for classical motion planning problems, when dealing with the uncertainty at planning time there are some issues that have to be considered. In this regard, heuristics still can be used for reducing the computational effort of addressing motion planning under uncertainty. However, their informative value will depend on the cost function to optimize. For example, since no good heuristic is known for estimating the evolution of the uncertainty from the current robot state, the motion planning approaches that attempt to minimize this criterion will not be able to benefit from heuristics as much as other approaches that try to minimize, among other criteria, the cost of the path.

Regarding the graduated fidelity approaches described above, using any strategy which relies on changing the state space representation in execution time might be difficult to combine with uncertainty management. This is because every time the lattice or its connectivity are modified, the uncertainty of all the affected paths must be re-computed and, if necessary,

their probabilities of collision updated. These are non trivial operations and further research would be needed to solve them in real time. Thus, the existing graduated fidelity lattice approaches [123, 159], which rely on obtaining an initial solution that is updated as the robot moves along it, cannot be directly applied for planning under uncertainty. Moreover, those strategies assume that the cost of the path monotonically decreases when the fidelity of the representation is augmented. There is no guarantee, however, that the new candidate paths that appear as a result of increasing the fidelity will lead to better solutions neither in terms of motion uncertainty nor for the probability of collision, since they depend on the executed controls and the expected measurements.

To address the drawbacks of prior works in this topic, in this thesis a novel graduated fidelity strategy is presented. To avoid the need of replanning in real time and make the approach suitable for managing motion uncertainty, the method relies on varying the connectivity of the state lattice in accordance with the obstacles in the map and the maneuverability of the robot. More concretely, subsets of motion primitives —the actions which connect the lattice states— are obtained, so that each group contains the maneuvers of the same kind but with different lengths. Then, the fidelity is selected at the action level in accordance with the predicted uncertainty and the obstacles in the map, in such a way that the fidelity decreases only if the probability of collision of the candidate paths is not affected. By doing so, the trade-off between the computational complexity and the quality of the solution is managed, causing a significant reduction in the planning runtimes. Moreover, our method does not require updating the solution unless the map changes, since the fidelity does not depend on the position of the robot. Thus, some of the prior work drawbacks mentioned above are solved.

2.6 Autonomous navigation for UAVs

Up to this point, the different techniques that can be used for motion planning have been analyzed. There is a great variety of methods that have proven successful in problems of different domains and, in this regard, most of the prior research focused on mobile robots of different kinds. The ultimate goal of those planning algorithms is integrating them in real systems for the purpose of navigating autonomously. However, this problem does not only involve the obtention of a suitable plan, but also other relevant aspects like the robot localization [52] and control [3, 17, 22].

The popularity increase that the Unmanned Aerial Vehicles (UAVs) have experienced in

recent years has motivated a growing interest from the research community in increasing their autonomous navigation capabilities. Many different kinds of vehicles may fall under the denomination of UAV, such as which fixed-wing platforms, helicopters or multi-rotors. However, the latter stand out for their versatility and their notable maneuvering capabilities, which make them suitable for addressing a variety of real world problems. Another reason for their popularity is their affordable cost. Thus, it is not surprising that many proposals in this field have reported results on multi-rotors.

Besides motion planning, many prior works have focused on addressing specific issues of UAV navigation, especially in GPS-denied environments. Among others, some noteworthy examples are visual SLAM techniques [140, 145], robust control [176] or landing in hazardous conditions [150]. Although these related problems are of great relevance as well, given the connection with the topics of this thesis, the analysis of the state of the art proposals will focus on those related to the problem of planning routes for the autonomous navigation of UAVs.

Due the computational complexity of the problem, many methods that make important simplifications and that allow coping with the limited computing resources of most on-board computers have been proposed. In this regard, some authors addressed autonomous navigation using strategies with very low computational complexity, but with certain limitations in their functionalities. Thus, in [211] the authors presented an approach focused on navigation in indoor corridors only with the information of an on-board camera. Their method obtains the most suitable action with a Proportional-Derivative (PD) controller and, although the UAV is able to navigate autonomously, the absence of a high level planner constraints this strategy to environments with the specific features for which it was developed. The approach of [60] relied on the FM^2 method to generate navigation plans in a non uniform terrain at a fixed altitude from the ground. Despite not considering the UAV kinematic constraints, their method can be adjusted in terms of the smoothness of the paths and the restrictions of the flight altitude, so the obtained solutions might be feasible depending on the parameter tuning.

Another kind of approaches use hybrid architectures consisting in a high level planning algorithm which computes collision-free paths for obstacle avoidance, and then using a low-level local planner which obtains the appropriate control commands for tracking the desired path. A variety of solutions of this kind have been described in the literature.

In [148] they used a grid A* based planner which relied on a multi-resolution representation for the sake of efficiency, and which was constantly updating the solution based on the

map updates. That obtained path was used by a lower level planner based on Potential Fields which dealt with the obstacle avoidance locally around the UAV, and computed the appropriate controls to follow the provided path. A drawback of this method is that the UAV might get stuck in local minimums if the plans obtained by A* end up being unfeasible due to the obstacles. This might occur due to the fact that, while the low level planner takes into account the UAV shape to cope with the obstacles in a safe manner, the high level path is computed considering a punctual robot. In those cases replanning is necessary to reach the goal using an alternative path, if it exists.

Similarly, in [80] they used the A* algorithm as high level planning solution. In that work, the authors proposed using Artificial Potential Fields to deal with the local minimums instead of relying on updating the high level plan to overcome blocking situations. However, the method needs to compute the control commands in real time and, therefore, its suitability in large environments might be compromised due to the scalability issues.

The two-level planner approach presented in [149] relied on Informed RRT* [53] to obtain optimistic high level paths comprised by straight lines. This was combined with a low level planner which represented the UAV dynamic constraints as polynomials to generate a smooth and feasible plan which was constantly updated. The main drawback of their method is that it requires a continuous defined cost function to deal with collision avoidance. In addition to this, they relied on a map representation in which the unknown space has to be treated either as occupied or free. Both options have different drawbacks that affect the obtention of the low level trajectories. As pointed out by the authors, this is especially relevant in real experiments, as the sensors rarely provide dense measurements, and this results in map representations that might have areas without information. Treating them as free areas might result in unwanted collisions if the sensor cannot observe the environment entirely, e.g. colliding with the ceiling due to not measuring the space above the UAV during the flight. Conversely, if those spaces are treated as occupied the difficulty of finding collision free trajectories increases substantially.

Another hybrid strategy was presented in [156], based on a variant of the Lazy Theta* algorithm [143, 144] which computed a high level path used for guiding the UAV to the next waypoint. Their high level planner obtains paths comprised by any-angle straight lines that are not constrained to the grid edges, which allows to minimize their smoothness issues. Moreover, it deals with the restrictions imposed by the placement of the on-board camera by preventing the UAV to be in poses for which the optical sensor has occlusions. However,

since the planner disregards the vehicle kinematic restrictions, the velocity commands are calculated by a low level strategy that deals with the kinematic restrictions and the accuracy of the trajectory tracking. Despite the good results obtained by this kind of approaches, a common drawback shared by all of them is that due to disregarding the vehicle dynamics at planning time, they are unable to retrieve optimal paths that might be of critical importance for some tasks.

Nevertheless, for some practical applications it might be indispensable to compute the low level trajectories separately. For example, aerial videography requires obtaining plans which are constrained in different ways —e.g. desired viewpoints, mutual visibility of cameras, people tracking, etc.— and that have to be adapted in real time to the environment conditions. In this regard, the hybrid approach of [142] achieved remarkable results by taking a user-specified high level plan and using a low level trajectory generator based on the Model Predictive Control (MPC) [44] framework.

Another family of methods addressed the generation of feasible paths for the autonomous navigation of UAVs taking into account the kinematic restrictions at planning time instead of using hierarchical approaches. In this regard, in [166] a method based on the POMDP framework which incorporated the UAV dynamic constraints at planning time was presented. The trajectory tracking error and the proximity of obstacles were taken into account as a cost criterion, and the most appropriate command was computed in real time according to the current state of the UAV, following a horizon control strategy. Although they proposed solving the POMDPs using an approximation method based on nominal belief-state optimization (NBO) [139] to address the efficiency issues of that framework, they worked with a limited motion model which assumed flying at a constant altitude.

Prior research has also reported results applied to UAVs using both randomized and deterministic sampling based methods for planning. Among the former, the approach of [208] represented the planning space in cylindrical coordinates and used RRT to plan feasible paths using the Dubins model. The authors also derived the conditions under which collision free paths for the UAV could be obtained and global convergence could be achieved. Despite this, the paths obtained by RRT usually have smoothness issues and require post-processing them to avoid undesirable maneuvers during flight. Moreover, as analyzed in Section 2.2, RRT is not asymptotically optimal.

State lattice based approaches have also been applied to the autonomous navigation of UAVs. In this regard, in [162] a state lattice based motion planner which coped with the

complex UAV motion constraints was presented. In that work they detailed the strategies for sampling the state and action spaces in accordance with the particularities of motion planning for UAVs. Experimental results showed the efficiency of their method compared to other approaches, while no further post-processing of the obtained paths was required. In [131] they also used a state lattice representation combined with the AD* algorithm [125] for an UAV with an asymmetric footprint. In addition to the general planning approach, they described a 3D heuristic based on grid search and Breadth First Search (BFS) for planning with improved efficiency. Despite the high informative value of the heuristic, it suffered from scalability issues in large environments, for which the authors introduced some optimizations that partially addressed them. They also analyzed their approach against RRT and RRT* based planners for the same problem, and showed that they achieved better efficiency and performance both in known and partially unknown environments.

Despite the good results of some hybrid planning approaches when dealing with partially unknown environments, they cannot offer guarantees regarding the optimality of the planned paths due to not taking into account the kinematic restrictions at planning time. This issue was overcome by the approaches based on sampling the state space which are able to retrieve solutions with the minimal cost observing these restrictions. However, no prior research taking into account the motion and sensing uncertainty at planning time has been applied to UAVs due to the high computational complexity of this planning problem. Yet, in some real world problems it is desirable to compute the best path in accordance with other measures than the cost of the path. In certain situations it might be interesting to compute solutions that maximize the probability of collision avoidance, e.g. for navigating in highly cluttered environments. Moreover, depending on the system setup the uncertainty that arises from the imprecise motions of the UAVs or the sensing inaccuracies might affect the reliability of the planning solution.

In this thesis we address these issues by presenting a motion planning approach which manages the motion and sensing uncertainty at planning time and selects the best path in terms of safety and traversal time. To achieve a reasonable planning efficiency, a state lattice with graduated fidelity representation —which is able to deal with the overhead caused by taking into account the uncertainty— is presented. Moreover, a novel multi-resolution heuristic which adapts to the obstacles in the map is described. Our approach makes it possible to obtain reliable solutions very efficiently, for which it is suitable for UAV autonomous navigation.

CHAPTER 3

MOTION PLANNING UNDER UNCERTAINTY IN GRADUATED FIDELITY LATTICES

In this chapter we detail our proposal for motion planning under uncertainty. In Section 3.1 we address the problem formulation, including how the motion primitives are obtained —Section 3.1.1—, an overview of the search algorithm main operations —in Section 3.1.2— and how the uncertainty is propagated throughout the candidate paths —Section 3.1.3.

Section 3.2 details the contributions of this thesis to the motion planning research field, which focus on improving the reliability and the efficiency. More concretely, this section introduces a method for estimating the probability of collision of the paths reliably —contribution C1, Section 3.2.1—, an approach for obtaining a graduated fidelity state lattice which adapts to the obstacles in the environment —contribution C2, Section 3.2.2— and a multi-resolution heuristic that estimates the cost to the goal due to the map occupancy information —contribution C3, Section 3.2.3.

Section 3.3 discusses the experimental results in different scenarios, showing the performance of the proposed methods for robots of different shapes and with a variety of motion models and uncertainty conditions. More concretely, Section 3.3.1 shows the reliability of the probability of collision estimation, and Section 3.3.2 the planning efficiency due to the graduated fidelity approach. Section 3.3.3 discusses the results regarding anytime search, and in Section 3.3.4 we report planning results for experiments in real scenarios. Section 3.3.5 analyzes the efficiency of the proposed multi-resolution heuristic. Finally, Section 3.4 ends the chapter with a discussion and some final remarks.

3.1 Planning on state lattices

3.1.1 Motion primitives

The motion planner presented in this thesis relies on a state lattice to sample the state space, $\mathscr{X}$, in a deterministic and regular manner. We have chosen a rectangular arrangement of the samples, although other configurations are possible. The states belonging to the lattice, $\mathscr{X}_{lat}$, are connected by a set of actions —$\mathscr{U}$, also called motion primitives— extracted from the dynamics model. Due to the regular arrangement of the sampled states, these actions can be computed offline and efficiently stored. As they are position-independent, the same motion primitive connects every pair of states equally arranged. Figure 2.4 illustrates the regularity of the states belonging to the lattice and the connectivity obtained via replication of the set of actions $\mathscr{U}$.

It is straightforward that using the motion primitives to connect the lattice states ensures that, by construction, this structure is generated in accordance with the robot dynamics. Since the kinematic restrictions are observed, all paths contained in it are feasible.

In our model, the control set was obtained applying a numerical optimization technique based on the Newton-Raphson method, introduced by [70]. The resulting actions are optimal in terms of cost, given the constraints: the initial an final states —which belong to $\mathscr{X}_{lat}$— and the dynamics model.

The motion primitives have been parametrized via cubic spline interpolation [38] to represent the evolution of the controls over time. The parameters are the knot points of the splines, a set of equally spaced values —$k_1, k_2, ..., k_n$— from which the rest of the function is interpolated, as shown in Figure 3.1. Each control variable is represented by a different spline, which might be given by a different number of knots. Hence, the parameter vector for a motion primitive is defined as follows:

$$p = [(k_0^1, k_1^1, ..., k_{n_1}^1), (k_0^2, k_1^2, ..., k_{n_2}^2), ..., t] \tag{3.1}$$

where k_i^j is the i-th knot belonging to the spline of the j-th control variable, and t is the duration of the motion primitive.

Finally, the parameter vector p is optimized with respect to an error function, $e(p)$, which measures the difference between the desired final state and the one given by the parameters

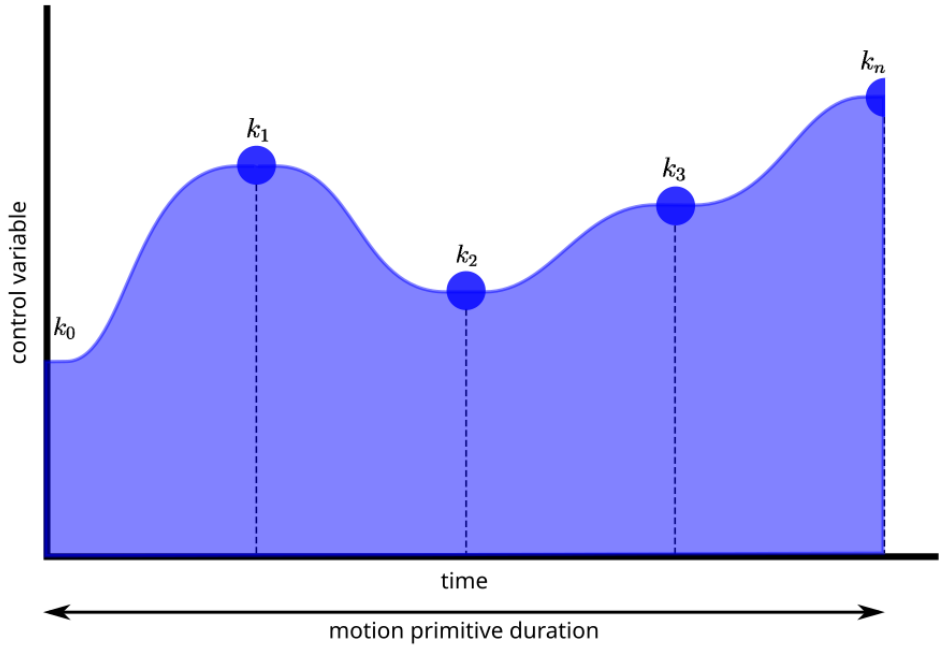


Figure 3.1: Parametrization of the control function via cubic spline interpolation. k_0 and k_n are given by the initial and final states connected by the motion primitive, while k_i are the knot points defining the rest of the function.

—obtained from the dynamics model. The parameter vector is modified following:

$$\Delta p = -\left[\frac{\partial e(p)}{\partial p}\right]^{-1} \cdot e(p) \tag{3.2}$$

This is repeated until the constraints are satisfied, typically when $e(p)$ is under a threshold.

The dynamics model is learned from motion data of the robot, as described in [1]. The approach relies on parametrizing the equations for all linear and angular velocities in this manner:

$$v_{t+1} = \beta_1^v \cdot v_t + \beta_2^v \cdot u_t^v + \beta_0^v \tag{3.3}$$

$$\omega_{t+1} = \beta_1^\omega \cdot \omega_t + \beta_2^\omega \cdot u_t^\omega + \beta_0^\omega \tag{3.4}$$

making use of an iterative least-squares method to obtain the parameters, β_i^v and β_i^ω, which best fit the input data. u_t^v and u_t^ω are the linear and angular controls, respectively. The motion primitives obtained with this model are accurate representatives of the robot maneuvering capabilities, since they encode its real response to the different controls.

3.1.2 Optimal path

As the state lattice has the structure of a graph, an informed search algorithm can be used to find the optimal path in it. In this thesis we use Anytime Dynamic A* [125] because of its capability to obtain sub-optimal bounded solutions varying a heuristic inflation parameter, ε. The solutions can be iteratively refined taking advantage of the information previously calculated, without need for replanning from scratch.

Algorithm 1 outlines the main operations of AD* —see [125] for the detailed pseudocode of AD*. The inputs are the initial state and the goal, x^0 and x^G, and the output is the path with minimal cost connecting them. In each iteration a state x^a is extracted from *OPEN* —Alg. 1:9. This state is the one which minimizes the sum of the cost from the start, c_x, and the estimated cost to the goal —h_x, given by the heuristic—, scaled by ε. Next, the successors of x^a are retrieved —X^b, in Alg. 1:10— and the evaluation of the outgoing actions is done in Algorithm 1:12. Finally, in Algorithm 1:13-16, the cost of x^b is updated, its heuristic obtained and it is inserted into the *OPEN* queue to be explored in following iterations —only if x^b is visited for the first time or the current path improves an existing one. The algorithm finds a valid path when the state extracted from *OPEN*, x^a, is the goal x^G.

The planning algorithm relies on the use of heuristics to efficiently explore the state space and obtain an optimal solution in fewer iterations. The heuristic function provides an estimation of the cost between each state x^a and the goal x^G, which influences the order in which the states are processed and therefore the number of iterations needed to find the optimal path.

AD* introduces a parameter, ε, which inflates the values of the heuristic. This allows obtaining sub-optimal bounded solutions faster than the optimal one. Thus, the algorithm is run in an iterative way, obtaining an initial solution for $\varepsilon = \varepsilon_0$. This solution is refined in subsequent executions, after decreasing the value of ε, taking advantage of the information previously calculated —Alg. 1:19-20. This is less computationally expensive than obtaining a new solution from scratch every time ε changes.

The heuristic function is a combination of two values —proposed by [123] and [161]— which allows using both the information of the obstacles in the map and the dynamics model:

one is the cost of the path considering only the kinematic restrictions, FSH, while the other is the cost of the path only taking into account the information of the map, H2D.

As it takes into account the kinematic restrictions, obtaining FSH is a costly operation. Therefore, it is computed offline and stored in a Heuristic Look-Up-Table, as described in [100]. The process starts with a first step in which Dijkstra's algorithm is applied to populate the table in a rapid way, followed by another step in which the most complex maneuvers are included. This heuristic takes advantage of the regularity of the lattice and the symmetries in the control set to improve the efficiency of its calculation and storage.

On the contrary, H2D has to be initialized every time the planner is run —Alg. 1:2—, since it depends on the location of the goal and the obstacles in the environment. Therefore, the lower the obtention time of this heuristic, the higher the overall efficiency of the planner.

Algorithm 1 Main operations of the search algorithm

Require: x^0, initial state
Require: x^G, goal state
Require: ε_0, initial value of ε
1: **function** MAIN (x^0, x^G, ε_0)
2: INITIALIZEHEURISTIC(x^0, x^G) ▷ Alg. 5
3: $\varepsilon = \varepsilon_0$
4: **while** $\varepsilon >= 1$ **do**
5: $c_{x^0} = 0$
6: $h_{x^0} =$ HEURISTIC(x^0)
7: $OPEN = \{x^0\}$
8: **repeat**
9: $x^a = \arg\min_{x \in OPEN}(c_x + \varepsilon \cdot h_x)$
10: $X^b =$ SUCCESSORS(x^a) ▷ Alg. 4
11: **for all** $x^b \in X^b$ **do**
12: $\hat{c}_{x^b} = c_{x^a} +$ COST(x^a, x^b) ▷ Alg. 3
13: **if** x_b not visited or $\hat{c}_{x^b} < c_{x^b}$ **then**
14: $c_{x^b} = \hat{c}_{x^b}$
15: $h_{x^b} =$ HEURISTIC(x^b) ▷ Alg. 5
16: $OPEN = OPEN \cup \{x^b\}$
17: $OPEN = OPEN - \{x^a\}$
18: **until** $x^a = x^G$
19: publish $path(x^0, x^a)$
20: decrease ε
21: **return**

3.1.3 Uncertainty management

Managing the uncertainty requires to predict the probability of the robot being in each state of the path. This uncertainty depends on the one at the initial state, the executed controls and the location accuracy, and therefore it varies along the different candidate paths in the lattice. The prediction of these probability distributions is integrated in planning time.

This proposal focuses on nonlinear, partially observable systems. Dynamics —f— and observations —z— are described in a discrete time manner:

$$x_{t+1} = f(x_t, u_t) + m_t, \qquad m_t \sim \mathcal{N}(0, M_t) \tag{3.5}$$

$$z_t = z(x_t) + n_t, \qquad n_t \sim \mathcal{N}(0, N_t), \tag{3.6}$$

where $x_t \in \mathcal{X}$ are the states of the robot, $u_t \in \mathcal{U}$ are the controls and z_t are the measurements. m_t and n_t are random motion and observation disturbances, which are described by Gaussian distributions. M_t and N_t are their respective covariances.

Obtaining the cost for a path between two states x^a and x^b is a two step process. First the uncertainty is propagated along the trajectory, and then the resulting probability distributions —PDFs— are used to estimate the probability that the robot collides when executing the path.

The former is done following the approach of [26], which has good results for the kind of systems described above. It is an EKF-based method and it manages the uncertainty which arises from the controls and the observations. Moreover, it also takes into account the influence of using a Linear Quadratic Gaussian controller —LQG, a widely extended controller to correct deviations from the planned path in execution time, detailed in [17]. For approximating the probability of collision along the paths we introduce a novel method in Section 3.2.1 which takes into account the real shape of the robot and provides a reliable estimation.

The uncertainty prediction is detailed in Algorithm 2. The inputs are the beginning and final states of the trajectory —x^a and x^b—, while the output is the list of PDFs along the path between them —$P^{a:b}$, which is obtained iteratively propagating the uncertainty at $x^a \sim \mathcal{N}(\bar{x}^a, \Sigma^a)$. $u^{a:b}$ are the control commands of the trajectory —Alg. 2:2. L_t is the gain of a LQG controller, which is taken into account due to its influence on the PDFs. H_t is the Jacobian of the measurement model and A_t and B_t are the Jacobians of the dynamics model.

The motion uncertainty is predicted as follows: first, an EKF is used to calculate the distributions of the state in the prediction step —$\mathcal{N}(\bar{x}_t, \bar{\Sigma}_t)$, in Alg. 2:7-8— and the true one after the update —$\tilde{x}_t \sim \mathcal{N}(\bar{x}_t, \widetilde{\Sigma}_t)$, Alg. 2:9-10. This distribution can be seen as $P(x_t|\tilde{x}_t)$, which

represents the probability of being in x_t if the EKF predicts so. After, $P(\tilde{x}_t) = \mathcal{N}(\bar{x}_t, \Lambda_t)$ is obtained in Algorithm 2:11-12. This models the uncertainty due to obtaining the state estimation without having taken the real observations. These two distributions are used to calculate $P(x_t, \tilde{x}_t) = P(x_t|\tilde{x}_t)P(\tilde{x}_t)$, the joint one of the real robot state and the true state given by the EKF. With all of the above we can finally get the PDF of the real state, $P(x_t) = \mathcal{N}(\bar{x}_t, \widetilde{\Sigma}_t + \Lambda_t)$, in Algorithm 2:13, which the motion planner uses as:

$$x_t \sim \mathcal{N}(\bar{x}_t, \Sigma_t) \tag{3.7}$$

This distribution is the one the planner uses to estimate the probability of collision along the paths.

Algorithm 2 Uncertainty propagation along a trajectory between x^a and x^b

Require: x^a and x^b, beginning and final states

1: **function** UNCERTAINTY(x^a, x^b)
2: $u^{a:b} = cmd(x^a, x^b)$
3: $\bar{x}_{t-1} = x^a$
4: $\Sigma_{t-1} = \Sigma_{x^a}$
5: $P^{a:b} = \emptyset$
6: **for all** $u_t^{a:b} \in u^{a:b}$ **do**
7: $\bar{x}_t = f(\bar{x}_{t-1}, u_t^{a:b})$
8: $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + M_t$
9: $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + N_t)^{-1}$
10: $\widetilde{\Sigma}_t = (I - K_t H_t)\bar{\Sigma}_t$
11: $C_t = A_t + B_t L_t$
12: $\Lambda_t = C_t \Lambda_{t-1} C_t^T + K_t H_t \bar{\Sigma}_t$
13: $\Sigma_t = \widetilde{\Sigma}_t + \Lambda_t$
14: $P^{a:b} = P^{a:b} \cup \{x_t \sim \mathcal{N}(\bar{x}_t, \Sigma_t)\}$
15: $\bar{x}_{t-1} = \bar{x}_t$
16: **return** $P^{a:b}$ ▷ PDFs of the path

3.2 Improving the reliability and the efficiency of the motion planner

In this section the contributions of this work are detailed. First, we present a method to estimate the probability of collision of each path from its associated uncertainty which takes

into account the real shape of the robot and the uncertainty in heading —contribution C1. Then, we propose a novel graduated fidelity approach which goes in accordance with the obstacles in the environment and the maneuverability of the robot —contribution C2. Finally we present H2DMR, a heuristic based on the idea of H2D which makes use of multi-resolution techniques to improve its scalability and efficiency —contribution C3.

3.2.1 Reliable probability of collision

The goal of the planner is to obtain paths minimizing the probability of collision and the traversal time. To achieve this, each candidate path between two states, x^a and x^b, is evaluated to obtain a cost comprised by three elements: a safety measurement —$c^{a:b}$, proportional to the probability of collision along it—, the traversal time $t^{a:b}$, and the uncertainty at the final state Σ^b.

Algorithm 3 details how this evaluation is done. As mentioned before, this requires first obtaining the PDFs in Algorithm 3:2, since they are needed to calculate $c^{a:b}$ and to know the final uncertainty Σ^b. On the contrary, the traversal time is directly obtained from the motion primitives —$t^{a:b}$, in Alg. 3:3.

Algorithm 3 Cost of a trajectory between x^a and x^b

Require: x^a and x^b, beginning and final states

1: **function** COST(x^a, x^b)
2: $P^{a:b} = \text{UNCERTAINTY}(x^a, x^b)$ ▷ Alg. 2
3: $t^{a:b} = time(u^{a:b})$
4: $c^{a:b} = 0$
5: **for all** $x_t^{a:b} \sim \mathcal{N}(\bar{x}_t^{a:b}, \Sigma_t^{a:b}) \in P^{a:b}$ **do**
6: $w_c = 0$
7: $w_t = 0$
8: $X^S = sampling(x_t^{a:b})$
9: **for all** $x^s \in X^S$ **do**
10: $w = pdf(x^s, x_t^{a:b})$
11: $w_t = w_t + w$
12: **if** $collision(x^s)$ **then** ▷ with real shape
13: $w_c = w_c + w$
14: $p_c = w_c / w_t$
15: $c^{a:b} = c^{a:b} - \log(1 - p_c)$
16: **return** $\left[c^{a:b}, t^{a:b}, \Sigma^b\right]$ ▷ Σ^b, uncertainty at x^b

The probability of collision is estimated from the PDFs by obtaining a set of samples X^S and checking collisions with the obstacles in the environment taking into account the real shape of the robot, as detailed in Algorithm 3:8-13. The sampling strategy is based on the method of the Unscented Kalman Filter —UKF [88]— to obtain the sigma points of a PDF, which represents the distribution reasonably well with a small number of samples. These sigma points are distributed in all dimensions, making them suitable for checking collisions also dealing with the uncertainty in heading. Moreover, their obtention only depends on the parameters of the PDF, retaining the deterministic nature of the motion planner.

A n-dimensional distribution belonging to the path, $x_t \sim (\bar{x}_t, \Sigma_t)$, is sampled as follows:

$$\begin{aligned} x^s_{[i^+]} &= \bar{x}_t + \lambda \cdot \left(\sqrt{\Sigma_t}\right)_i \qquad \text{for } i = 1, ..., n \\ x^s_{[i^-]} &= \bar{x}_t - \lambda \cdot \left(\sqrt{\Sigma_t}\right)_i \end{aligned} \tag{3.8}$$

The scaling factor λ allows obtaining samples with different distance to the mean —i.e. $\lambda = 3$ will generate samples with a distance of 3 standard deviations from the most probable state. Two samples —$x^s_{[i^+]}$ and $x^s_{[i^-]}$— are obtained for each dimension of the distribution, adding and subtracting each column of the factorized matrix —$\left(\sqrt{\Sigma_t}\right)_i$— to the mean of the PDF, $\bar{x}_t$. This generates $2 \cdot n + 1$ samples for each value of λ, the sigma points. While these are good representatives of the distribution, their coverage for checking collisions with obstacles in the environment may not be enough. For example, samples with different headings would be generated only in the position of $\bar{x}_t$. For this reason, our method also operates with different columns of the covariance matrix to obtain samples not only in the main axes of the distribution, but also in the diagonals, which results in an improved coverage of the PDF. Thus, for each sigma point from Equation 3.8 —$x^s_{[i^-]}$ and $x^s_{[i^+]}$— the following samples are obtained:

$$\begin{aligned} x^s_{[i^-,\, j^-]} &= x^s_{[i^-]} - \lambda \cdot \left(\sqrt{\Sigma_t}\right)_j \qquad \text{for } i, j = 1, ..., n \\ x^s_{[i^-,\, j^+]} &= x^s_{[i^-]} + \lambda \cdot \left(\sqrt{\Sigma_t}\right)_j \qquad j \neq i \\ x^s_{[i^+,\, j^-]} &= x^s_{[i^+]} - \lambda \cdot \left(\sqrt{\Sigma_t}\right)_j \\ x^s_{[i^+,\, j^+]} &= x^s_{[i^+]} + \lambda \cdot \left(\sqrt{\Sigma_t}\right)_j \end{aligned} \tag{3.9}$$

Figure 3.2 shows the samples obtained applying this method for $\lambda = 1, 2$ and 3. These samples are used to check collisions with the surrounding obstacles, for which the real shape of the robot is taken into account. Each sample x^s has a weight, w, in accordance with its

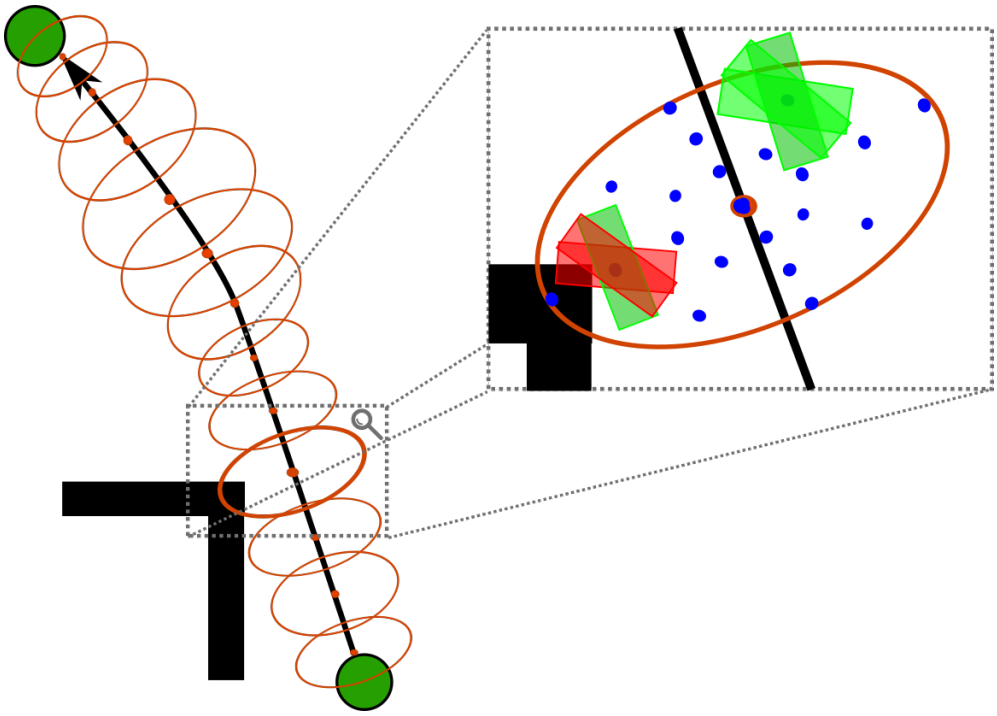


Figure 3.2: Estimation of the probability of collision taking into account the real shape of the robot, via sampling the PDF using the strategy of an UKF.

probability —function pdf, in Alg. 3:10. w_t is the sum of weights of all samples, while w_c is the sum of those in which the real shape collides —Alg. 3:11 and Alg. 3:13, respectively. The quotient between them is the probability of collision of the distribution, p_c, in Algorithm 3:14. Finally, the individual estimations are combined in a logarithmic scale to obtain a cost related to the probability of collision of the entire path, as detailed in Algorithm 3:15. By doing so we assume that the probabilities of collision in different stages of the path are independent, in the same way that most approaches in the state of the art. Although this is not the case, it is a reasonable assumption for practical purposes.

Each element returned by the cost function —Alg. 3:16— represents an objective to be minimized by the motion planner, and therefore an order of priority was introduced when

comparing the cost of two paths, which is done as follows:

$$\begin{aligned} cost(x^{a:b}) < cost(x^{a:c}) \Leftrightarrow (c^{a:b} < c^{a:c}) \vee \\ (c^{a:b} = c^{a:c} \wedge t^{a:b} < t^{a:c}) \vee \\ (c^{a:b} = c^{a:c} \wedge t^{a:b} = t^{a:c} \wedge \Sigma^b < \Sigma^c) \end{aligned} \tag{3.10}$$

Thus, the motion planner first minimizes the cost related to the probability of collision, prioritizing the safety of the planned paths. Among all the safe paths, it selects the one minimizing the traversal time and, finally, the uncertainty at the goal. Therefore, the planned paths are safe and optimal. Moreover, since the probability of collision is estimated taking into account the real dimensions of the robot, it is reliable for all kinds of shapes.

3.2.2 Graduated fidelity lattice

The efficiency of the planner strongly depends on the fidelity of the state lattice. While a higher fidelity allows more precision in representing the state space and the maneuvering of the robot, decreasing it significantly diminishes the runtime of the search. Although the latter might result in paths with higher costs, the use of a graduated fidelity lattice can balance precision and efficiency adequately. Moreover, if the fidelity adapts to the obstacles in the environment and the maneuverability of the robot, the efficiency can be improved with minimal impact in the planning results, as shown in Section 3.3.

Figure 3.3 depicts a state lattice with graduated fidelity, in which only those areas which require complex maneuvering to avoid obstacles in the environment are represented with high fidelity. The proposed approach is to select, whenever possible, the longest maneuver of each type to move between states. To achieve this, those motion primitives in $\mathscr{U}$ which are similar maneuvers of different lengths are grouped. Then, the longest primitive of each group which does not affect the probability of collision is selected, and the rest discarded. This allows to diminish the number of states belonging to the lattice, and at the same time the number of candidate paths connecting them, therefore simplifying the state space. This technique is applied when generating the successors of the state x^a explored by the search algorithm, in Algorithm 1:10.

Algorithm 4 details how the outgoing trajectories of a state x^a are selected using the proposed graduated fidelity technique. As mentioned before, this procedure first requires grouping the motion primitives from $\mathscr{U}$ in maneuvers of the same kind but different length. Those trajectories with the same values for orientations, linear and angular speeds both at the begin-

ning —θ_i, v_i, ω_i— and at the end —θ_f, v_f, ω_f— belong to the same group —$\mathscr{U}_{(\theta_i,v_i,\omega_i,\theta_f,v_f,\omega_f)}$. The union of all groups is the whole set of primitives —$\mathscr{U} = \bigcup \mathscr{U}_{(\theta_i,v_i,\omega_i,\theta_f,v_f,\omega_f)}, \forall(\theta, v, \omega) \in \mathscr{X}_{lat}$—, whereas the intersection of any two of them is empty.

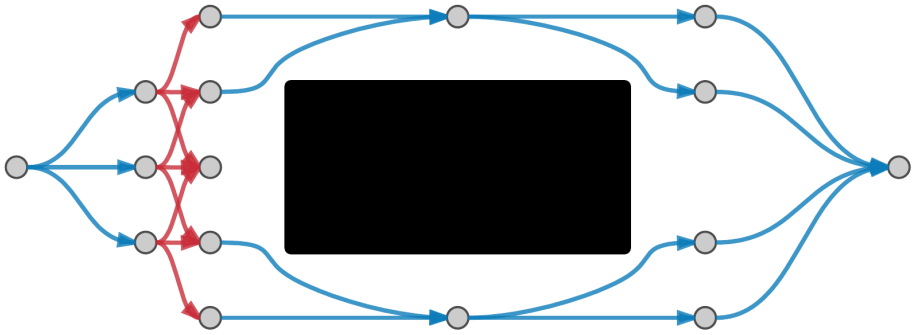


Figure 3.3: Example of graduated fidelity lattice. Trajectories in red are those with highest fidelity, required to maneuver near the obstacle —in black. The rest of trajectories have lower fidelities —in blue—, since the obstacles do not affect the maneuverability.

Algorithm 4 Successor generation for a graduated fidelity state lattice

Require: $\mathscr{U} = \{\mathscr{U}_{(\theta_i,v_i,\omega_i,\theta_f,v_f,\omega_f)}\}, \forall(\theta, v, \omega) \in \mathscr{X}_{lat}$

function SUCCESSORS(x^a)
  $\theta_i = x^a_\theta$; $v_i = x^a_v$; $\omega_i = x^a_\omega$
  $\Gamma = \emptyset$
  for $U \in \{\mathscr{U}_{(\theta_i,v_i,\omega_i,\theta_f,v_f,\omega_f)}\}, \forall(\theta_f, v_f, \omega_f)$ **do**
    repeat
      $U^c = \arg\max_{t^{a:b}}(U)$ ▷ Get longest
      $U = U \setminus U^c$
    until CHECK($U^c_{x^a}, U^c_{x^b}$) $\vee$ $U == \emptyset$
    $\Gamma = \Gamma \cup U^c$
  return Γ
function CHECK(x^a, x^b)
  $\kappa^a = cell(x^a)$; $\kappa^b = cell(x^b)$ ▷ Get map cells
  $s^a = size(\kappa^a)$; $s^b = size(\kappa^b)$ ▷ Get size of cells
  $c^{a:b}$ = COST(x^a,x^b)[0] ▷ Alg. 3
  return $(s^a + s^b \geqslant \left\|x^a - x^b\right\|) \wedge (c^{a:b} == 0)$

The successors of a state, Γ, are generated as follows: First, those groups of trajectories with the same initial speeds —v_i and ω_i— and orientations —θ_i— as x^a are selected, in Algorithm 4:4. Then, the algorithm chooses a primitive of each group to be part of the successors and discards the rest. The candidates, U^c, are evaluated in descending order by length until one that fulfills the restrictions is found, as Algorithm 4:5-8 details. If none of the trajectories in the group fulfills the restrictions, the shortest one is selected.

With regard to the evaluation of candidates, two conditions must be fulfilled to select them: the resolution of both the source and destination octree cells, s^a and s^b, and the probability of collision of the candidate trajectory U^c. The former is related to the structure of the octree: in the vicinity of the obstacles the cells contain different occupancy information and cannot be compacted in higher level cells, so the higher the cluttering the lower the size of the cells in the map. Thus, limiting the length of the maneuvers in accordance with the size of the cells —Alg. 4:12-13— results in a lattice with high fidelity only in those areas challenging for the planner. The second condition is introduced to maintain the safety of the solutions. Those maneuvers which affect the probability of collision —$c^{a:b}$, in Alg. 4:14— are discarded. This is obtained from the cost of the trajectory, as detailed in Algorithm 3.

Figure 3.4 illustrates how this approach discards the longest maneuvers in the vicinity of obstacles due to their probability of collision, while navigating in uncluttered areas causes the acceptance of the first explored candidates. This graduated fidelity approach results in a lattice with a considerably lower density of states and maneuvers except in those areas in which complex maneuvering is required for obstacle avoidance. Consequently, the efficiency of the planner is considerably improved, while its performance —the cost of solutions— is barely affected.

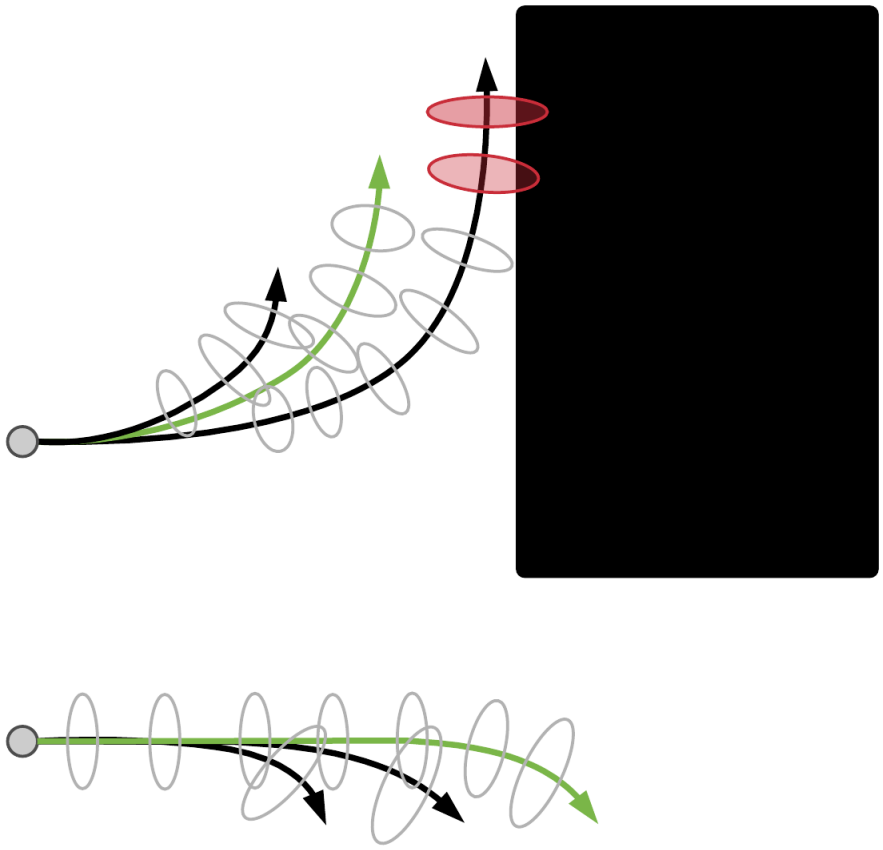


Figure 3.4: Example of the selected candidates with the graduated fidelity approach in different situations —in green. Long trajectories are discarded when they affect the probability of collision with obstacles —in black—, while in free space they are selected to have a lower fidelity.

3.2.3 Multi-resolution heuristic

As mentioned in Section 3.1, our approach combines two heuristics: one takes into account the kinematic restrictions in free space, FSH, while the other only considers the obstacles in the environment, H2D. In this section H2DMR, a multi-resolution heuristic based on the latter, is proposed. Unlike H2D, this novel heuristic takes advantage of the octree structure of the map to obtain a multi-resolution grid, resulting in improved efficiency and scalability.

Algorithm 5 details how H2DMR is calculated. The process is done applying Dijkstra's algorithm to obtain a multi-resolution grid. The inputs are the initial state of the robot and the goal, x^0 and x^G. Since the heuristic uses positions instead of states, their counterparts —p^0 and p^G— are obtained in Algorithm 5:4-5. The grid is generated backwards, starting in p^G. Thus, the resulting grid will contain the estimated cost between each point and the goal, which is used as the heuristic value for planning.

Iteratively, the point with the lowest cost from the start —p, in Alg. 5:9— is selected and its successors obtained. κ is the cell of the map containing the selected point —Alg. 5:10. To generate its successors in accordance with the resolution of the map, the adjacent cells of κ —$adjacent(\kappa)$, in Alg. 5:11— are explored.

Figure 3.5 details how the resolution of the octree is taken into account when obtaining the neighbors of p. Given the size of a cell —s, in Alg. 5:12— and the highest fidelity of the motion primitives, f^+, two situations are considered: on the one hand, a cell is split into subcells when f^+ exceeds the resolution of the map, in order to keep the accuracy of the heuristic —Alg. 5:14-17. On the other hand an upper bound in the resolution was introduced to avoid generating neighbors with a distance lower than f^+ —Alg. 5:19-22. Therefore, the size of the cells this heuristic works with is in fact limited according to f^+ —Alg. 5:19. In both cases the resulting neighbors are obtained from the center of the selected cells — $position(\kappa'')$, in Alg. 5:15 and Alg. 5:20.

Finally, the cost between p and each neighbor p' —the distance between them— is obtained and p' is introduced into the *OPEN* queue to be explored by the algorithm later. Collisions are checked using the inscribed circle in the robot shape, also called optimistic shape, in Algorithm 5:25. Thus, the optimistic nature of the heuristic is maintained.

The stopping condition of the algorithm is to expand a point with a cost which doubles the one between p^0 and p^G —Alg. 5:23. Those areas of the map left outside the generated grid are not interesting for planning due to their distance to the most promising path. H2D uses this same stopping condition, which was introduced by [123] for efficiency purposes.

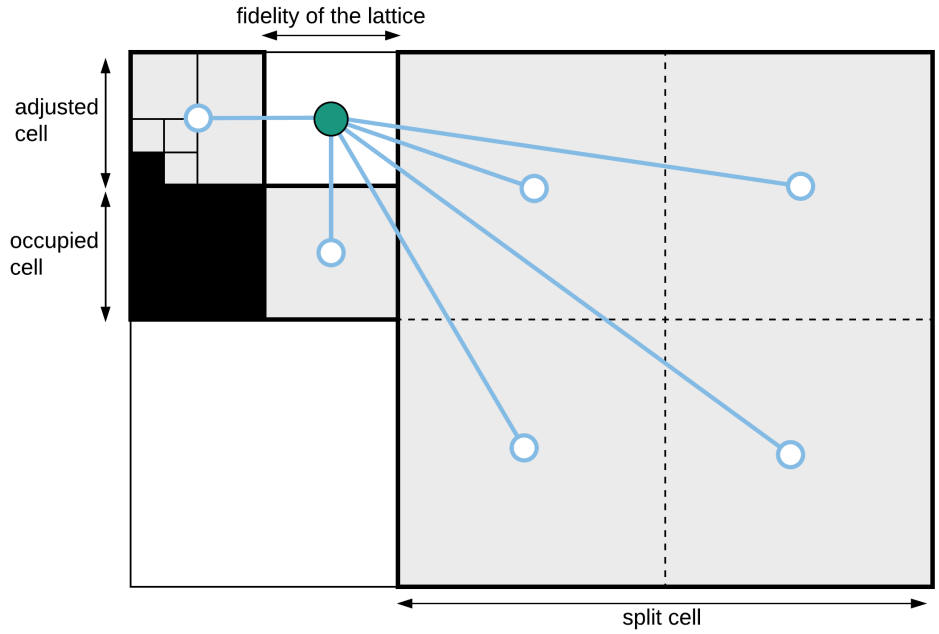


Figure 3.5: Neighborhood of H2DMR —in blue. Given a point —in green—, those cells adjacent to the one containing it are explored —in gray. When a cell is smaller than the highest fidelity of the lattice —f^+—, a bigger one containing it is selected. On the contrary, a cell is split into subcells when its size exceeds f^+.

This algorithm allows the obtention of a multi-resolution grid which contains the cost between each point and the goal, which is used by AD* as heuristic. Unlike H2D, this grid takes into account both the resolution of the octree map and the highest fidelity of the motion primitives. Thus, this approach outperforms H2D in the number of iterations required to explore the map, and consequently the time spent in initializing the heuristic. This is specially noticeable in large environments, since H2DMR scales better than H2D due to its capability to use lower resolutions in uncluttered areas.

Since the positions of the multi-resolution grid and the states in the lattice do not directly match, obtaining the heuristic value for a state x^a is done as follows: first, the octree cell containing it is retrieved, and then all positions of the grid within this cell and the adjacent

Algorithm 5 Obtention of H2DMR

Require: f^+, highest fidelity of the lattice

1: **function** HEURISTIC(x)
2: **return** $\max(h2dmr(x), fsh(x))$
3: **function** INITIALIZEHEURISTIC(x^0, x^G)
4: $p^0 = position(x^0)$ ▷ Initial position
5: $p^G = position(x^G)$ ▷ Goal position
6: $c(p^G) = 0$
7: $OPEN = \{p^G\}$
8: **repeat**
9: $p = \arg\min_{p \in OPEN} c(p)$
10: $\kappa = cell(p)$ ▷ Get map cell containing p
11: **for all** $\kappa' \in adjacent(\kappa)$ **do** ▷ Iterate over those cells adjacent to κ
12: $s = size(\kappa')$ ▷ Size of cell κ'
13: **if** $s > f^+$ **then**
14: **for all** $\kappa'' \in subcells(\kappa')$ **do** ▷ Split κ' into subcells
15: $p' = position(\kappa'')$ ▷ Get center of cell κ''
16: $c(p') = c(p) + \text{COSTH}(p, p')$
17: $OPEN = OPEN \cup \{p'\}$
18: **else**
19: $\kappa'' = adjust(\kappa', f^+)$ ▷ Adjust size of κ' to f^+
20: $p' = position(\kappa'')$ ▷ Get center of cell κ''
21: $c(p') = c(p) + \text{COSTH}(p, p')$
22: $OPEN = OPEN \cup \{p'\}$
23: **until** $c(p) > 2 \cdot c(p^0)$
24: **function** COSTH (p, p')
25: **if** $collision_0(p')$ **then** ▷ Optimistic shape
26: **return** ∞
27: **else**
28: **return** $\|p' - p\|$

ones are explored. The heuristic of the state is given by the position —p— which minimizes the sum of its cost —$c(p)$— and the distance to x^a:

$$h2dmr(x^a) = \arg\min_p(\|x^a - p\| + c(p)) \tag{3.11}$$

Heuristics play a significant role in the anytime search capabilities of AD*, since their value is scaled by the parameter ε. By doing so, a sub-optimal solution can be retrieved faster and then improved iteratively until the optimal one is found or the available computing time is used up. The planner takes advantage of this possibility by adjusting the quality of the solution in terms of traversal time. However, the safety of the solutions is not affected by the use of anytime search. This is because the heuristic only estimates the traversal time of the path, even though the cost function has three elements —probability of collision, traversal time and uncertainty at the goal. Since anytime search works inflating the heuristic and these elements are compared hierarchically —see Eq. 3.10—, the probability of collision is always minimal regardless the value of ε.

3.3 Results

In this section results of the proposed motion planner in different scenarios and uncertainty conditions are reported. Moreover, tests varying the robot shape were run, showing the relevance of taking it into account to predict the probability of collision along the paths. Also, results for the proposed graduated fidelity approach are detailed, comparing them with those of a standard state lattice planner. Thus, we show the ability of the proposed method to improve the efficiency while maintaining the performance —the cost of the solutions. Finally, we detail results for H2DMR, the proposed multi-resolution heuristic, focusing on its validity and the improved efficiency, specially in large scenarios. Runtimes reported in this section are for a computer with a CPU Intel Core^{TM} i7-4790 at 3.6 GHz and 16 GB of RAM.

All tests were run on a 2D world and a robot with Ackermann dynamics, in which $M_t = 0.01 \cdot I$. Robot dimensions are 3.0×0.75 m, with the rotation center located at 0.9 m from its back side, centered in the short axis. Linear speed ranges between 0 and 0.5 m/s, and the angular speed is between -30 and 30 deg/s. The state vector is 5-dimensional and contains the pose of the rotation center of the robot, and also the linear and angular speeds:

$$[x, y, \theta, v, \omega] \tag{3.12}$$

The control vector contains the linear and angular speeds. These commands are sent to the robot at 3 Hz. The measurements are the position and the heading, with an uncertainty of $N_t = 0.01 \cdot I$ in normal conditions. However, there are location denied areas in the environments in which the robot does not receive any measurement. Within them, sensing uncertainty is $N_t = \infty \cdot I$.

With regard to the dynamics model, it was learned from 183 s of navigation data for a simulated robot in Gazebo. Then, a set of 336 motion primitives was obtained —shown in Figure 3.6. These primitives connect states of distances 1, 2, 4, 8 and 16 in a lattice with a highest fidelity, f^+, of 0.5 m.

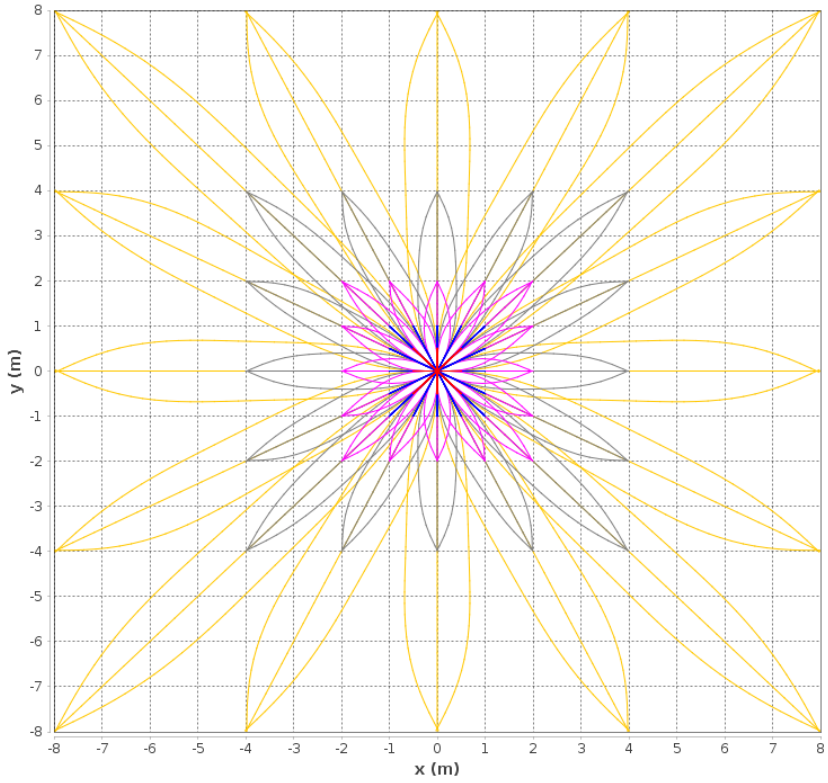


Figure 3.6: Control set used in the experiments: 336 trajectories connecting neighbors of levels 1, 2, 4, 8 and 16 —in red, blue, pink, gray and yellow, respectively. The highest fidelity, f^+, is 0.5 m.

Figure 3.7 shows a plan obtained with the proposed algorithm. Areas with high uncertainty exist in the region where maneuvering is required to enter the corridor, which together with the robot dimensions makes difficult to find a safe path trough it. In spite of this, the planner provides a solution which is not only optimal but also smooth. This contrasts with other approaches based on RRT and RRT*, which rely on smoothing techniques to achieve similar results, affecting the predicted uncertainty.

Figure 3.8 details the behavior of the planner under different uncertainty conditions. In this environment, several alternatives are available to reach the goal, but the maneuvering is limited due to the robot length. Moreover, motion uncertainty has a great impact in the front side of the robot, since any small deviation in heading can result in a collision. The chosen path depends on the PDFs along the different candidates, as the planner favors the safety of

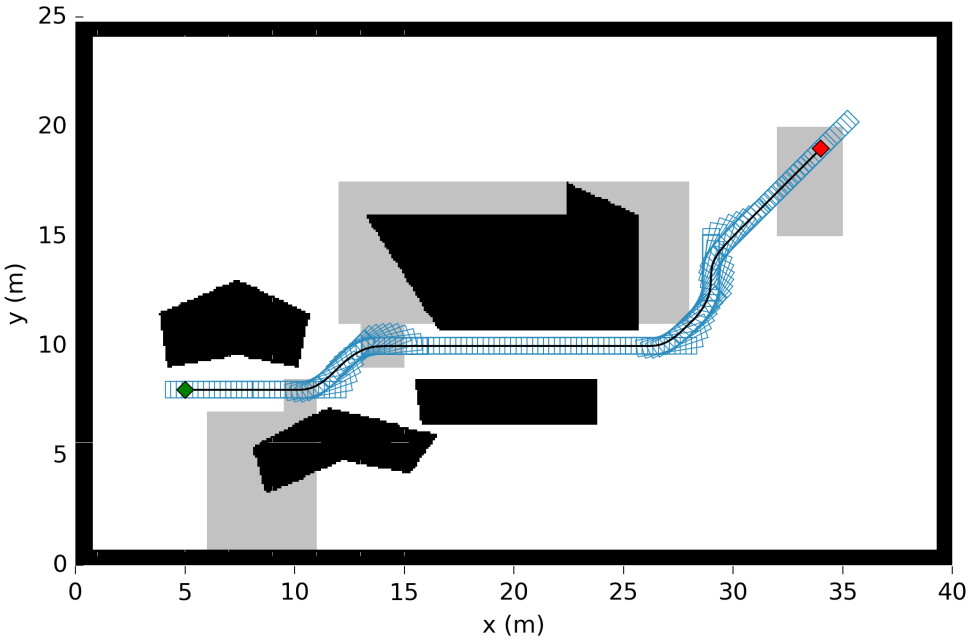
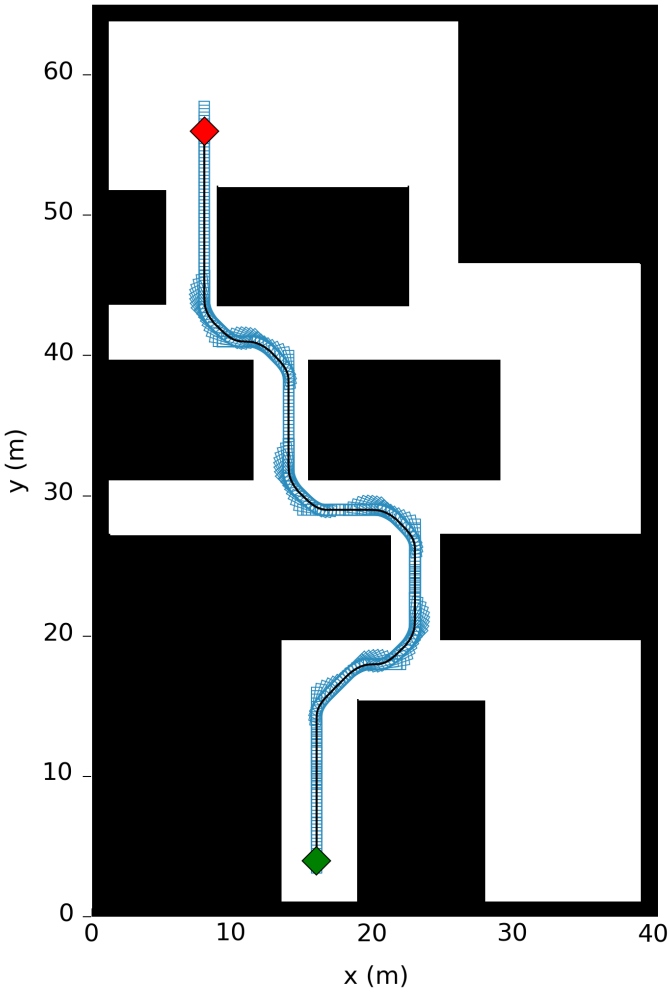
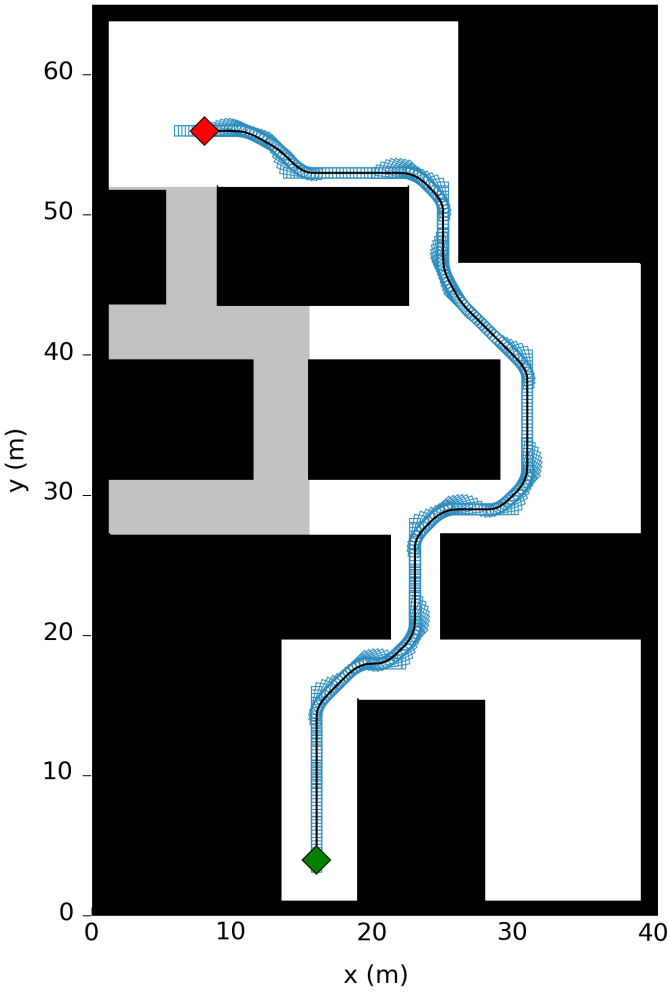


Figure 3.7: Planned path for a car-like robot with dimensions 3.0 × 0.75 m in a cluttered environment. Obstacles are in black, while regions in light gray are those with no location signal. The maximum-likelihood path is represented with a black line, with the robot trail in light blue. The green and red diamonds are the start and the goal points.

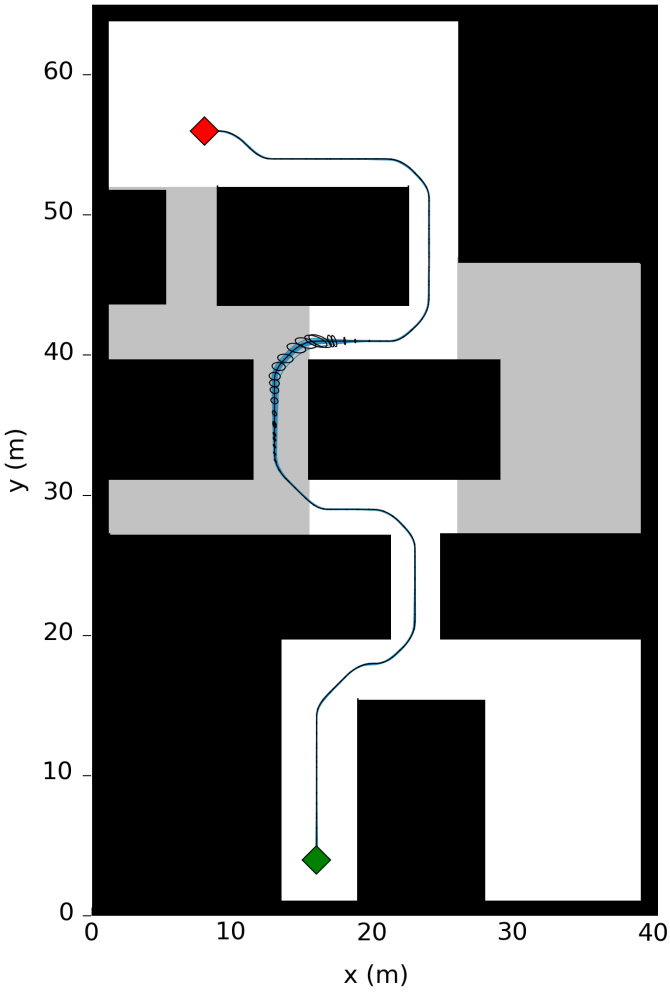


(a) Path without location denied areas.

Figure 3.8: Planned paths for a car-like robot of 3.0 × 0.75 m under different uncertainty conditions. The robot trail is in blue, the maximum-likelihood path is in black, and the diamonds in green and red are the start and the goal points.



(b) Selected path when two of the corridors are location denied areas.



(c) Solution obtained when the corridors of both sides are location denied areas. Here the black ellipses are 3·Σ of the predicted PDFs.

the solution rather than its traversal time. The optimal solution is shown in Figure 3.8(a), in which there are no location denied areas and motion uncertainty only arises from the controls. A safe plan that crosses the two narrow corridors is obtained, maneuvering only in those areas in which there is enough space to avoid collisions.

Figure 3.8(b) shows the solution for the same environment but different uncertainty conditions. Here the probability of collision of the prior path is affected, as the motion uncertainty grows due to the location denial in the corridors of the left. Therefore, a safer alternative which avoids this area is selected despite increasing the traversal time. Having another location denied region in the right of the map, as shown in Figure 3.8(c), results in a variation of this path. In this case, the planner minimizes the probability of collision avoiding to maneuver just after leaving the location denied area in the corridor of the right. Instead, the planned path crosses the first narrow corridor and allows the robot to receive measurements before turning.

Figure 3.8(c) shows in black the predicted PDFs, which explain the solution obtained. The uncertainty grows when the navigation is done without receiving measurements; thus, the distance to obstacles must be higher to maintain the safety of the path. After leaving those areas, the uncertainty shrinks due to the measurements and the use of a LQG controller, which is taken into account when predicting the distributions at planning time. Hence, the path can go through the final obstacles and safely reach the goal.

As mentioned before, Figure 3.8(c) shows in black the uncertainty estimated at planning time. The ellipses represent $3 \cdot \Sigma$ of the PDFs, a 99.7% of confidence. Moreover, $1,000$ simulations of each planned path were run to estimate the real behavior of the robot, and also to compare the predicted distributions with the real ones. These simulations allow the comparison of the probability of collision obtained from the PDFs, $\hat{p}_c$, and the one from the simulated paths, p_c, in which collisions were checked with the real shape.

In Figure 3.9 the behavior of the planner in a map with high uncertainty is shown. In this experiment the robot has a significant uncertainty at the beginning of the path —$\Sigma_{x^0} = I$—, and the only area in which measurements are received is far away from the obstacles. Directly crossing the door following the shortest path, without taking into account the uncertainty conditions, is too risky and it would result in a plan with a 55% of probability of collision. Instead, our planner obtains a path which goes outside the location denied area to diminish the uncertainty and ensure that the robot safely crosses the door. In fact, our planner is conservative in this matter and considers that the robot only receives measurements when there is no overlap between the PDFs and the location denied areas.

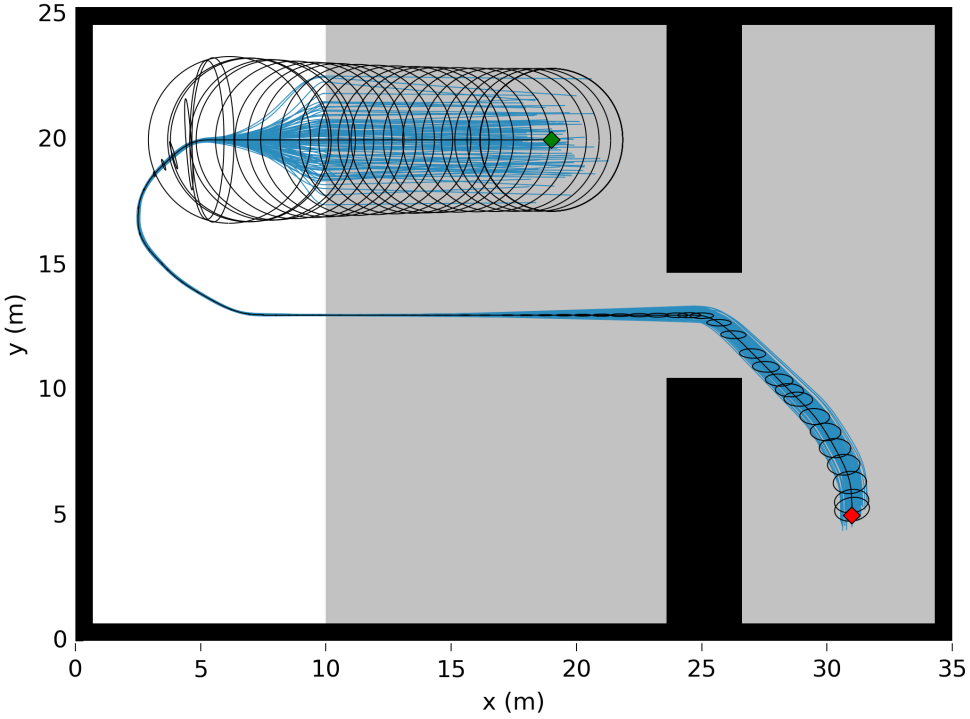


Figure 3.9: Planned path for a car-like robot of 3.0 × 0.75 m in an environment with high uncertainty. The nominal path is in black, while the simulated executions are in blue. The ellipses are 3·Σ of the predicted PDFs. The robot receives measurements only when the PDF is completely outside the gray region.

Prior works already reported planning results in this environment, so we used it to compare the performance of our proposal with them. The approach of [26] finds a valid solution in these conditions and converges in approximately 17,500 iterations, with a planning time of 40 s. As reported in that work, the method of [197] fails in this environment due to the Voronoi-bias that prevents the expansion of the paths going through the area in which the robot can receive measurements. Our approach outperforms these results and finds the best solution in 2.18 s and 684 iterations for our dynamics model[3].

The approach of [199] finds an initial path and then obtains a locally-optimal control

[3] [26, 199] use a punctual robot with 2D dynamics, while our results are for a car-like robot.

policy. The convergence of their method took 3.65 s, and the 93% of the simulated executions were collision free. On the contrary, for the plans obtained with our proposal the simulations showed that in all cases the robot was able to safely cross the narrow passage, also varying the starting point. Their method does not assume a fixed control gain along each section of the planned path. However, with their approach a different control policy has to be obtained for each homotopy class of trajectory in order to find the globally optimal solution. As a consequence, the planning times would be higher in more complex environments —i.e. the map of Figure 3.8 already contains 4 homotopy classes for the given start and goal points. Table 3.1 contains the detailed results for all the environments in this section. In all cases the planner finds the optimal solution in a few seconds.

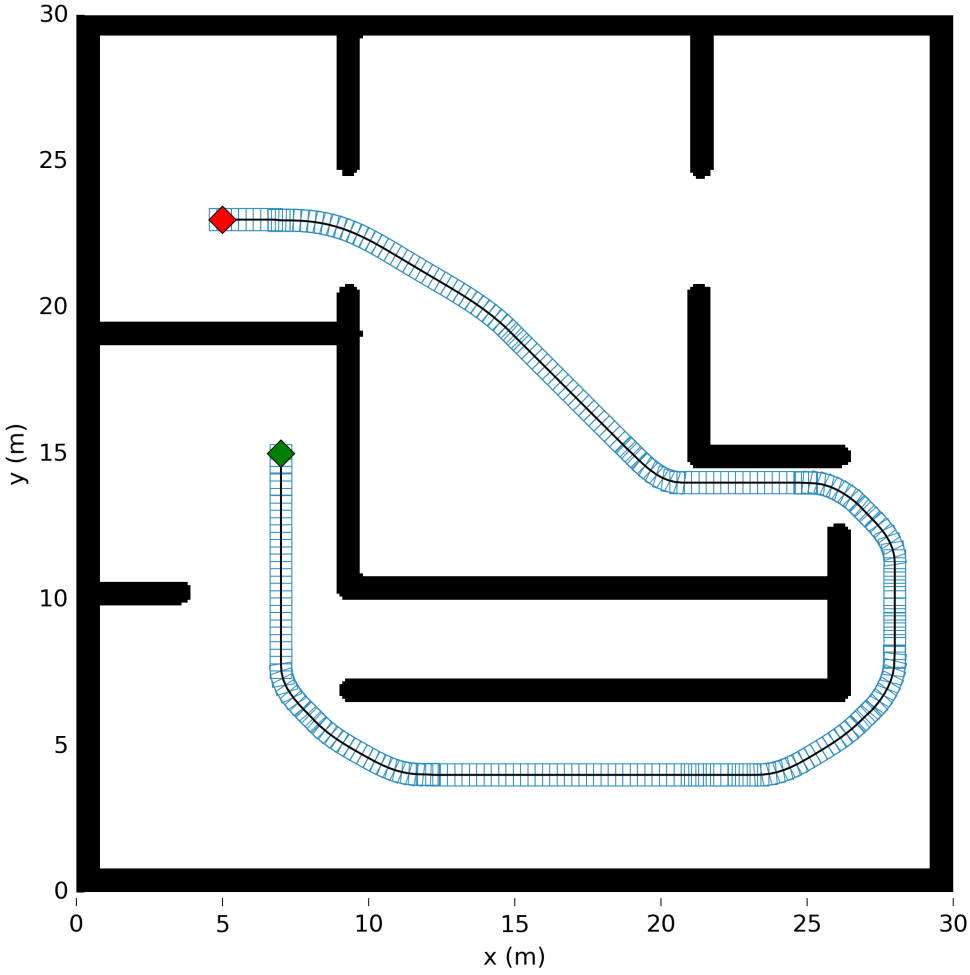
Table 3.1: Planning results for the experiments in Section 3.3 using the graduated fidelity lattice and the heuristic H2DMR. Robot dimensions are 3.0×0.75 m. The planner was run with $\varepsilon_0 = 1.5$ and decreasing it iteratively. Column ε has the value for which the optimal solution was found. Columns "Iterations", "Insertions" and "Time" detail the planning efficiency —nodes expanded by AD*, nodes inserted in the *OPEN* queue and planning time. "Cost" is the traversal time. The estimated probability of collision ($\hat{p}_c$) and the real one (p_c), obtained from 1,000 simulations, are 0 in all cases. All times are in seconds.

		Planning			Solution
Problem	ε	**Iterations**	**Insertions**	**Time (s)**	**Cost (s)**
Fig. 3.7	1.05	228	1,043	0.38	68.66
Fig. 3.8(a)	1.5	252	639	0.32	132.02
Fig. 3.8(b)	1.5	659	2194	0.85	157.37
Fig. 3.8(c)	1.5	799	1,872	2.26	172.28
Fig. 3.9	1.03	684	2,595	2.18	108.27
Fig. 3.10(b)	1.06	611	1,559	0.96	131.14

3.3.1 Reliable collision check

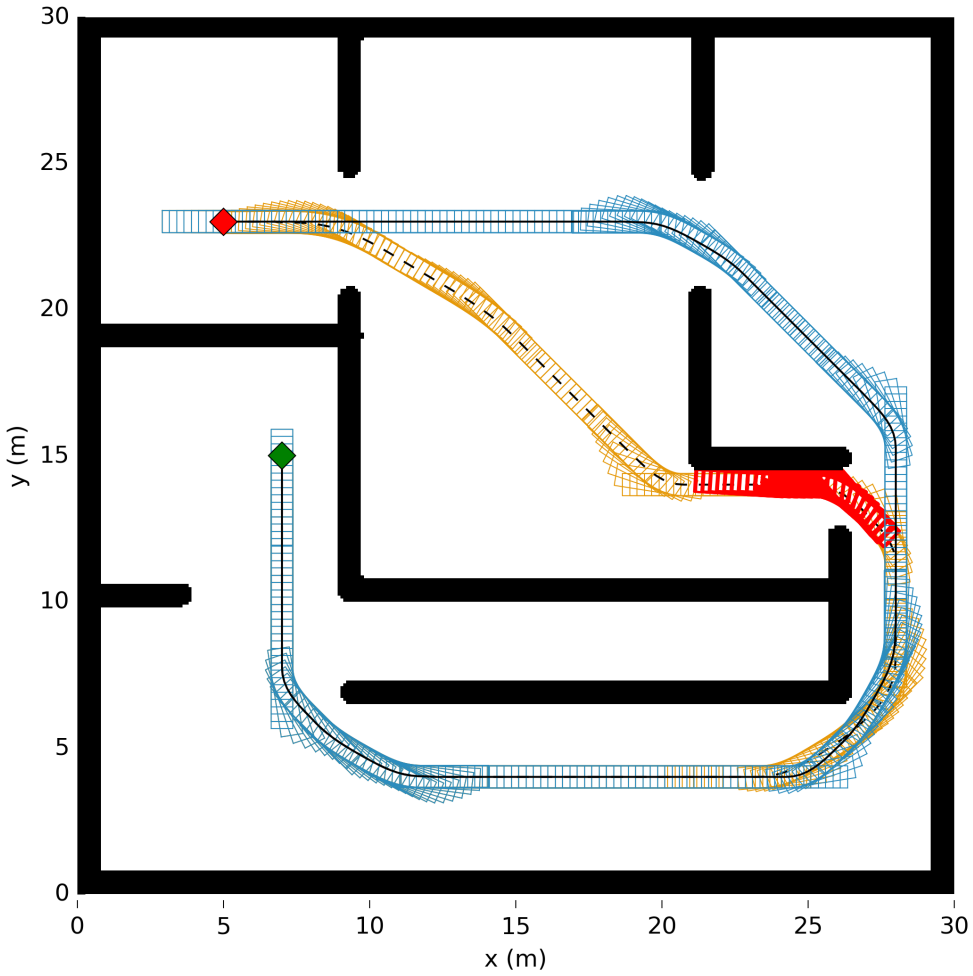
The capability of the planner to work with different robot shapes is shown in Figure 3.10. In this example the planner was run in the same environment and uncertainty conditions, but using two different shapes, a squared one with size 0.75×0.75 m, and another one with size 3.0×0.75 m. Approximating the former by a circle could be a reasonable assumption, but for the latter this cannot be done without drawbacks.

The proposed method accurately estimates the probability of collision of the path in accordance with the real shape, which leads to the obtention of different paths in the example of Figure 3.10. Figure 3.10(a) shows the motion plan obtained for a squared robot of



(a) Plan for a shape of 0.75 × 0.75 m.

Figure 3.10: Influence of the robot shape in the planned paths.



(b) In blue, the path for a larger robot, of 3.0 $\times$ 0.75 m, is shown. The path that this robot would follow if its real dimensions were not taken into account is depicted is shown in orange and, in red, the most probable collisions in that situation are highlighted. The diamonds in green and red are the start and the goal points.

0.75×0.75 m. This robot can safely pass through the room at the center of the map after maneuvering in the narrow passage. However, for the robot of 3.0×0.75 m this maneuver would be too risky due to its dimensions. In fact, the robot collides with its right front corner even before crossing the door, while afterwards the collision involves all its right side. Our motion planner manages that and obtains for this robot a slightly longer path, but with the minimal probability of collision, shown in blue in Figure 3.10(b). Table 3.1 details the planning results for the 3.0×0.75 m shape.

This example shows the relevance of taking into account the real shape, which is systematically disregarded in the state of the art. In this environment, approximating the large robot by the inscribed circle —with a diameter of 0.75 m— would result in underestimating the probability of collision, which would lead to unplanned collisions, as shown in red in Figure 3.10(b). The same happens to chance-constraint planning only using the PDFs. On the contrary, with the outer circle the probability of collision of the paths would be highly overestimated.

In Figure 3.11 we show that our approach is able to reliably estimate the probability of collision and find safe paths regardless the robot shape. In this experiment the robot shape is a "T", which makes more difficult to maneuver in narrow spaces. The planned path is similar to that of Figure 3.10(b), since the robot cannot enter through the first door due to its dimensions. However, in this case the robot has to enter the corridor completely aligned with the walls to avoid collisions with them. Similarly, the robot has to take the curve wide before crossing the doors at the top of the map. The planner finds the optimal solution in 1.38 s —525 iterations. The cost is 136.89 s, slightly higher than in the experiment of Figure 3.10(b).

Whatever the kind of environment, the uncertainty in heading clearly influences the probability of collision for those robots with non-circular shapes. This is aggravated if, due to the robot dimensions, like the one in the example, slight changes in heading cause large variations in the distance to the obstacles. This is managed by the proposed method, since the probability of collision is obtained by sampling PDFs in a way that not only takes into account deviations in position, but also in heading. The estimated probability of collision, $\hat{p}_c$, is correct regardless the robot dimensions, and it approximates well the real value, p_c, obtained from $1,000$ simulations of the planned path. Thus, reliable and safe paths are found for any shape.

Fig 3.12 compares the estimation error of our method with that of other approaches [120, 197] which approximate the robot by its bounding circle. They estimate the probability of collision using the regularized gamma function, $\Gamma(n/2, r^2/2)$, which gives the probability

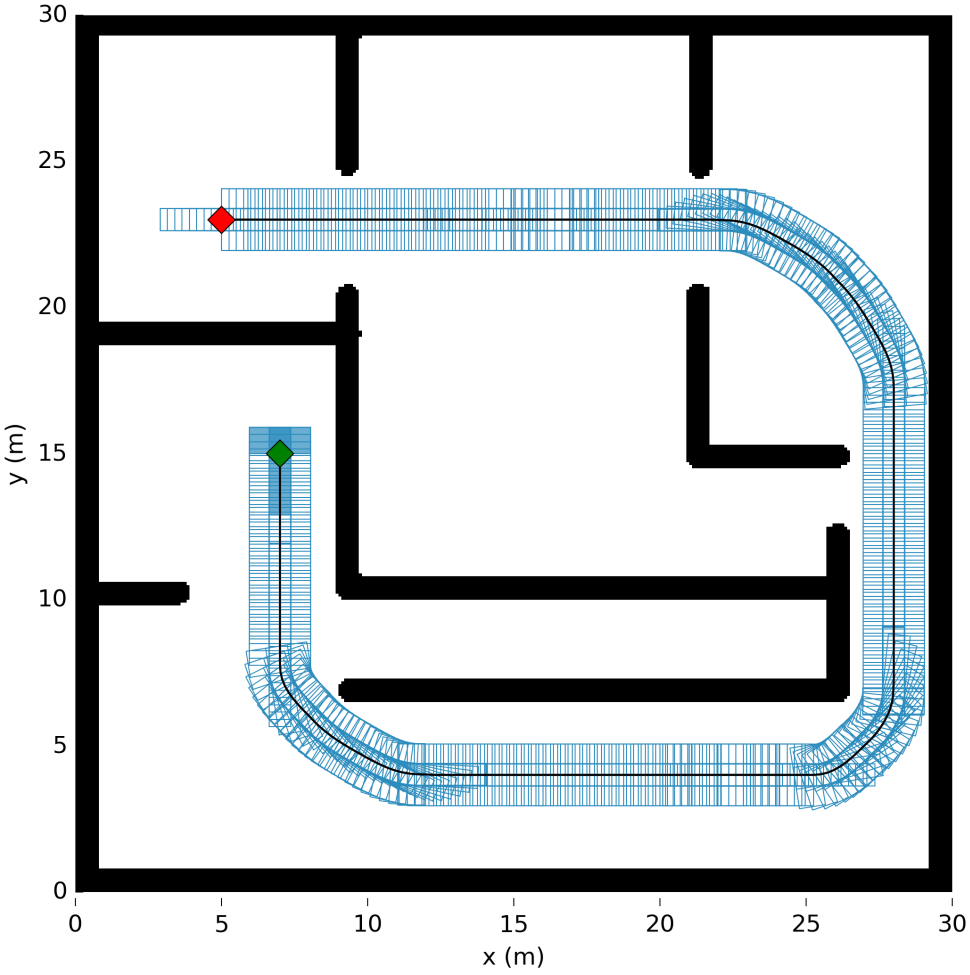


Figure 3.11: Planned path for a robot with "T" shape, of 3.0 × 2.2 m, shown in blue. The diamonds in green and red are the start and the goal points.

that a sample is within r standard deviations for a multivariate Gaussian distribution of dimension n. A challenging situation for planning is when a non-circular robot is approaching a diagonally placed obstacle —i.e. entering a corridor or turning a corner. In these situations, the reliability of the gamma function decreases due to the fact that the robot was approximated by a circle and the uncertainty in heading was not taken into account. This results in a significant error of the estimated probability of collision with respect to the real one. Instead, our method accurately approximates it with only a 1.5% of average error.

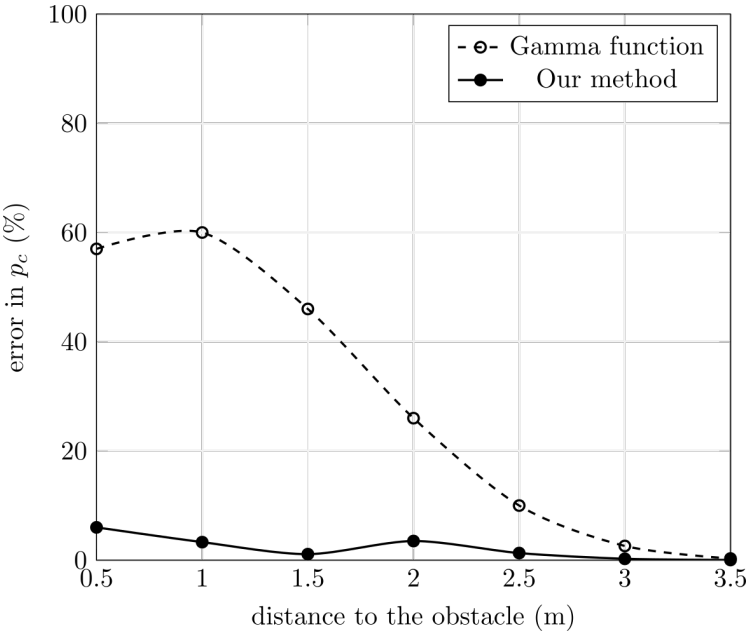


Figure 3.12: Error of the probability of collision estimated with our method compared with that of the gamma function. The error was measured varying the distance between a robot with shape 3.0 $\times$ 0.75 m and a single obstacle placed diagonally with it, and for $\Sigma = I$.

3.3.2 Graduated fidelity lattice

Most of the planning time, approximately a 80%, is spent in propagating the PDFs through the candidate paths and estimating their probability of collision. Diminishing the fidelity of the lattice allows to reduce the number of paths contained in the state lattice, obtaining better computation times. However, doing it in a general way affects the robustness of the system

and its capability to find near-optimal solutions. On the contrary, the proposed graduated fidelity approach allows to drastically reduce the number of analyzed paths, highly increasing the efficiency of the planner while keeping nearly the same performance.

Figure 3.13 shows the lattice for the motion plan in Figure 3.7 and the results of the graduated fidelity approach. The highest fidelity is selected in those areas with more density of obstacles, such as the corridors. In those areas the octree cannot be compacted due to the different occupancy information of adjacent cells, and therefore cells tend to be small. This leads to the selection of the highest fidelity, f^{+}, using the motion primitives with the minimum length, 0.5 m. Similarly, higher fidelities are selected when the uncertainty affects the probability of collision of the paths —i.e. maneuvering in the vicinity of obstacles.

The opposite occurs in the uncluttered areas of the map. In those regions lower fidelities are selected, which significantly reduces the density of states in the lattice. Moreover,

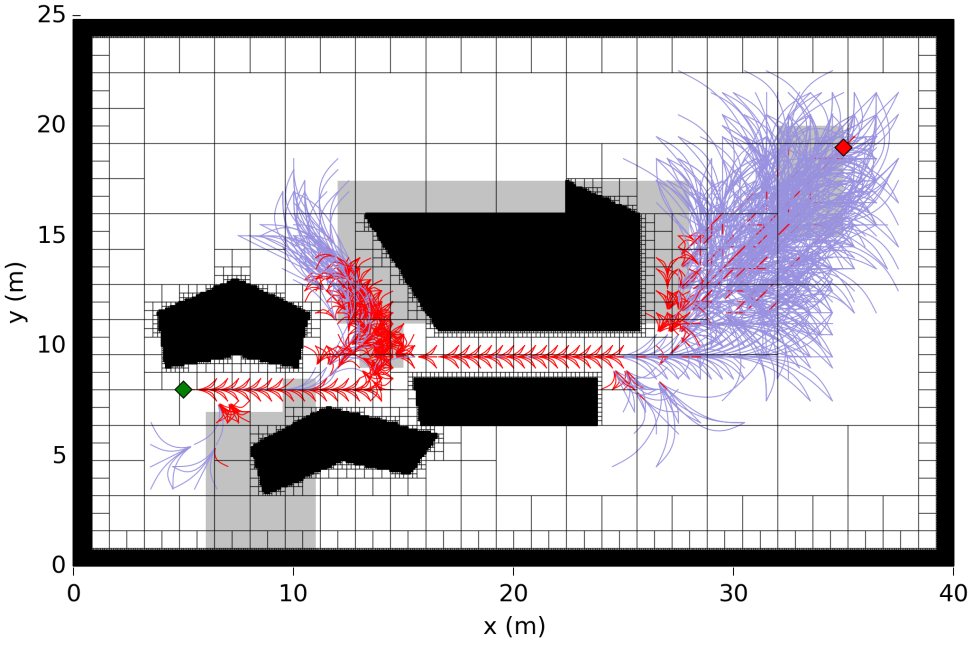


Figure 3.13: Graduated fidelity selection for an environment combining cluttered and non-cluttered areas. Areas making use of the highest fidelity are in red, while those with lower ones are in blue. This lattice corresponds to the motion plan of Figure 3.7.

reducing the maneuverability in those areas has a limited influence in the cost of the solutions retrieved, while saving significant computational resources. Only near the goal an exception is made, and the whole set of actions $\mathscr{U}$ is used to facilitate maneuvering in this area regardless the density of obstacles.

Table 3.2 shows a comparison between the executions of the proposed planner with graduated fidelity, GF, and those of a standard state lattice planner. For the latter, the connectivity is given by the whole set of primitives.

This comparison shows a significant reduction in the number of iterations of the search algorithm —an average decrease of 83.3%—, but more importantly in the number of insertions in the *OPEN* queue —reduced in a 88.1%. The latter highly influences the planning time, as the insertions involve propagating the PDFs and estimating the probability of collision along the candidate paths. Thus, the planning time diminished on average by 88.5% when making use of the graduated fidelity lattice. Conversely, the cost of the solutions slightly increases —an average of 6.2%— due to the selection of longer maneuvers for turning. However, when compared with the solutions provided without the graduated fidelity approach, the differences are minimal.

Table 3.2: Performance comparison between the planner using the graduated fidelity lattice (GF) and a standard one. All tests were run with $\varepsilon = 1.0$. Robot shape is 3.0×0.75 m. Columns "Iterations", "Insertions" and "Time" detail the planning efficiency —nodes expanded by AD*, nodes inserted in the *OPEN* queue and planning time. "Cost" is the traversal time. The estimated probability of collision ($\hat{p}_c$) and the real one (p_c), obtained from 1,000 simulations, are 0 in all cases. All times are in seconds.

		Planning			Solution
Problem	**GF**	**Iterations**	**Insertions**	**Time (s)**	**Cost (s)**
Fig. 3.7	✓	144	586	0.59	68.66
Fig. 3.7	✗	272	1,446	1.58	68.04
Fig. 3.8(a)	✓	318	774	0.84	132.02
Fig. 3.8(a)	✗	1,501	6,287	5.25	127.19
Fig. 3.8(b)	✓	820	2,093	1.54	154.71
Fig. 3.8(b)	✗	2,957	12,346	12.96	146.91
Fig. 3.8(c)	✓	2,030	4,522	4.52	172.28
Fig. 3.8(c)	✗	7,277	26,942	25.90	160.55
Fig. 3.9	✓	1,302	2,868	2.68	108.27
Fig. 3.9	✗	15,952	42,594	45.38	95.96
Fig. 3.10(b)	✓	593	1,321	1.29	131.14
Fig. 3.10(b)	✗	3,196	12,183	8.53	126.27

3.3.3 Anytime search

This section focuses on the capabilities of combining anytime search with a graduated fidelity lattice. This feature allows obtaining sub-optimal solutions faster and refine them later, which is desirable when using the planner in real world domains.

To test this feature the planner was run with $\varepsilon_0 = 1.5$, and later the parameter was iteratively decreased to refine the solution. Thus, an initial plan is retrieved in a very short time and the robot can move while the plan is improved. Table 3.3 details the results of these tests.

Table 3.3: Anytime planning performance. Planner was run starting in $\varepsilon_0 = 1.5$, decreasing it iteratively. Column ε has the value for which the optimal solution was found. Robot shape is 3.0 $\times$ 0.75 m. Columns "Iterations", "Insertions" and "Time" detail the planning efficiency —nodes expanded by AD*, nodes inserted in the *OPEN* queue and planning time. "Cost" is the traversal time. The estimated probability of collision ($\hat{p}_c$) and the real one (p_c), obtained from 1,000 simulations, are 0 in all cases. All times are in seconds.

		Planning			Solution
Problem	ε	**Iterations**	**Insertions**	**Time (s)**	**Cost (s)**
Fig. 3.7	1.5	92	484	0.15	68.78
Fig. 3.7	1.05	228	1,043	0.38	68.66
Fig. 3.8(a)	1.5	252	639	0.32	132.02
Fig. 3.8(b)	1.5	659	2194	0.85	157.37
Fig. 3.8(b)	1.15	1,032	3,152	1.40	154.71
Fig. 3.8(c)	1.5	799	1,872	2.26	172.28
Fig. 3.9	1.5	466	1,164	1.57	112.79
Fig. 3.9	1.03	684	2,595	2.18	108.27
Fig. 3.10(b)	1.5	64	225	0.12	132.98
Fig. 3.10(b)	1.06	611	1,559	0.96	131.14

The cost of the sub-optimal solutions is bounded by the value of ε, so the cost of the solution decreases in accordance with the value of the parameter. Also, the higher its value, the lower the number of iterations and insertions, and consequently the planning time. For $\varepsilon = 1.0$ the solution retrieved is guaranteed to be optimal. While this applies to the traversal time of the path, the probability of collision is always minimized —$\hat{p}_c$ is always minimal, even for anytime solutions. This is because anytime search works scaling the value of the heuristic by ε, which only estimates the traversal time, not the rest of the elements of the cost function, as detailed in Section 3.2.3.

It is specially noticeable the reduction in the planning times for those cases in which the heuristic is too optimistic with respect to the real cost of the path —due to the uncertainty

conditions, e.g. in Figure 3.9. In those cases, finding the optimal solution is not trivial and anytime search avoids long waiting times. However, while increasing ε_0 speeds up the obtention of the initial solutions, it might lead to longer computational times if too many values of ε are tested. This is because each time ε changes, the priorities of all the nodes in *OPEN* have to be updated, which can be costly depending on its number. Therefore, if there are too many nodes in *OPEN* it might be better to obtain a new plan from scratch, as pointed out in [125].

3.3.4 Planning in real environments

In this section we report results for a large real environment —292×167 m. It is a map of the Freiburg University campus built from 3D laser scans, and it was obtained from a public dataset[4]. The location denied areas are located around the obstacles to simulate the effect that they have on the GPS signal. Concretely, for this experiment we considered that the space of 2 m around the obstacles has high uncertainty. The motion model was that of an odometric robot. The set of motion primitives has 246 motions which connect the lattice states of distances 0, 1, 2, 4, 8 and 16. The highest fidelity, f^+, is 1.0 m.

Figure 3.14 shows the optimal solution for this environment. Our approach finds a valid path, and the robot is able to safely maneuver between the trees of the right. Its cost was 359.28 s and it was obtained in 34 s of planning time, after $26,972$ iterations, for $\varepsilon = 1.05$. Due to the anytime search capabilities of AD*, a first sub-optimal solution is available after 1.1 s of planning time, in 823 iterations, with a cost of 402.20 s for $\varepsilon = 1.5$.

The cluttered areas of the map do not affect the ability of the planner to find the shortest path. Nevertheless, the runtime of the planner might be slightly higher than in other experiments due to the size of the map and the cluttered areas.

Finally, results for an experiment with a real robot are reported. The hardware platform was Pepper, manufactured by Softbank Robotics[5]. The motion primitives were obtained taking into account the dynamics model, which was obtained from 512 s of navigation data. A set of 236 actions with lengths between 0.2 and 1 m allowed the robot to move with the kinematic restrictions of an odometric motion model, with a maximum linear speed of $0.5\,\mathrm{m/s}$.

Figure 3.15 shows the testing set of this experiment. The map was built using a SLAM algorithm [63], and a Monte Carlo method [47] was used to localize the robot in it. The accuracy of the localization decreases when the distance to the obstacles exceeds the range

[4] Available at `http://ais.informatik.uni-freiburg.de/projects/datasets/octomap/`
[5] `https://www.softbankrobotics.com/emea/en/pepper`

of the laser sensor —1.5 m for Pepper—, e.g. near the start or after waypoint 2. This causes deviations that are corrected by the LQG controller. Despite this, the planned path was safe, and no collisions were reported in 20 executions. Finally, the efficiency of the planner allowed the robot to operate without noticeable delays. On average, the plans between consecutive waypoints were obtained in less than 5 seconds and 1,000 iterations due to our graduated fidelity approach and the anytime search capabilities of AD*. Also, H2DMR was obtained in 0.2 s —941 iterations. The total cost of the path was 115 s.

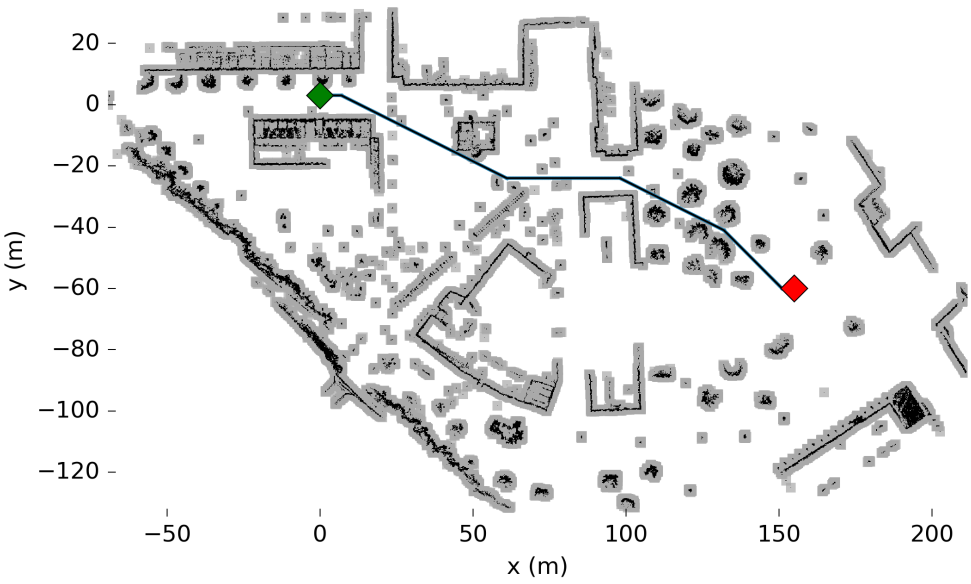


Figure 3.14: Planned path for an odometric robot of 0.75 × 0.75 m in a real environment. Obstacles are in black, while regions in light gray are those with no location signal. The robot trail is in blue and the diamonds in green and red are the start and the goal points.

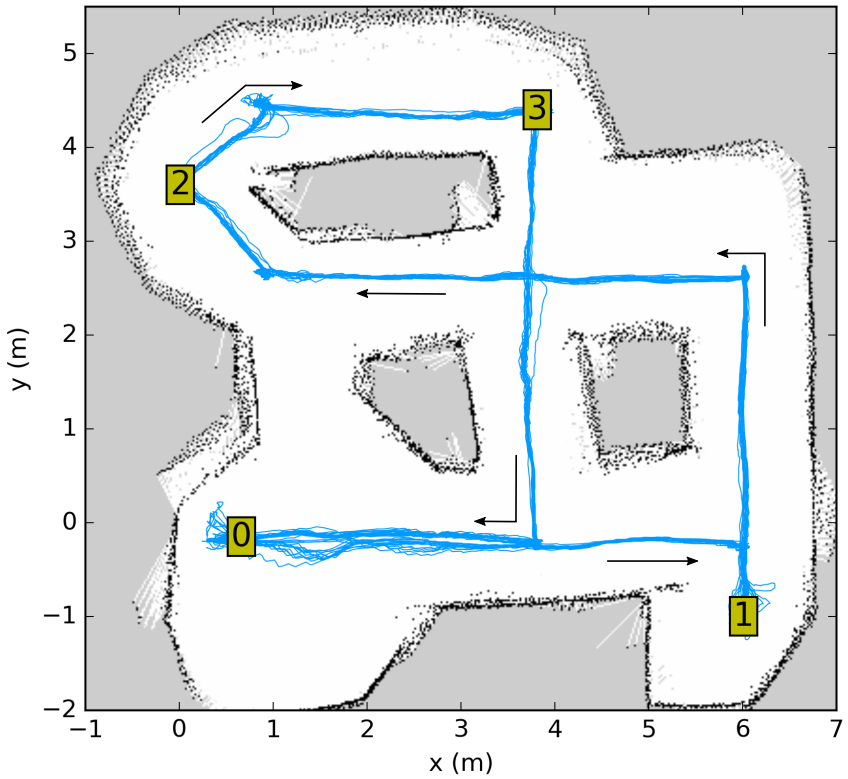


Figure 3.15: Executions in a real environment with the robot Pepper —the robot starts in waypoint 0, then it goes to 1, 2, 3, and back to 0. Arrows indicate the direction of motion.

3.3.5 Multi-resolution heuristic

Several tests were run to compare the performance of H2DMR, the multi-resolution heuristic, with the one with a fixed resolution, H2D. For the latter, the grid is built with the resolution given by the highest fidelity of the motion primitives, f^+. A map of an empty environment with dimensions 50×50 m was built for testing purposes, and then several octrees with the same occupancy information were obtained. For all of them, the minimum size of the cells is 0.1 m while the maximum one, C_+, ranges between 0.1 m and 25.6 m, depending on the test. Finally, both heuristics were run in each octree until they completely explored the free space —until the *OPEN* queue was empty.

Figure 3.16 compares the performance of H2DMR with that of H2D in terms of runtime and number of iterations. In the case of H2D, the number of iterations is constant, as the grid is built only taking into account f^+. Conversely, as H2DMR takes advantage of the octree

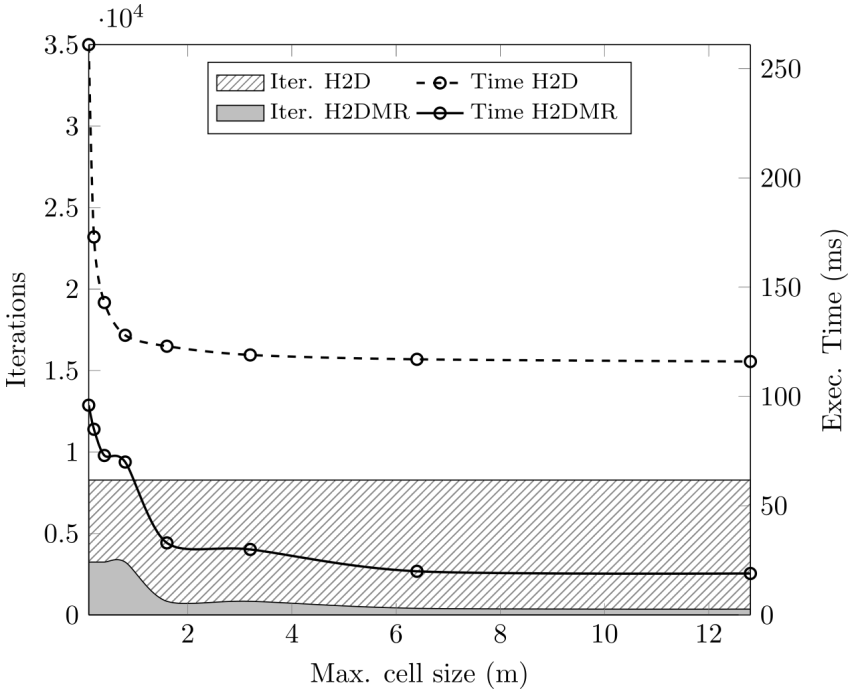


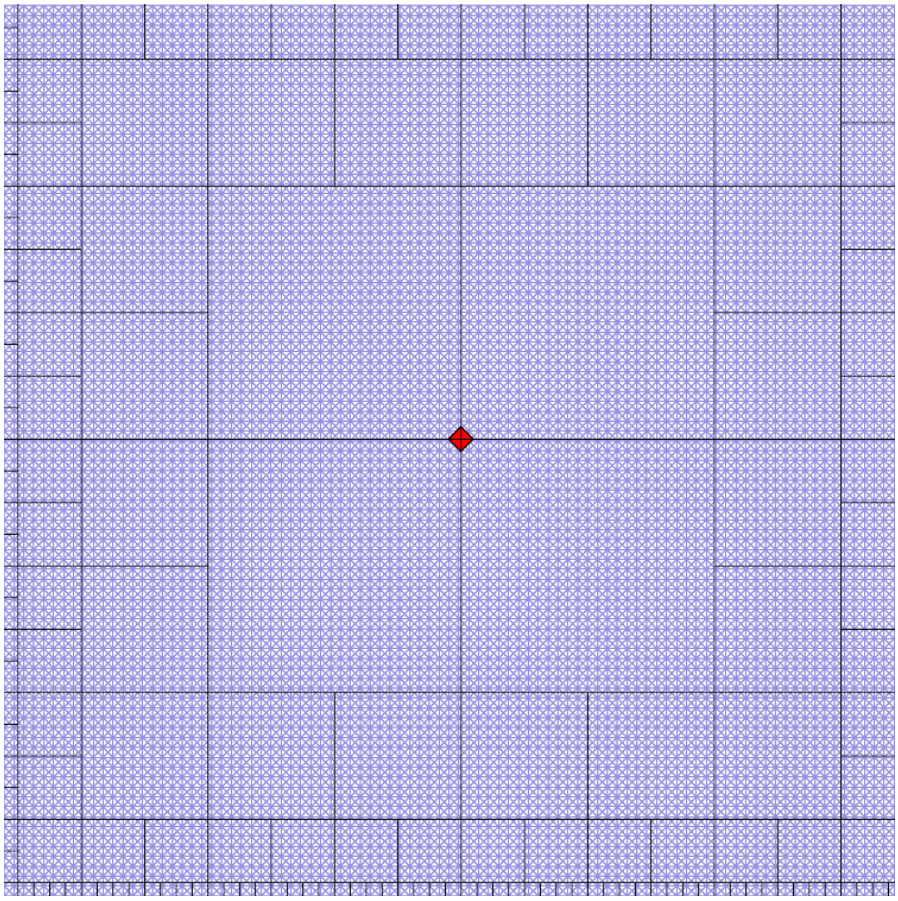
Figure 3.16: Performance of H2DMR compared with that of H2D. Results for the run time and the number of iterations needed to calculate heuristics over an empty 50×50 m map are shown.

structure to decide the best resolution for each area of the map, the number of iterations is significantly lower, and so it is the obtention time. Not only H2DMR is more efficient for all resolutions, but also the divergence from H2D increases with the cell size. Consequently, the benefits of this approach are specially noticeable in large environments, where the octree cells can be larger. The reduction in the runtime of H2D is only related with the collision check operations, which are more efficient with larger octree cells. Table 3.4 gives full detail of these tests, also quantifying the performance gain achieved by H2DMR with respect to H2D for each maximum octree resolution.

Table 3.4: Efficiency of H2DMR compared with that of H2D. Runtimes and number of iterations to obtain the heuristics over an empty map of 50 $\times$ 50 m are given. The maximum resolution of the octree is 0.1 m, while $f^+ = 0.5$ m. Tests were run for several maximum cell sizes, C_+ (in meters).

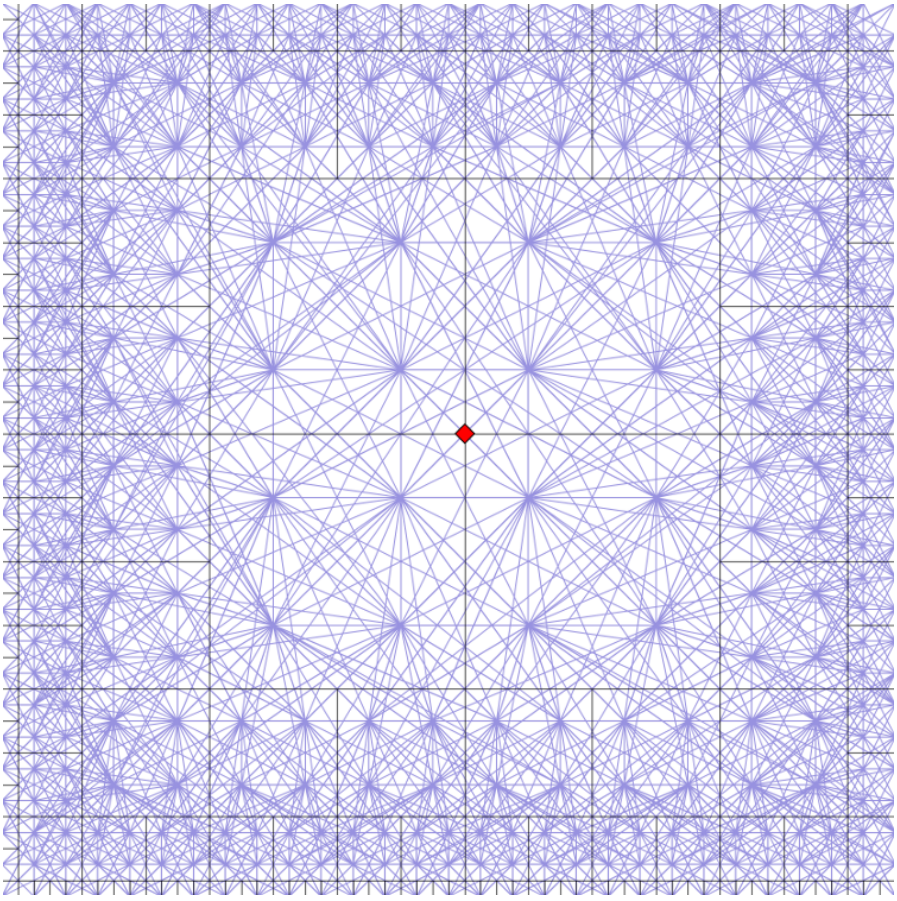
	Iterations			**Time (ms)**		
C_+	**H2D**	**H2DMR**	**Gain**	**H2D**	**H2DMR**	**Gain**
0.1	8,281	3,250	60.8%	261.2	96.0	63.2%
0.2	8,281	3,250	60.8%	173.7	85.3	50.9%
0.4	8,281	3,250	60.8%	143.8	73.0	49.2%
0.8	8,281	3,250	60.8%	128.6	70.0	45.6%
1.6	8,281	842	89.8%	123.4	33.0	73.3%
3.2	8,281	842	89.3%	119.0	30.0	74.8%
6.4	8,281	410	95.1%	117.8	20.0	83.0%
12.8	8,281	362	95.6%	116.2	19.0	83.7%

Finally, Figure 3.17 compares the grids built by H2DMR and H2D. The goal is located at the center of the map. In the case of H2DMR, shown in Figure 3.17(b), the resolution goes in accordance with the size of the octree cells and f^+ —0.5 m in this example. This results in a grid with a complexity considerably lower than that of H2D. However, this grid has an increased connectivity, as all points between adjacent cells have a link between them, instead of the 8-connected grid of H2D. This compensates the reduction in the number of paths due to diminishing the density of positions in the grid, specially in those areas with large octree cells. This allows retaining the quality of the estimations. In fact, the costs given by H2DMR were compared with those of H2D, showing an average reduction of a 4%. This means that H2DMR is also more precise than H2D.



(a) Single-resolution (H2D)

Figure 3.17: Comparison of the grid generated by H2D with that of H2DMR. The grid is in blue, the structure of the octree is in black, and the goal is the red diamond. The environment is 50 $\times$ 50 m.



(b) Muti-resolution (H2DMR)

3.4 Discussion and final remarks

In this chapter we have detailed a motion planning approach based on a state lattice with graduated fidelity that takes into account the kinematic restrictions, manages the motion and sensing uncertainty at planning time, and obtains the optimal solution in terms of probability of collision and traversal time. Our research has contributed to the motion planning field from different perspectives.

Firstly, we have presented a deterministic sampling based method that uses the predicted uncertainty to estimate the probability of collision of the paths in a realistic manner, taking into account the robot shape. This method relies on obtaining a minimal set of samples that properly represent the PDFs for the purpose of checking collisions in the environment. Each one of these samples is a different pose that the robot might have in execution time. Hence, the uncertainty in heading is also considered for estimating the probability of collision, which results in more reliable planning solutions. In this regard, the experimental results show that our method outperforms the Gamma function, another approach to obtain this estimation which disregards the uncertainty in heading, with only a 1.5% of average error. Moreover, the planning results in different environments show that the probability of collision was reliably estimated for robots of different shapes, under different uncertainty conditions and obstacle configurations. Thus, no collisions have been reported in the $1,000$ different executions of the simulated experiments, nor in the 20 executions of the real ones.

Secondly, we have presented an approach to obtain a graduated fidelity state lattice. This method takes into account the obstacles in the map and the maneuverability of the robot, as well as the predicted uncertainty. With this technique the areas of the map which are challenging for planning —e.g. due to their high probability of collision or the obstacle cluttering— are represented with high fidelity and contain a higher number of candidate paths. Also, with the proposed method other regions of the map which do not present these difficulties are represented with a lower number of candidate paths. More importantly, the fidelity is selected for each group of similar maneuvers separately. This way the diversity of the connectivity between the lattice states is retained because no motions are discarded, only their length is selected according to their availability and the environment conditions. Therefore, the quality of the solutions is preserved. With this graduated fidelity approach the planning runtimes have decreased on average a 88.5% with respect to a standard lattice planner due to the lower number of iterations and insertions of the search algorithm —the visited states—, which experienced a similar reduction.

Lastly, in this chapter we have introduced H2DMR, a multi-resolution heuristic that represents a low dimensional version of the motion planning problem. This heuristic estimates the cost to the goal due to the obstacles, building a grid which stores the estimated cost of traversing between each position and the goal. The resolution of the grid is selected according to the obstacles in the map, taking into account the maximum fidelity of the state lattice used for planning. The experimental results show that the multi-resolution grid is computed a 66.5% faster than the fixed-resolution counterpart, while its scalability in large environments is improved. The performance of H2DMR has been also studied, showing that the admissibility and consistency of this heuristic are retained and, in fact, the estimated costs are on average a 4% more precise than those of the fixed-resolution version.

The whole motion planning approach has been validated in 21 experiments —in different environments and under a variety of uncertainty conditions, robot shapes and kinematic restrictions— that have shown its reliability and efficiency. Moreover, it has been shown that our planning approach combined with the anytime search capabilities of AD* makes it possible to obtain sub-optimal solutions faster and refine them incrementally, retaining their safety regardless of their cost.

CHAPTER 4

AUTONOMOUS NAVIGATION FOR UAVS MANAGING MOTION AND SENSING UNCERTAINTY

In the previous chapter we have detailed the approach for motion planning under uncertainty proposed in this thesis, and we have validated it in a variety of mobile robots, focusing on 2D environments. The proposed motion planning strategy can be also used for 3D problems and systems with more complex dynamics, like UAVs. However, this requires addressing the increased dimensionality of the problem and find new strategies to obtain reasonable planning times. This chapter details the new algorithms that open the door to the obtention of plans for 3D environments in an efficient manner, as well as a whole approach for the autonomous navigation of UAVs.

More concretely, in Section 4.1 the system architecture is detailed, focusing on how the different components are combined, as well as their interdependencies. Section 4.2 details new algorithms for the autonomous navigation of UAVs, and also their integration in the motion planning approach that was described in Chapter 3 —contribution C4. In particular, we present a new method for estimating the probability of collision of the PDFs and a multi-resolution heuristic which work on 3D maps. In Section 4.3 we show the experimental results for a variety of environments, as well as the application of the proposed autonomous navigation system for planning routes for the reconstruction of 3D environments with a RGB-D camera —contribution C5. Finally, in Section 4.4 we conclude the chapter with some remarks.

4.1 System overview

The autonomous navigation system was designed focusing on the modularity and extensibility of the resulting architecture, which is detailed in Figure 4.1, so that each component can work independently and can be developed or replaced without affecting the rest. More concretely, the developed system is comprised by several independent software modules which work in parallel and communicate between them: the waypoint generator, the motion planner, the UAV state estimation and the flight controller. For the implementation we used the widely extended Robotic Operating System framework (ROS) [165], which facilitates the communications between the different software modules and the use of different hardware configurations. Moreover, the modular and flexible architecture allows adapting the different components without affecting the rest.

The state estimation module combines the UAV pose with the information of the inertial unit. The pose can be obtained from an external source, like a GPS, or be estimated by other means, such as a visual odometry algorithm which uses an on-board camera. Any localization method can be used with our system, provided that its error is bounded in time and it can be represented by a Gaussian distribution. Independently of the source, an Extended Kalman Filter combines the localization and the inertial unit measurements to improve the reliability of the estimated state. Moreover, this enhances the robustness of the system when dealing with inaccuracies and partial information.

The controller tracks the planned paths, minimizing the deviations that might occur during the flight. To achieve this, it adjusts the velocity commands in real time according to the estimated state of the UAV. The design of our system is not tied to a specific vehicle. For this reason, our approach assumes that a low level controller calculates the suitable throttle for each motor given the velocity commands.

The main advantage of the proposed architecture is that any of the software modules can be developed independently without affecting the others. This enables using different algorithm than the proposed ones —e.g. for the generation of waypoints or estimating the UAV state— without affecting the rest of the components.

4.1.1 Motion planner

The motion planner takes as input the set of waypoints to be visited by the UAV. These can be obtained in different manners depending on the nature of the mission: for autonomous naviga-

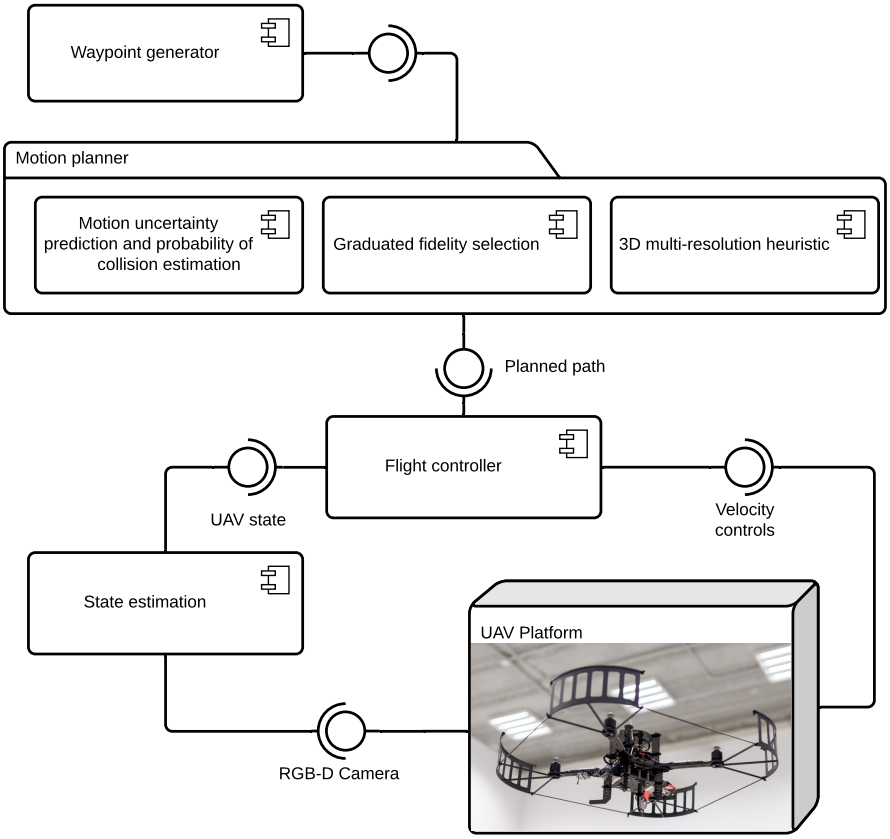


Figure 4.1: Architecture of the autonomous navigation system. The modularity of the software components makes it possible to execute the system with different configurations.

tion purposes only the current location and the goal are needed, while for scene reconstruction a set of waypoints allowing to collect data without leaving unseen areas is required.

As mentioned before, the autonomous navigation system for UAVs proposed in this chapter relies on the motion planning approach that was detailed in Chapter 3, which focused on 2D mobile robots. The applicability of these algorithms to 3D problems is not straightforward

due to the impact that the increased dimensionality has in the planning efficiency. Although this section focuses on the new algorithms that we propose for planning for UAVs, a brief summary of the motion planning strategy is offered below for the sake of clarity.

The motion planner obtains the path that minimizes the probability of collision and the flight time between consecutive waypoints. The outcome is the list of velocity commands that the robot has to execute to follow the plan, which are sent to the flight controller. The proposed planning approach: i) manages the motion and sensing uncertainty at planning time, predicting the probability distributions of the UAV being in each state of the path during its execution; ii) estimates the probability of collision for each path given the predicted uncertainty; iii) uses a state lattice and a graduated fidelity strategy which adapts to the obstacles in the map.

The lattice states are connected by a set of motion primitives, that are obtained from the dynamics model of the UAV, and that can be computed offline for the sake of efficiency. This way of representing the state space can be seen like a graph, so the optimal path between any pair of lattice states can be found with a search algorithm. This path, due to the construction of the state lattice, is formed by the set of feasible actions connecting them. As search algorithm we used AD* [125, 171], which allows obtaining sub-optimal solutions faster than the optimal one. The parameter ε of AD* acts as boundary for the cost of the sub-optimal solutions, which are refined iteratively with the information previously calculated. ε can be initially set to any sufficiently high value, which depends on the desired upper bound to the cost of the sub-optimal solutions, in order to compute the first path quickly.

Algorithm 6 outlines the main steps of the motion planner —a more detailed explanation can be found in [61]. AD* explores the lattice states in an iterative manner, for which a list containing the states yet to be explored is maintained —*OPEN*. The states in this list are ordered according to their current cost from the start and their estimated cost to the goal, which is given by the heuristics.

Our motion planner uses two different heuristics, combined by their maximum, to estimate the cost between each state and the goal. The first one, FSH [123], copes with the kinematic restrictions of the UAV considering free space, while the second one is a low-dimensional heuristic which copes with the obstacles in the map, H3DMR. Since this heuristic depends on the map, it has to be initialized every time the planner is run —Alg. 6:2—, so its efficiency is key for obtaining low runtimes. This is achieved using multi-resolution techniques which adapt the resolution of this heuristic to the obstacles in the map and reduce its obtention time

Algorithm 6 Main operations of the motion planner

Require: x^0 and x^G, initial and goal states, and ε_0

1: **function** MAIN (x^0, x^G, $\varepsilon = \varepsilon_0$)
2: INITIALIZEH3DMR(x^0, x^G) ▷ See Alg. 7
3: **while** $\varepsilon >= 1$ **do**
4: $OPEN = \{x^0\}$
5: **repeat**
6: $x^a = \arg\min_{x \in OPEN}(c_x + \varepsilon \cdot h_x)$
7: **for all** $x^b \in$ SUCCESSORS(x^a) **do**
8: $h_{x^b} =$ HEURISTIC(x^b) ▷ See Alg. 7
9: $c_{x^b} = c_{x^a} +$ COST(x^a, x^b) ▷ See Alg. 8
10: $OPEN = OPEN \cup \{x^b\}$
11: $OPEN = OPEN - \{x^a\}$
12: **until** $x^a = x^G$
13: publish $path(x^0, x^a)$ and decrease ε
14: **return**

to the lowest possible, as detailed in Section 4.2.1.

The inputs of the search algorithm are the initial state of the UAV, x^0, and the goal, x^G. AD* is initialized processing x^0 in the first place —Alg. 6:4. Then the algorithm processes other states iteratively until the goal is found. Firstly, the state x^a for which the sum of its cost and heuristic —c_x and h_x, respectively— is minimum, is pulled from *OPEN* —Alg. 6:6. As the heuristic solely informs about the traversal time, only this term is considered for combining c_x and h_x, and the rest of the information is taken unchanged from c_x. Secondly, in Alg. 6:7 the successors of x^a are obtained from the motion primitives, using the graduated fidelity strategy that was described in Section 3.2.2. For each successor, x^b, its heuristic and cost are calculated —Alg. 6:8-9—, and finally the state is inserted in *OPEN* to be explored in subsequent iterations. The optimal solution for the current value of ε is available after reaching the goal, x^G, after which the value of ε is decreased —Alg. 6:13. This process is repeated to refine the current plan until the optimal one is found —this is, $\varepsilon = 1$. Although there is no rule for decreasing the value of ε, better results are achieved if it is done at small steps [126].

4.2 Motion planning for UAVs

In this section we describe the new algorithms for addressing autonomous navigation of UAVs that are integrated in the motion planning approach proposed in Chapter 3. Section 4.2.1 introduces a novel multi-resolution heuristic which copes with the obstacles in the map, and Section 4.2.2 describes a method for estimating the probability of collision of the UAV in a reliable manner taking into account its real dimensions and the uncertainty in heading.

4.2.1 Multi-resolution 3D heuristic

H3DMR is a multi-resolution heuristic for motion planning in 3D environments. It estimates the cost to the goal taking into account the obstacles in the map, for which a 3D low-dimensional grid containing the estimated cost between each position in the map and the goal is built. To increase the efficiency of the heuristic and improve its scalability in large environments, the resolution of the grid adapts to the obstacles in the map. Since the precision of the heuristics affects the efficiency and the performance of the motion planner, the resolution of the H3DMR also takes into account the fidelity of the lattice used for planning. Moreover, as already mentioned in Section 4.1, this heuristic is combined with FSH, which improves the precision of the estimations due to coping with the kinematic restrictions of the UAV.

Algorithm 7 details how H3DMR is calculated. The inputs are the beginning position and the goal state of the UAV, q^0 and q^G; and the maximum fidelity of the state lattice, f^+. The grid is obtained applying the Dijkstra's algorithm, starting in q^G. Generating the grid backwards allows storing the estimated cost between every position and the goal, which is easily queried later and used as heuristic for planning. As stopping condition we used the double of the cost between the starting position, q^0, and the goal —Alg. 7:4. This is done for efficiency purposes, since the areas with higher costs are unlikely to be interesting for planning.

The algorithm stores all the positions to be explored in the *OPEN* queue, and it iteratively selects the position with the lowest cost from q^G —Alg. 7:5-6— to generate its neighbors, which are processed in subsequent iterations. The neighbors of a position q are obtained from the free space cells adjacent to κ, the one containing q —Alg. 7:7. This way they go in accordance with the resolution of the map. Moreover, with the goal of adapting the resolution of the grid to the maximum fidelity of the state lattice, f^+, each cell κ' adjacent to κ is

Algorithm 7 H3DMR

Require: f^+, highest fidelity of the state lattice
Require: q^0 and q^G, initial and goal positions of the UAV

function INITIALIZEH3DMR(q^0, q^G)
  $c_{q^G} = 0$; $c_{q^0} = \infty$
  $q = q^G$
  while $c_q > 2 \cdot c_{q^0}$ **do**
    $q = \arg\min_{q \in OPEN}(c_q)$
    $OPEN = OPEN - \{q\}$
    $\kappa = cell(q)$
    for all $\kappa' \in adjacent(\kappa)$ **do**
      if $size(\kappa') > f^+$ **then**
        INSERTSUBCELLS(q, κ')
      else
        $\kappa'' = adjust(\kappa', f^+)$
        $q' = position(\kappa'')$
        if $collisionBetween(q, q')$ **then**
          INSERTSUBCELLS(q, κ'')
        else
          $c(q') = c_q + \|q' - q\|$
          $OPEN = OPEN \cup \{q'\}$
function INSERTSUBCELLS(q, κ)
  for all $k'' \in subcells(\kappa')$ **do**
    if κ'' not adjacent to κ **then**
      continue
    $q' = position(\kappa'')$
    $c_{q'} = c_q + \|q' - q\|$
    $OPEN = OPEN \cup \{q'\}$
function H3DMR(x)
  $\kappa = cell(x)$
  $K = \{\kappa \cup adjacent(\kappa)\}$
  $Q = \{q$ inside $K \mid q$ position of the grid $\}$
  return $\arg\min_{q \in Q}(\|x - q\| + c_q)$
function HEURISTIC(x)
  return $\max(\text{H3DMR}(x), \text{FSH}(x))$

processed differently depending on its size. Figure 4.2 represents the two situations that are considered. On the one hand, those cells that are larger than f^+, like the one labeled as "A" in the image, are split into subcells of equal size —Alg. 7:9-10— to avoid obtaining neighbors at distances much greater than f^+, thus obtaining more precise estimations. For the sake of the efficiency, only those subcells of κ' that are adjacent to κ are considered.

On the other hand, the resolution of the cells is limited to f^+, since obtaining neighbors at lower distances would affect the efficiency of the heuristic. Cells smaller than f^+ are not considered for the purpose of generating the grid. Instead, H3DMR adjusts the resolution to that of the lattice maximum fidelity, like in cells "B" and "C" of Figure 4.2, so that the

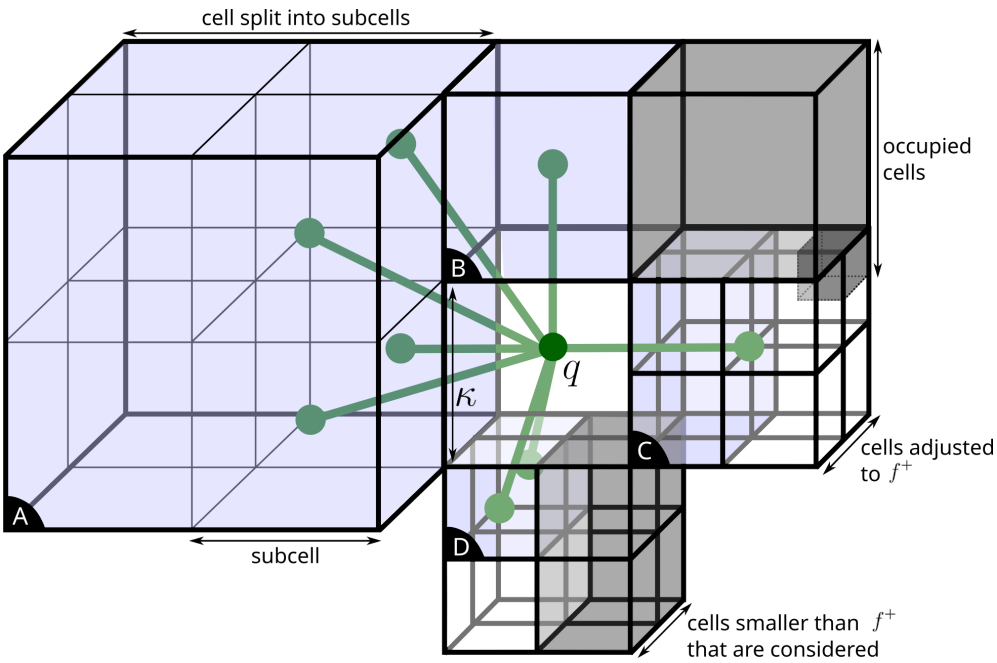


Figure 4.2: Neighbors of a point q of the multi-resolution grid, in green. κ is the cell which contains q, and the free adjacent cells used for obtaining the neighbor points are highlighted in blue. The large cell labeled as "A" is split into equal subcells, while "B" is used as it is, since its resolution is already adjusted to f^+. The cells on the right are smaller due to the obstacles —the occupied cells are colored in gray—, so they are adjusted to a lower resolution and represented by the larger cell containing them, labeled as "C". Some cells like "D" that are smaller than f^+ due to the obstacles are nevertheless considered for the sake of the precision of the heuristic.

neighbor point q' is obtained from cells that are at least as large as f^+ —Alg. 7:12-13. Although the general rule is to decrease the resolution whenever possible, an exception is made when there are obstacles between q and q'. In this case, faced the impossibility of using the adjusted cell for generating the neighbors of q due to the obstacles, its subcells are considered instead —Alg. 7:14-15. As some of these subcells might be free space —like the one labeled as "D" in Figure 4.2—, this way the H3DMR copes with the obstacles in a more precise manner and the optimistic nature of the heuristic is retained.

Finally, each neighbor position q' is inserted into the *OPEN* queue to be explored by the search algorithm in subsequent iterations, using the euclidean distance as the cost between q and q' —Alg. 7:17-18.

The multi-resolution grid obtained by H3DMR allows to estimate the cost between each point of the map and the goal, which is used by the motion planning algorithm as heuristic. A direct match between the states of the graduated fidelity lattice and the positions of the grid may exist, but this is not guaranteed. For this reason, we obtain the heuristic of a state x from the position of H3DMR for which the estimated cost to the goal is minimal, considering those which are inside the cell containing x and the adjacent ones —Alg. 7:26-30.

The strategy for obtaining H3DMR takes into account the fidelity of the lattice used for planning and also the obstacles in the map. The cluttered areas are represented in the map with cells of lower sizes, for which the resolution of the H3DMR grid increases. Conversely, the free space areas are represented with larger cells, resulting in lower resolutions for the H3DMR grid. This allows to estimate the cost to the goal very efficiently. Despite decreasing the resolution in some areas of the map, the estimations are precise due to taking into account the fidelity of the motion primitives.

4.2.2 Probability of collision estimation

The candidate paths are evaluated in terms of probability of collision, traversal time and the uncertainty at the final state, following the procedure detailed in Algorithm 8. These elements are compared hierarchically, so the motion planner minimizes, in the first place, the probability of collision. Among the safe paths, it gets the one with the minimal traversal time, and if several alternatives exist, it chooses the one with the lowest uncertainty at the goal. While the traversal time of the path is given by the motion primitives, estimating the probability of collision —with the purpose of obtaining the best plan in terms of safety— requires to predict the probability distributions of the UAV state during the execution of that path —Alg. 8:2.

Algorithm 8 Path evaluation

Require: x^a and x^b, beginning and final states

1: **function** COST(x^a, x^b)
2: $P^{a:b} =$ PREDICTUNCERTAINTY(x^a, x^b) ▷ See [6]
3: $c^{a:b} = 0$
4: **for all** $x_t \sim \mathcal{N}(\bar{x}_t, \Sigma_t) \in P^{a:b}$ **do**
5: $X^s =$ SAMPLE(x_t) ▷ See Alg. 9
6: $w_c = \sum_{\{x^s \in X^s \mid collides(x^s)\}}(w_{x^s})$ ▷ See Eq. 4.1
7: $c^{a:b} = c^{a:b} - \log(1 - w_c / \sum_{\{x^s \in X^s\}}(w_{x^s}))$
8: **return** $\left[c^{a:b}, time(x^a, x^b), tr(\Sigma^b)\right]$ ▷ $x^b \sim (\bar{x}^b, \Sigma^b)$

Estimating the probability of collision reliably is crucial for the robustness of the motion planner. To achieve this, from each predicted distribution $x_t \sim \mathcal{N}(\bar{x}_t, \Sigma_t)$ we obtain a set of samples —Alg. 8:4—, which we use to check collisions with the surrounding obstacles, as shown in Figure 4.3(a). The strategy of sampling the distributions allows considering the uncertainty in heading, in such a way that each sample represents a different pose of the UAV. In this way, the real dimensions of the UAV are taken into account centering the real shape in each sampled pose for the purpose of checking collisions with the obstacles in the map. Then, the probability of collision is estimated calculating the ratio between the weights of the colliding samples and the total —Alg. 8:6-7. The weight of each sample $x^s \in X^s$ is obtained according to its likelihood:

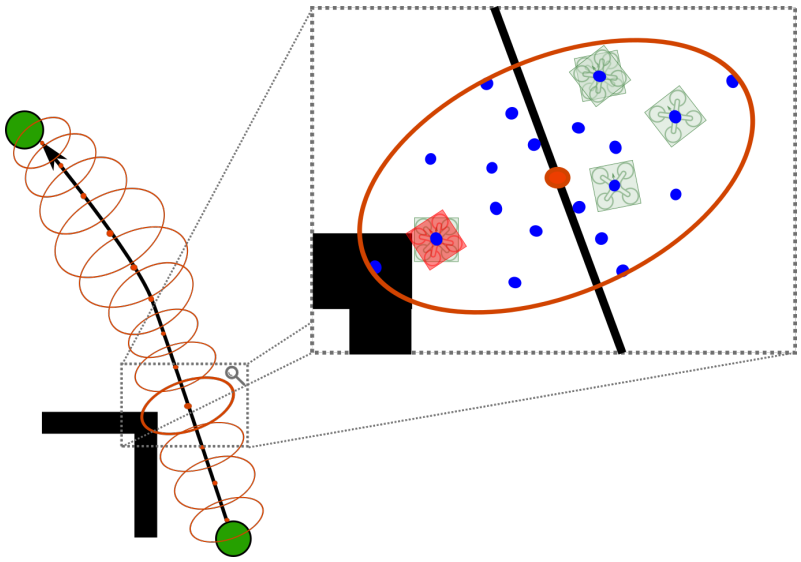
$$w_{x^s} = 2\pi^{-\frac{n}{2}} |\Sigma_t|^{-\frac{1}{2}} \exp\{-\frac{1}{2}(x^s - \bar{x}_t)' \Sigma_t^{-1} (x^s - \bar{x}_t)\} \tag{4.1}$$

and the cost for the entire path, $c^{a:b}$, is then obtained from the individual estimations —Alg. 8:7. Although by doing so we assume that they are independent, this is reasonable for practical purposes.

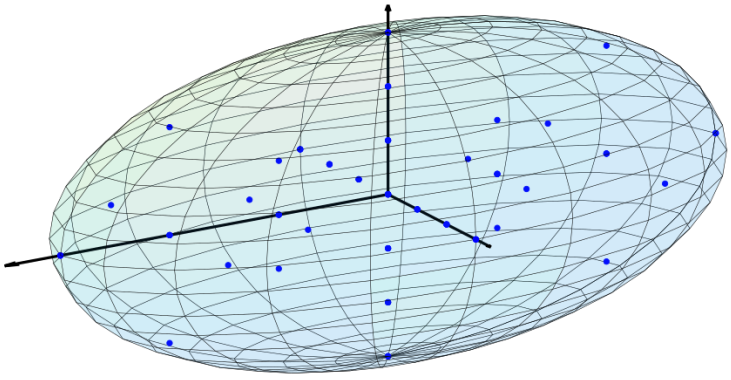
Despite the benefits of sampling the distributions, the high number of stochastic samples that would be needed to represent the distributions may severely impact the efficiency of the motion planner. For this reason, our approach uses a deterministic sampling strategy that represents the distributions with a limited number of samples, making it possible to use it in a real setup.

Our strategy relies on, for each one of the predicted distributions, obtaining a set of samples samples arranged in a regular manner within the limits of a confidence region. The

[6] PREDICTUNCERTAINTY is explained in detail in Section 3.1.3.



(a) Estimation of the probability of collision of a path taking into account the real shape and the uncertainty in heading —2D view.



(b) The samples —in blue— are obtained with a deterministic strategy which distributes them regularly according to the maximum direction of spread of the distributions —3D view.

Figure 4.3: Probability of collision estimation.

confidence region represents the volume of the space in which the UAV might be, with a certain probability, during the execution of the path. Arranging the set of samples following a regular pattern, as shown in Figure 4.3(b), allows to represent the distributions in a way which is suitable for collision check purposes, as well as to retain the deterministic nature of the planner.

Given a distribution for the state of the UAV, $x_t \sim \mathcal{N}(\bar{x_t}, \Sigma_t)$, the eigenvectors and eigenvalues of Σ_t represent the directions and magnitude of maximum spread of x_t in each dimension, which allow distributing the samples accordingly.

The volume of the confidence region depends on the probability that a random sample of x_t would fall in it. In our approach, the motion uncertainty is represented with multivariate Gaussian distributions, for which the confidence region consists in those values of x satisfying:

$$(x - \bar{x_t})' \Sigma_t^{-1} (x - \bar{x_t}) \leq \chi_n^2(p) \tag{4.2}$$

where $\chi_n^2(p)$ is the quantile function for a Chi-Square distribution with n degrees of freedom, and p is the probability that the confidence region encompasses the state of the UAV. In the 3D space, this confidence region can be represented by an ellipsoid, whose implicit equation —without considering any rotation given by the eigenvectors— is:

$$\sum_i^n \frac{x_i^2}{\lambda_i^2} = \chi_n^2(p) \tag{4.3}$$

where λ_i is the i-th eigenvalue of Σ_t. From Equation 4.3 follows that the length of each semi-axis of this ellipsoid is:

$$a_i = \sqrt{\chi_n^2(p) \cdot \lambda_i} \tag{4.4}$$

Our goal is to obtain a minimal set of samples which represent the distribution well enough for estimating the probability of collision reliably. We use the a_i values from Equation 4.4 to obtain samples at l different levels, all equally spaced. While most of them are in the directions of maximum spread due to their likelihood, we also sample in the diagonals to avoid leaving gaps which might be relevant for collision check purposes. This is achieved by combining the direction of maximum spread in different dimensions, given by the corresponding eigenvectors. However, in order to limit the number of samples and avoid obtaining poses with very low likelihoods, we only combine two dimensions at the same time.

Algorithm 9 details the sampling process for a distribution $x_t \sim \mathcal{N}(\bar{x_t}, \Sigma_t)$. First, in Algorithm 9:2, we factorize the covariance matrix into a canonical form, obtaining its representation in terms of eigenvectors and eigenvalues —U_t and V_t, respectively. U_t is a $n \times n$ matrix

whose i-th column —$U_{t[i]}$— is the i-th eigenvector, and V_t is a diagonal matrix containing the eigenvalues —$\lambda_i, ..., \lambda_n$. Then, in Algorithm 9:3 we obtain the quantile of the chi-square distribution which satisfies Equation 4.2, and we initialize the set of samples, X^S, with the mean of the distribution —Alg. 9:4.

X^S is populated iterating over the different dimensions of x_t. We obtain samples aligned with the directions of maximum spread, given by the corresponding eigenvector, $U_{t[i]}$ —Alg. 9:7-10. These samples are combined with the direction of spread of the j-th dimension to obtain samples also in the diagonals —Alg. 9:13-16. The parameter l allows obtaining L samples with different distances to the mean —Alg. 9:8 and Alg. 9:14—, which are equally spaced in their direction axis and with the maximum distance which is given by the corresponding eigenvalue and $\chi^2_n(p)$, as shown in Equation 4.4 —Alg. 9:7 and Alg. 9:13.

Algorithm 9 Regular sampling of the distributions

Require: $x_t \sim \mathcal{N}(\bar{x}_t, \Sigma_t)$
Require: p, probability of the confidence region
1: **function** SAMPLE(x_t)
2: $U_t V_t U_t^{-1} = \Sigma_t$ ▷ Eigendecomposition of Σ_t
3: $\chi_p = \chi^2_n(p)$ ▷ Quantile of the chi-square
4: $X_s = \{\bar{x}_t\}$
5: **for** l in $1, ..., L$ **do**
6: **for** i in $1, ..., n$ **do**
7: $a_i = \sqrt{\chi_p \cdot \lambda_i}$
8: $d_i = l/L \cdot a_i$
9: $x_s^+ = \bar{x}_t + d_i \cdot U_{t[i]}$
10: $x_s^- = \bar{x}_t - d_i \cdot U_{t[i]}$
11: $X_s = X_s \cup \{x_s^+, x_s^-\}$
12: **for** j in $1, ..., n; j \neq i$ **do**
13: $a_j = \sqrt{\chi_p \cdot \lambda_j}$
14: $d_j = l/L \cdot a_j$
15: $X_s = X_s \cup \{x_s^+ + d_j \cdot U_{t[j]}, x_s^+ - d_j \cdot U_{t[j]}\}$
16: $X_s = X_s \cup \{x_s^- + d_j \cdot U_{t[j]}, x_s^- - d_j \cdot U_{t[j]}\}$
17: **return** X^s

4.3 Results

In this section we report planning results for 2D and 3D autonomous navigation, and also for the generation of trajectories for autonomous scene reconstruction. The UAV is an AscTec

Pelican[7] equipped with a RGB-D camera, the Orbbec Astra Pro[8], which has VGA resolution at 30 FPS for both the depth and RGB information. We used the publicly available implementations of the ORB-SLAM2 [140] algorithms to localize the UAV in the environment, and KinectFusion [145] to obtain a 3D model from the data gathered during the flight. For collision check purposes we have approximated the UAV as a $0.5 \times 0.5 \times 0.4$ m cuboid, which includes the space occupied by the RGB-D sensor and the propellers. During the experiments we used several localization techniques with different uncertainties. In Section 4.3.1 we relied on a motion capture system, while in Sections 4.3.2 and 4.3.3 we used the RGB-D camera and the publicly available implementation of ORB-SLAM2 [140], a visual-based SLAM system which uses loop closure to limit the long term drift and to keep the localization errors bounded. Moreover, in Section 4.3.3 KinectFusion [145] was used to obtain a dense 3D model from the data gathered with the RGB-D camera during the flight.

The UAV controller and the state estimation algorithm are run in the on-board computer, which has a CPU Intel Atom™ x5-Z8300 at 1.84 GHz, while the motion planner, the visual odometry and the 3D reconstruction algorithms are run in a laptop with CPU Intel Core™ i7-4720 at 2.6 GHz.

The state vector of the UAV contains the position of its center of rotation, the heading and the current linear and angular speeds in each axis:

$$\left[x, y, z, \varphi, \psi, \theta, v_x, v_y, v_z, v_\varphi, v_\psi, v_\theta\right] \tag{4.5}$$

The control vector, on the other hand, contains the variables that can be managed for a multirotor system. These are the desired linear speeds in each dimension, and the angular speed in the vertical axis:

$$\left[u_x, u_y, u_z, u_\omega\right] \tag{4.6}$$

The UAV motions are represented with acceleration based equations. The dynamics model was obtained from 24 min and 18 s of real flight data, recorded prior to the experiments described in this section, as described in [1]. Thus, the real behavior of the UAV to the different velocity commands was learned. Finally, the set of motion primitives used for planning purposes was obtained from the dynamics model, following the method described in [70]. In order to represent a variety of maneuverability restrictions, different motion primitives were used for each experiment despite the fact that the UAV dynamics model was the same.

[7] http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-pelican/
[8] https://orbbec3d.com/product-astra-pro

4.3.1 2D autonomous navigation

We tested the reliability and the precision of the autonomous navigation approach in a cluttered environment of 5×4 m with several narrow passages. For this experiment the set of motion primitives was comprised of $3,316$ actions which connected states between 0.2 m and 0.8 m. With these motion primitives the UAV was allowed to move forward and laterally at a maximum of 0.7 m/s, and rotate at a maximum of 30 deg/s. Motion and sensing uncertainty were $M_t = 0.08 \cdot I$ and $N_t = 0.006 \cdot I$, respectively. The UAV state was estimated combining the information of the inertial unit with the position given by an external motion capture system. The use of the RGB-D camera for this experiment was discarded due to the limitations of the depth sensor —its optimal range is from 0.6 m to 5 m. Moreover, in this experiment the UAV navigates at a minimum distance to the obstacles of only 0.11 m, which requires a very precise localization.

There are several waypoints in the map that the UAV has to visit, ordered by number, starting and ending the route in the number 0. The maximum altitude was limited to 1.0 m, so that the planned paths go through the obstacles instead of surpassing them from above. The controller makes a request to the motion planner to calculate the path to the next waypoint when the UAV is approaching the current goal. Due to the efficiency of the motion planning approach, these paths can be obtained on-demand during the flight without causing delays.

Figure 4.4 shows the route followed by the UAV in 20 different executions of the experiment. As it can be seen, the predicted distributions adjust well to the real UAV state. The average deviation of the UAV during the 20 executions was 6.4 cm.

This scenario is challenging because of the proximity between the UAV and the obstacles. The minimum distance between them is as low as 11 cm in some points of the route, so reliable plans —in terms of estimated probability of collision— and precision in the control are required to avoid collisions. Our motion planner achieves this by reducing the speed of the UAV in those maneuvers which could affect the safety of the flight, e.g. just before passing between the obstacles in the middle of the map. This way the UAV stabilizes itself and can approach the obstacles in a safe manner, minimizing the overall probability of collision. In fact, in the 20 executions of this experiment no accidental collisions were reported. However, this would not be possible if both the kinematic restrictions and the uncertainty were not considered at planning time. In fact, in this kind of narrow passages those approaches that obtain a coarse high level solution would have to evaluate *in situ* the feasibility of the operation and either rely on their low level planner or being conservative and find an alternative path, if it

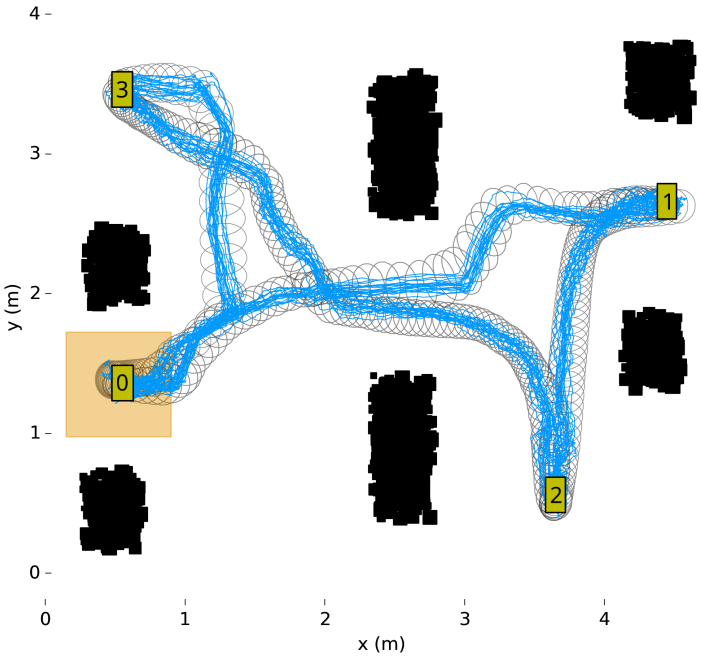


Figure 4.4: 20 different executions of our autonomous navigation approach in a cluttered environment. The followed routes are in blue, obstacles are in black, and waypoints are in yellow. The UAV takes off in 0, then visits 1, 2, 3, goes back to 0 and lands. Ellipses show $3 \cdot \Sigma$ of the predicted distributions.

exists. For both approaches the safety and cost of the solution cannot be quantified in advance, which might limit their suitability in some cases. In a similar manner, those approaches that consider the kinematic restrictions but not the uncertainty at planning time cannot offer any optimality guarantees regarding the safety of the plans. Instead, our approach is able to obtain reliable solutions that are optimal both in terms of safety and traversal time before the UAV starts moving.

Table 4.1 details the planning results for this experiment. All the solutions are obtained within seconds, so the UAV can navigate between points without noticeable delays. The total cost of the route —the time the UAV is navigating between waypoints— is 119.6 s, although the execution time is slightly higher because the UAV stabilizes itself for a couple of seconds at each waypoint[9].

[9] A video recording of this experiment is available at `https://youtu.be/HTCd3cwix60`.

Table 4.1: Planning results for the navigation experiment of Figure 4.4. Column "Iterations" is the number of iterations of AD*, "Insertions" is the number of nodes inserted in *OPEN*, and "Time" is the planning time. "Cost" is the traversal time of the planned path, and "Min dist." is the minimum distance to the obstacles, in meters. All times in seconds.

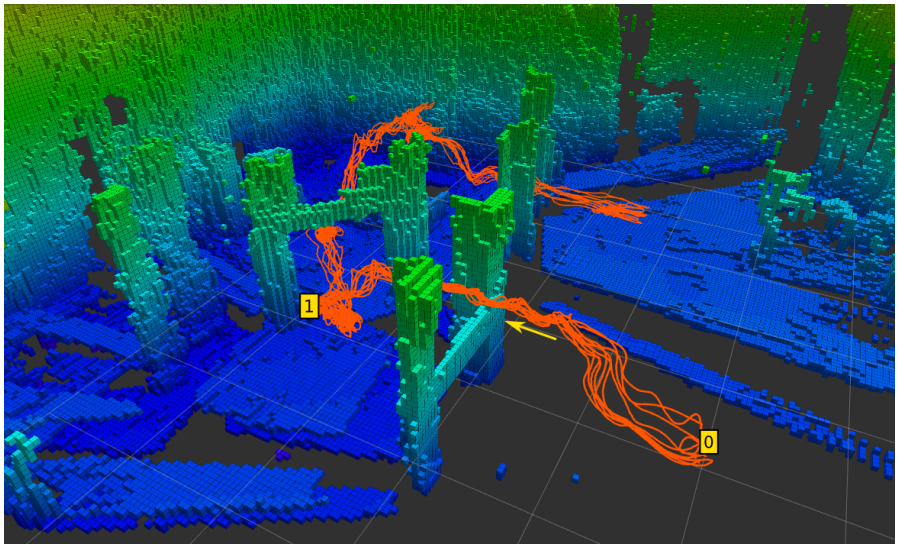
	Planning			Solution	
Path	**Iterations**	**Insertions**	**Time**	**Cost**	**Min dist.**
0-1	194	1,584	2.4	18.5	0.11
1-2	160	1,549	2.9	17.2	0.16
2-3	428	3,540	7.3	29.0	0.15
3-0	385	2,906	6.1	16.9	0.18
TOTAL	1,167	9,579	18.7	119.6	—

4.3.2 3D autonomous navigation

We have also tested our autonomous navigation approach in an environment of $8 \times 4 \times 2$ m with obstacles at different altitudes, which compelled the UAV not only to navigate between them, but also to change its altitude while moving. For this experiment the UAV state was estimated from the on-board RGB-D camera, using ORB-SLAM2 [140] with loop closure, combined with the inertial unit using an Extended Kalman Filter. Due to the minimum range of the depth sensor, the UAV cannot approach the obstacles closer than 0.6 m on its front side. For this reason, for motion planning purposes we have defined the UAV shape taking into account the additional distance which is needed for data acquisition. In this experiment the set of primitives used for planning included $1,700$ motions which allowed the UAV to move forward, rotate and change its altitude. These motions connected states of distances 0.25 m, 0.5 m and 1 m, so the maximum fidelity is 0.25 m. Motion uncertainty was that of Section 4.3.1, while for this experiment sensing uncertainty was $N_t = 0.05 \cdot I$.

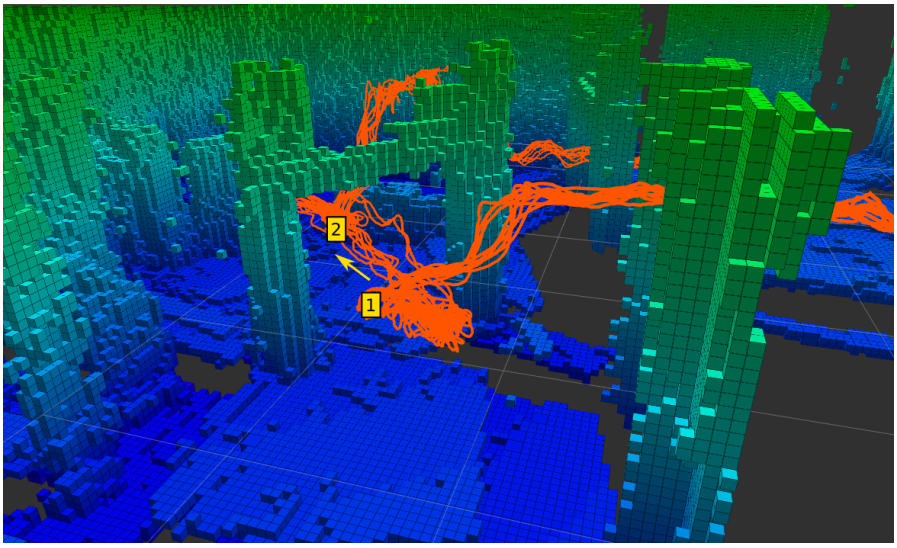
The map of the environment was obtained beforehand from a handheld RGB-D camera, using ORB-SLAM2 and OctoMap [69] to store the result. In execution time, the map was updated at 1 Hz with the latest measurements of the RGB-D camera. Figure 4.5(a) shows a 2D view of the final map and the distribution of the waypoints. Moreover, it details the results of 20 different executions of the UAV following the planned paths, and how well the predicted distributions adjust to the true UAV state. The average deviation of the UAV during these executions was 12.3 cm. There are several structures formed by vertical columns that the UAV has to cross. These columns are bonded by horizontal strips placed at different altitudes, and the planned paths have to surpass them from above or from below depending on their position and the predicted motion uncertainty.

(a) 2D view of the executions of this experiment. The obstacles are in black, while the waypoints are numbered in yellow. Ellipses show $3 \cdot \Sigma$ of the predicted distributions.

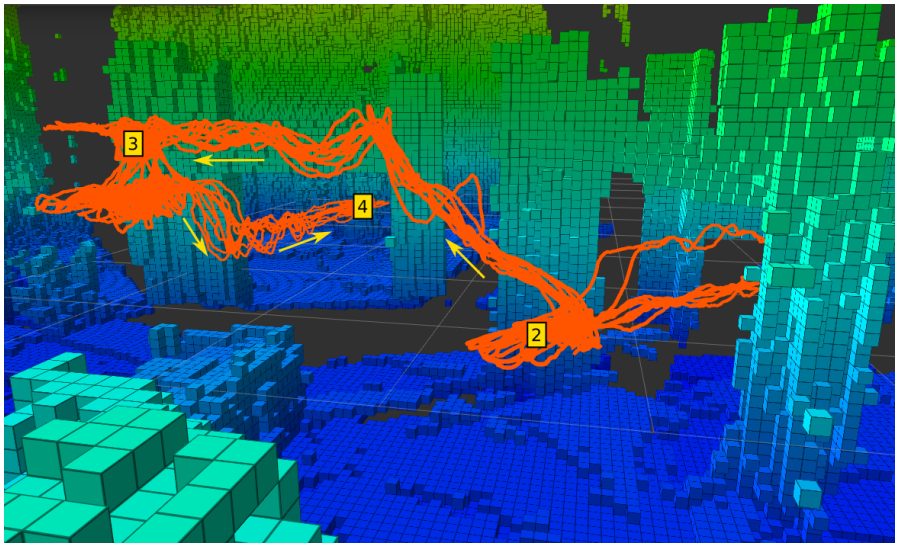


(b) 3D view of the executions, in orange, focusing on the path between waypoints 0-1.

Figure 4.5: Executions of the planned paths in a 3D environment, avoiding obstacles at different altitudes. The different routes followed by the UAV are in orange, while the waypoints and directions of motion are in yellow. The cells of the map are colored between blue and green, depending on their altitude. Detailed planning results are shown in Table 4.2.



(c) Detail of the executions of the planned route between waypoints 1-2.



(d) Executions of the planned path between waypoints 2-3-4.

Figure 4.5(b) shows the path to the first waypoint, which avoids the horizontal strip from above to take advantage of the vertical speed acquired during the takeoff. After that, the UAV descends again to reach the first waypoint, which is located between the two first obstacles. The second path, shown in Figure 4.5(c), goes through the second pair of columns, but before crossing them the UAV has to change its heading, because only motions in the direction of the camera view are allowed. These kind of turnings are challenging because of the motion and sensing uncertainty. In fact, the UAV does not remain in a stable position while executing these maneuvers because of the inaccuracies of the estimated state, so the controller has to correct the possible deviations to avoid collisions. This is taken into account by the motion planner, which copes with this situation leaving enough time to stabilize the UAV before approaching the obstacles.

As shown in Figure 4.5(d), after going through the second pair of columns, the UAV goes to the third waypoint, which is on the top of an obstacle with irregular shape. Finally, it goes to the goal, which is beyond the final structure. To achieve it, the UAV changes its heading again and then descends enough to surpass the horizontal strip from below.

Table 4.2 contains the detailed planning results for this experiment. The map used for planning purposes was obtained *a priori* and updated in real time at 1 Hz while the UAV was moving. There were several waypoints that the UAV had to visit consecutively, and each path was obtained before reaching its starting point. The reported planning times allowed the UAV to execute all the routes without delays[10].

Table 4.2: Planning results for the 3D autonomous navigation experiment of Figure 4.5. Column "Iterations" is the number of iterations of AD*, "Insertions" is the number of nodes inserted in *OPEN*, and "Time" is the planning time. "Cost" is the traversal time of the resulting path. All times in seconds.

	Planning			Solution
Path	**Iterations**	**Insertions**	**Time**	**Cost**
0-1	24	297	2.78	15.08
1-2	23	569	4.63	15.13
2-3	55	841	6.70	15.97
3-4	86	1,569	9.4	20.98
TOTAL	188	3,276	23.51	67.16

[10] A video recording of this experiment is available at `https://youtu.be/xephfR35PYo`.

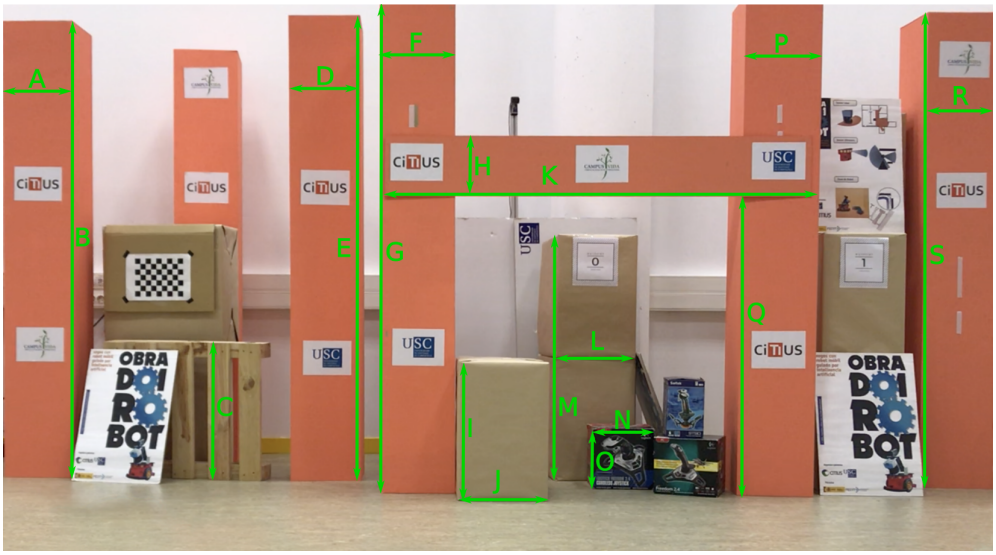
4.3.3 Autonomous navigation for scene reconstruction

In this section we report results for the generation of trajectories to achieve the autonomous reconstruction of a real scene. The goal is to obtain an accurate 3D model without manually controlling the UAV. Like in the experiment of Section 4.3.2, the UAV state was estimated with the on-board RGB-D camera, using ORB-SLAM2 with loop closure, and the inertial unit. For the scene reconstruction we used the KinectFusion [145] algorithm. The uncertainty conditions were the same as in the experiment of Section 4.3.2.

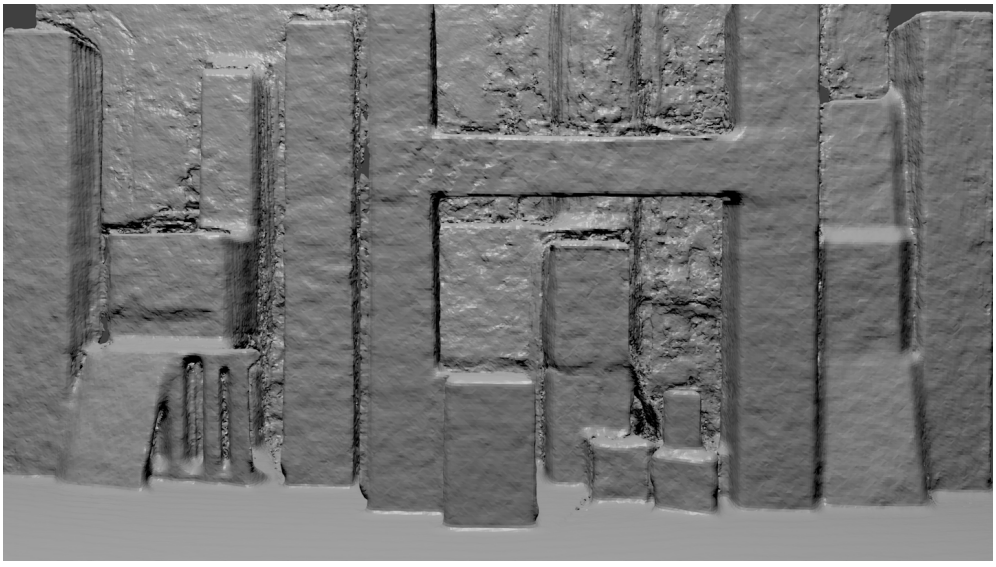
The autonomous navigation module takes as input the dimensions of the scene to be reconstructed and obtains a set of waypoints for the data acquisition. These waypoints guide the flight of the UAV in horizontal layers from one side of the scene to another, increasing the altitude at regular intervals. Like in the other experiments, the paths between waypoints are planned on demand, so the next one is obtained just before arriving to the current goal. However, in this case the navigation is done without prior knowledge of the map. The occupancy data, stored in the multi-resolution map, is updated in real time and used not only for planning purposes, but also to check collisions in the current path. If this is the case, the motion planner would obtain a new collision free plan taking into account the new obstacles.

The UAV has to move in a smooth and consistent manner to successfully obtain the reconstruction. This is because KinectFusion [145], like many other reconstruction algorithms, relies on Iteratively Closest Point (ICP) techniques —matching the current depth scan with the previous ones to incrementally build a consistent model. Our autonomous navigation approach is key to guarantee a successful reconstruction as, unlike other strategies, it manages the uncertainty, the kinematic restrictions and the controller all together, generating paths that ensure smooth camera motions.

This experiment consisted in the autonomous reconstruction of a scene of 2.5×0.75 m with objects of different sizes, which is shown in Figure 4.6(a). The reconstruction algorithm was able to obtain a dense model with a high level of detail, shown in Figure 4.6(b), with no holes in the surfaces that were directly seen by the RGB-D camera. The accuracy of the reconstruction is shown in Table 4.3, which compares some distance measurements of the objects in the real scene —labeled in Figure 4.6(a)— to their counterparts in the obtained model. The error was, on average, of 1.08% with a standard deviation of 0.71%. The overall accuracy of the reconstruction is very high, with errors as low as 1 mm. The highest error was measured in the height of the orange columns —which were overestimated between 2.5 and 4 cm, which is around a 2% of their real size. This was caused by limiting the altitude



(a) Real environment for the autonomous reconstruction experiment. The green labels are the distance measurements to show the accuracy of the obtained model, as detailed in Table 4.3.



(b) Render of the point cloud resulting from the dense reconstruction algorithm applied to the real environment in a).

Figure 4.6: Autonomous dense reconstruction of a scene with a UAV and a RGB-D camera.

Table 4.3: Distance measurements taken in the real scene compared to their counterparts in the reconstructed model, showing the accuracy of the 3D scene reconstruction. The highlighted measurements are those with the minimum and maximum error. All distances in centimeters.

Distance	Reconstruction	Real	Error
A	30.9	31.0	0.32%
B	203	200.5	1.25%
C	59.7	60.0	0.50%
D	31.1	31.0	0.32%
E	204.5	200.5	2.00%
F	29.6	30.3	2.31%
G	203.7	200.5	1.60%
H	23.7	24.0	1.25%
I	54.3	53.2	2.07%
J	34.1	34.6	1.45%
K	177.3	177.5	0.11%
L	33.6	34.3	2.04%
M	107.8	106.5	1.22%
N	24.9	25.0	0.40%
O	24.8	25.0	0.80%
P	30.2	30.3	0.33%
Q	130.4	129.8	0.46%
R	30.8	31.0	0.65%
S	203.5	200.5	1.50%

of the flight, which prevented the RGB-D camera from taking measurements from above the columns.

The set of motion primitives of this experiment was comprised of 492 actions which allowed the UAV to change its altitude, move forward and laterally. The speed was limited to $0.3\,\mathrm{m/s}$, which was suitable for the purpose of scene reconstruction. Several routes were planned to acquire the data for the reconstruction. Concretely, the UAV went from side to side three times and changed its altitude twice. Planning results are detailed in Table 4.4. The total cost of the obtained routes was 34.09 s, although the duration of the flight is longer because of stabilizing the UAV at the end of each path[11].

Something to take into consideration is that the heuristic H3DMR has to be re-calculated every time the map changes, although operating in real time is not an issue due to the reduced planning times. In fact, due to its efficiency, H3DMR is obtained on average in 94 iterations and less than 40 ms.

[11] A video recording of this experiment is available at `https://youtu.be/lUAwEXGYBOA`.

Table 4.4: Planning results for the autonomous reconstruction environment of Figure 4.6. Column "Iterations" is the number of iterations of AD*, "Insertions" is the number of nodes inserted in *OPEN*, and "Time" is the planning time. "Cost" is the traversal time of the resulting path. All times in seconds.

	Planning			Solution
Path	**Iterations**	**Insertions**	**Time**	**Cost**
1	6	120	0.73	9.89
2	3	113	0.52	2.21
3	6	125	0.52	9.89
4	3	116	0.62	2.21
5	6	125	0.49	9.89
TOTAL	24	599	2.88	34.09

As mentioned before, a smooth flight is key for getting a good 3D model. The consistency of the reconstruction benefits from a stable flight, so we measured this in the experiment. We have recorded the state of the UAV, discarding those measurements which did not correspond with the planned paths —the takeoff, the landing and the moments in which the UAV is holding its position in the air. The distribution of the real speed during the execution of the planned routes is shown in Figure 4.7. The average speed is $0.289\,\mathrm{m/s}$, almost in the upper bound of $0.3\,\mathrm{m/s}$. More importantly, the standard deviation is only of 0.053, resulting in a navigation which is smooth enough for obtaining a consistent model.

A drawback of using visual odometry to estimate the state of the UAV is the increased delay with respect to other localization systems. Since the LQG controller uses these estimations to adjust the controls and minimize deviations from the expected state, there might be convergence issues in certain situations. However, our motion planner takes into account the dynamics model of the UAV, obtained from navigation data which is acquired with this localization system. Thus, the delay in the estimations is implicitly taken into account, overcoming this issue and guaranteeing the stability of the controller.

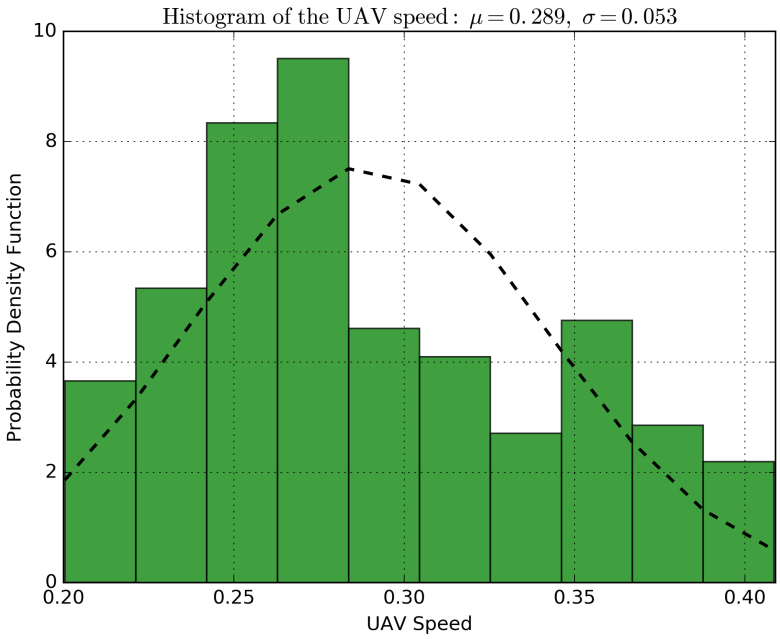


Figure 4.7: Speed of the UAV during the execution of the autonomous navigation and reconstruction experiment, which was limited to 0.3 m/s. The mean was $\mu = 0.289$, and the standard deviation was $\sigma = 0.053$, which resulted in a stable flight.

4.4 Final remarks

In this chapter we have detailed the new algorithms that are necessary to apply the motion planning approach —proposed in Chapter 3— to UAVs and cope with the increased dimensionality of the problem, as well as their validation on a real platform. The new algorithms have been integrated in a flexible architecture for the autonomous navigation of multi-rotor UAVs which includes the motion planner, a feedback controller, and a method which estimates the state of the UAV during the flight.

The new method for estimating the probability of collision avoids obtaining extremely low likelihood samples that could affect the planning efficiency, despite being poorly representative for the outcome. In contrast, the redesigned spatial distribution of the samples retains a good coverage for collision check purposes while reducing their number. In addition to this, in this chapter a new multi-resolution heuristic which works in 3D environments, H3DMR,

has been introduced. In this case, the connectivity of the grid has been improved to reduce the number of connections between adjacent cells while maintaining their diversity. Thus, the obtention time and the precision of the estimated costs are properly balanced.

We have validated the proposed autonomous navigation system using a real UAV, model AscTec Pelican, to fly autonomously in a cluttered environment where high precision maneuvers are required. In the same way, we have demonstrated the capability of flying autonomously in a cluttered, 3D environment using a visual SLAM algorithm to localize the UAV. These tests show the reliability of the motion planning approach, as in 20 different executions of the planned routes no collisions were reported.

The autonomous navigation approach has been also tested in a practical application, the generation of paths for the reconstruction of 3D scenes with a RGB-D camera in an autonomous manner. In this regard, the KinectFusion algorithm was later used for the obtention of a dense model, for which the measured error was on average of 1.08%.

In all these cases, the experimental results have shown that the efficiency of the motion planning algorithms allow operating the UAV without delays. But more importantly, the probability of collision of the plans was realistically estimated and no collisions were reported during the experiments.

CHAPTER 5

CONCLUSIONS

The number of practical applications that might benefit from some sort of automatization increases as the capabilities of the different robotic platforms improve and become more complete. Some examples of this include traffic monitoring, post-disaster assessment and management, rapid delivery of basic life support in cases of cardiorespiratory arrest, infrastructure inspection or the reconstruction of 3D scenarios. In this context, navigating autonomously in environments of different natures has become increasingly relevant due to the fact that it is inevitably part of the automation of these and many other tasks.

This problem has been addressed in the state of the art using different techniques, whose strengths and drawbacks have been analyzed in this thesis in Chapter 2. In this revision of prior research the necessity of improving the efficiency of the existing methods for planning under uncertainty has been identified. Predicting the uncertainty and, more specifically, estimating the probability of collision of the candidate paths, is a computationally expensive operation that causes a significant overhead in the motion planning algorithms. Despite the challenge that planning under uncertainty poses, its relevance in those cases for which safety or accuracy are of critical importance makes it essential to propose solutions that address this problem reliably.

This thesis has focused on developing a motion planning approach which manages the trade-off between planning efficiency and performance. More concretely, this approach is designed for partially observable systems with nonlinear dynamics for which the random motion and sensing disturbances can be described with Gaussian distributions. From a practical point of view, our strategy can be used for the autonomous navigation of real robots of differ-

ent kinds, among which wheeled platforms and UAVs stand out for their applicability to real world problems. The proposed motion planning approach relies on some strategies that have been combined in a novel manner with the goal of obtaining a reliable, efficient and scalable approach:

- The motion planning strategy relies on state lattices, which allow to deal with the kinematic restrictions in an efficient manner. The discrete states are connected by a set of position independent, canonical actions which can be replicated throughout the entire lattice. These actions have been obtained from the vehicle motion model offline using a Newton-Rapshon based numerical optimization method. This way the kinematic restrictions are encoded in the state lattice structure in such a way that the planning problem is formulated as an unconstrained graph search.

- The uncertainty has been predicted in advance, i.e. before the robot starts executing the planned paths, taking into account the motion and sensing disturbances and the kind of controller. This has been achieved propagating an initial belief throughout the different candidate paths in the state lattice during the execution of the search algorithm. These predicted probability distributions have been used to estimate the probability of collision of each path and prioritize the safety of the plans.

- The Anytime Dynamic A* algorithm [125] has been used to retrieve paths which are optimal according to a hierarchical cost function comprised by three different criteria. Thus, the motion planner minimizes the probability of collision in the first place. Among the paths with the minimal probability of collision, it retrieves the one with the lowest traversal time and, finally, if several candidates exist, the one with the minimum uncertainty at the goal is retrieved. There is the possibility of obtaining sub-optimal solutions faster than the optimal one due to the anytime search capabilities of this algorithm. This is interesting for those practical applications in which the runtime of the planner is of critical importance, because once the first sub-optimal path is obtained it can be refined in subsequent iterations of the motion planner.

- The planning efficiency has been improved due to the graduated fidelity lattice and the use of heuristics that reduce the computational effort of finding optimal solutions. On the one hand, the graduated fidelity has made it possible to represent the robot state space with a lower number of discrete states, thus reducing the number of candidate

paths in selected areas of the environment. On the other hand, two different heuristics have been combined to estimate the cost to the goal taking into account the information of the obstacle configuration and the robot kinematic restrictions.

The work developed in this thesis has contributed to address some drawbacks of prior research in the motion planning field. These contributions have been detailed in Chapter 3, and can be summarized as follows:

- In Section 3.2.1 we have presented a method to reliably estimate the probability of collision of the paths from the predicted uncertainty which is based on sampling the distributions in a deterministic manner. The estimation is obtained checking collisions with the set of sampled poses in such a way that it takes into account the real dimensions of the robot, as well as the uncertainty in heading when sampling the PDFs. This method have been validated in 15 environments of different natures and under a variety of uncertainty conditions, as detailed in Section 3.3.1. The results show that the probability of collision has been reliably estimated in all cases, even for robots with irregular shapes or for problems in which the uncertainty in heading has a great impact in the safety of the paths.

- Section 3.2.2 has introduced an approach for obtaining a graduated fidelity lattice taking into account the obstacles in the map, the maneuverability of the robot and the predicted uncertainty. The proposed method reduces the number of candidate paths in those lattice areas that represent a minor challenge for planning, such as free space or low uncertainty regions. Moreover, the fidelity is selected at the level of actions. This provides the sufficient flexibility to retain the diversity of the lattice connectivity and therefore the quality of the solutions. This technique has been validated with several motion models and environments with very different obstacle configurations —Section 3.3.2—, achieving on average a 88.5% decrease of the runtime when compared to a classical state lattice planner. The motion planning algorithm has also been validated making use of the anytime search capabilities of AD* —in Section 3.3.3— and in real environments —Section 3.3.4.

- In Section 3.2.3 we have proposed a multi-resolution, low dimensional heuristic which estimates the cost to the goal due to the obstacles. This heuristic relies on a grid that represents a simplified version of the planning problem and stores the estimated cost

of moving between each position and the goal. The resolution of this grid is selected according to the obstacle configuration and taking into account the maximum fidelity of the state lattice. The experimental results of Section 3.3.5 show that this heuristic is computed on average a 65.5% faster than the analogue one which uses a fixed resolution grid, and its scalability in large environments is improved as well. At the same time, the proposed heuristic is on average a 4% more precise than the fixed-resolution one, while it retains its admissibility and consistency properties, needed for guaranteeing the optimality of the solutions.

In Chapter 4 we have addressed the application of the proposed motion planning approach for the autonomous navigation of UAVs. This chapter has also detailed the new developed algorithms that deal with the increased dimensionality of the search problem, as well as their validation in real experiments. The contributions can be summarized as follows:

- In Section 4.1 we have presented a flexible architecture for the autonomous navigation of UAVs which has integrated the developed motion planning algorithms with a feedback controller and a state estimation system. Extending the problem of motion planning under uncertainty to 3D environments required developing a new heuristic and a new method for estimating the probability of collision to maintain reasonable planning times. In this regard, Section 4.2.1 detailed how the connectivity of the grid used by the multi-resolution heuristic has been redefined to reduce the density of arcs between adjacent cells while maintaining their diversity. Moreover, in Section 4.2.2 the method for estimating the probability of collision sampling the PDFs has been adapted to reduce the number of extremely low likelihood poses while maintaining a good coverage for collision check purposes. The experimental results of Sections 4.3.1 and 4.3.2 show that the efficiency of the motion planner has enabled the UAV to navigate autonomously without delays. We have conducted flying experiments in 3 different environments and motion models, using a visual SLAM algorithm for the UAV state estimation and showing the capabilities of the proposed system. Each experiment was repeated several times to validate the method for estimating the probability of collision. In this regard, no collisions have been reported during the execution of the plans, and the LQG controller has been able to produce a stable flight while accurately tracking the planned routes.

- From a practical point of view, the proposed autonomous navigation system has been used for the autonomous reconstruction of 3D scenes using a RGB-D camera —Section

4.3.3. The proposed system has been used for the generation of trajectories for obtaining a dense model using the KinectFusion algorithm. Results for this experiment show that the average measured reconstruction error was of 1.08%.

5.1 Current and future work

As a result of the work developed in this thesis, some directions of research that might be worthy to explore have been identified. These ideas focus on further improving the efficiency of the proposed motion planning approach and expanding the applicability of the autonomous navigation strategy to other real world problems.

On the one hand, the motion planner is currently limited to obtaining new solutions from scratch. Although replanning is allowed by AD*, properly integrating this feature would require identifying the candidate paths which are affected by any map changes and modifying the state lattice fidelity in accordance with the last occupancy information. Moreover, due to these modifications, the uncertainties and probabilities of collision of all the affected paths would have to be recomputed, which apart from being computationally expensive is an order sensitive operation. Future research will attempt to find a strategy for minimizing and prioritizing these fidelity changes while retaining the performance of the motion planner, as well as defining the conditions under which it is less computationally expensive to plan from scratch than trying to replan with the information that has been already computed.

On the other hand, from a practical point of view, it would be interesting to extend the proposed autonomous navigation architecture to support sensors of other kinds. In this thesis a RGB-D camera has been used for estimating the state of the UAV during the flight, mainly due to its suitability for indoor tests, its low price and the availability of open source visual SLAM algorithms. However, RGB-D cameras are based on an active ranging strategy which relies on emitting infrared light and measuring its Time of Flight in different points to build a 3D mesh of the obstacles within the camera field of view. The main limitations of RGB-D cameras are their operating range —usually between 0.5 m and 8 m—, the interferences due to natural light and the inaccurate measurements in presence of objects with reflective surfaces. Nowadays there are some high quality, lightweight, low cost alternatives —like stereo or monocular cameras— that could be used to address these issues and enable operating the UAV in environments where active ranging sensors are not suitable. The proposed architecture is flexible enough to allow this without introducing substantial changes in the controller and

planning modules. Notwithstanding, it is necessary to describe proper measurement models to integrate these systems with the uncertainty prediction method.

5.2 Socially aware motion planning

In a context of growing presence of robotic systems that address an increasing number of practical applications, navigating in environments shared with humans has become especially relevant. This problem has to be formulated not uniquely from the robot perspective of *how to react to human presence*, but also from the opposite point of view: *what is this robot expected to do?* In this regard, the robot should behave in a predictable, socially acceptable manner, avoiding to approach humans or execute motions in such a way that might make them feel startled or uncomfortable.

This problem has received an important attention from the research community as human-robot interaction has become increasingly frequent. Recent work in this field that might be used to improve the quality of the motion plans from this perspective include: i) human-robot proxemics [109, 133, 190, 202, 203], which analyzes human reactions to robots in different contexts, and ii) human aware motion planning [104, 170, 183].

The reasons given in Chapter 1 for supporting the necessity of taking into account the motion uncertainty are extensible to navigating in social environments. In this regard, some approaches considered a certain degree of stochasticity in both the robot and human trajectories simulating the different actions that they might take in an immediate time horizon [135]. Others relied on obtaining a socially acceptable navigation behavior from examples [32] with the goal of replicating a human-like behavior. Despite the good results obtained, most approaches have dealt with this problem in such a way that no optimality guarantees in regards to the quality of the paths can be provided, since they have focused on planning only for an immediate time horizon. Thus, it is interesting to extend the motion planning approach proposed in this thesis to produce optimal and socially aware global plans while coping with the robot state uncertainty. An idea that might be worthy to explore would be integrating the theory of human-robot proxemics in the evaluation of the candidate paths.

In this regard, we have taken some preliminary steps in extending the probability of collision estimation method. The same set of samples that check for collisions with the map has been used to measure the acceptability of a given human-robot context. The cost for each sample is obtained from its distance to the nearest human and the category into which that

distance falls according to the interpersonal proxemic distances theory [65, 106] —intimate, personal, social or public distance. Then, the costs of the different samples are combined in accordance with its probability in the distribution that represents the robot state. This measure is easily integrated in our motion planning approach as an additional criterion of the cost function, which is optimized by Anytime Dynamic A* after the probability of collision but before the traversal time.

Current and future research will include considering the vehicle dimensions for evaluating the robot-human positioning, as the robot size affects how people perceive the robot, according to some studies [109, 190]. Another direction of research would consist in also evaluating the paths taking into account the direction of motion of the robot with respect to that of the people around it. This also impacts how humans perceive and react to the robot presence [201] and could be used for the benefit of the navigation in crowded environments [105]. Our approach already takes into account the robot heading and dimensions for planning purposes, and therefore these improvements might be easily integrated in our algorithms.

The final purpose of this socially aware motion planning approach would be the obtention of plans that, apart from prioritizing the safety of the autonomous navigation, allow the robot to move in the presence of humans in such a way that they could describe as natural. As a result of considering the predicted uncertainty in this matter, the solutions would have a human-robot navigation distance adapted to the different circumstances.

Bibliography

[1] Pieter Abbeel. *Apprenticeship learning and reinforcement learning with application to robotic control*. Stanford University, 2008.

[2] Pieter Abbeel, Varun Ganapathi, and Andrew Y. Ng. Learning vehicular dynamics, with application to modeling helicopters. In *Advances in Neural Information Processing Systems (NIPS)*, volume 18, pages 1–8. MIT Press, 2006.

[3] Chaouki Abdallah, Darren M Dawson, Peter Dorato, and Mohammad Jamshidi. Survey of robust control for rigid robots. *Control Systems Magazine*, 11(2):24–30, 1991.

[4] Swiss Post AG. Drones, a vision has become reality. Available at: `https://www.post.ch/en/about-us/innovation/innovations-in-development/drones`. (Online; Accessed on Sept. 18, 2019).

[5] Randa Almadhoun, Tarek Taha, Lakmal Seneviratne, Jorge Dias, and Guowei Cai. A survey on inspecting structures using robotic systems. *International Journal of Advanced Robotic Systems*, 13(6):1–18, 2016.

[6] Ron Alterovitz, Thierry Siméon, and Ken Goldberg. The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty. In *Robotics: Science and Systems*, pages 246–253, 2007.

[7] David A. Anisi, Johan Hamberg, and Xiaoming Hu. Nearly time-optimal paths for a ground vehicle. *Journal of Control Theory and Applications*, 1:2–8, 2003.

[8] Karl J Astrom. Optimal control of markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965.

[9] M. Athans and P. Falb. *Optimal Control*. McGraw-Hill, 1966.

[10] J. A. Bagnell and J. G. Schneider. Autonomous helicopter control using reinforcement learning policy search methods. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1615–1620, 2001.

[11] Haoyu Bai, David Hsu, Wee Sun Lee, and Ngo Anh Vien. Monte carlo value iteration for continuous-state pomdps. In *Algorithmic Foundations of Robotics IX - Selected Contributions of the Ninth International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, pages 175–191, 2010.

[12] Jérôme Barraquand, Lydia E. Kavraki, Jean-Claude Latombe, Tsai-Yen Li, Rajeev Motwani, and Prabhakar Raghavan. A random sampling scheme for path planning. *International Journal of Robotics Research*, 16(6):759–774, 1997.

[13] Jérôme Barraquand, Bruno Langlois, and Jean-Claude Latombe. Numerical potential field techniques for robot path planning. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(2):224–241, 1992.

[14] Jérôme Barraquand and Jean-Claude Latombe. Robot motion planning: A distributed representation approach. *International Journal of Robotics Research*, 10(6):628–649, 1991.

[15] Richard Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, pages 679–684, 1957.

[16] Dmitry Berenson, Thierry Siméon, and Siddhartha S. Srinivasa. Addressing cost-space chasms in manipulation planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4561–4568, 2011.

[17] Dimitri Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

[18] Robert Bohlin. Path planning in practice; lazy evaluation on a multi-resolution grid. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 49–54, 2001.

[19] Robert Bohlin and Lydia E. Kavraki. Path planning using lazy PRM. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 521–528, 2000.

[20] Michael S. Branicky, Ross A. Knepper, and James J. Kuffner. Path and trajectory diversity: Theory and algorithms. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1359–1364, 2008.

[21] Michael S. Branicky, Steven M. LaValle, Kari Olson, and Libo Yang. Quasi-randomized path planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1481–1487, 2001.

[22] Douglas A Bristow, Marina Tharayil, and Andrew G Alleyne. A survey of iterative learning control. *Control Systems Magazine*, 26(3):96–114, 2006.

[23] Alex Brooks, Alexei Makarenko, Stefan B. Williams, and Hugh F. Durrant-Whyte. Parametric pomdps for planning in continuous state spaces. *Robotics and Autonomous Systems*, 54(11):887–897, 2006.

[24] Rodney A. Brooks and Tomás Lozano-Pérez. A subdivision algorithm in configuration space for findpath with rotation. In *International Joint Conference on Artificial Intelligence*, 1983.

[25] Rodney A. Brooks and Tomás Lozano-Pérez. A subdivision algorithm in configuration space for findpath with rotation. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(2):224–233, 1985.

[26] Adam Bry and Nicholas Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 723–730, 2011.

[27] Arthur E. Bryson and Yu-Chi Ho. *Applied Optimal Control: Optimization, estimation and control*. Hemisphere Publishing, 1975.

[28] Guowei Cai, Jorge Dias, and Lakmal Seneviratne. A survey of small-scale unmanned aerial vehicles: Recent advances and future development trends. *Unmanned Systems*, 2(02):175–199, 2014.

[29] Salvatore Candido and Seth Hutchinson. Minimum uncertainty robot navigation using information-guided POMDP planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6102–6108, 2011.

[30] John Canny. *The complexity of robot motion planning*. MIT press, 1988.

[31] Brodie Chan, Hong Guan, Jun Jo, and Michael Blumenstein. Towards uav-based bridge inspection systems: A review and an application perspective. *Structural Monitoring and Maintenance*, 2(3):283–300, 2015.

[32] Yu Fan Chen, Michael Everett, Miao Liu, and Jonathan P How. Socially aware motion planning with deep reinforcement learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1343–1350, 2017.

[33] Peng Cheng and Steven M. LaValle. Resolution complete rapidly-exploring random trees. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 267–272, 2002.

[34] Benjamin Coifman, Mark McCord, Rabi G Mishalani, Michael Iswalt, and Yuxiong Ji. Roadway traffic monitoring from an unmanned aerial vehicle. In *IEE Proceedings-Intelligent Transport Systems*, volume 153, pages 11–20, 2006.

[35] Xunta de Galicia Consellería de Economía, Empleo e Industria. La Xunta licita un pionero sistema para asistir con la ayuda de drones a los peregrinos que sufran una parada cardiorrespiratoria. Sept. 2019. (Online; Accessed on Sept. 18, 2019).

[36] Joseph C. Culberson and Jonathan Schaeffer. Pattern databases. *Computational Intelligence*, 14(3):318–334, 1998.

[37] M. Montemerlo D. Dolgov, S. Thrun and J. Diebel. *Path Planning for Autonomous Driving in Unknown Environments*, page 55–64. Springer, 2009.

[38] Carl De Boor. *A Practical Guide to Splines*, volume 27 of *Applied Mathematical Sciences*. Springer-Verlag New York, 1978.

[39] Lester E Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79:497–516, 1957.

[40] Milan Erdelj, Enrico Natalizio, Kaushik R. Chowdhury, and Ian F. Akyildiz. Help from the sky: Leveraging uavs for disaster management. *IEEE Pervasive Computing*, 16(1):24–32, 2017.

[41] Tom Erez and William D. Smart. A scalable method for solving high-dimensional continuous pomdps using local approximation. In *Twenty-Sixth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 160–167, 2010.

[42] Lawrence H. Erickson and Steven M. LaValle. Survivability: Measuring and ensuring path diversity. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2068–2073, 2009.

[43] Carlos Alphonso F Ezequiel, Matthew Cua, Nathaniel C Libatique, Gregory L Tangonan, Raphael Alampay, Rollyn T Labuguen, Chrisandro M Favila, Jaime Luis E Honrado, Vinni Canos, Charles Devaney, et al. Uav aerial imaging applications for post-disaster assessment, environmental management and infrastructure development. In *IEEE International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 274–283, 2014.

[44] Timm Faulwasser, Benjamin Kern, and Rolf Findeisen. Model predictive path-following for constrained nonlinear systems. In *IEEE Conference on Decision and Control (CDC)*, pages 8642–8647, 2009.

[45] Dave Ferguson and Anthony Stentz. Anytime rrts. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5369–5375, 2006.

[46] Gerardo Flores, Shuting Zhou, Rogelio Lozano, and Pedro Castillo. A vision and GPS-based real-time trajectory planning for a MAV in unknown and low-sunlight environments. *Journal of Intelligent & Robotic Systems*, 74(1-2):59–67, 2014.

[47] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Sixteenth National Conference on Artificial Intelligence and Eleventh Conference on Innovative Applications of Artificial Intelligence (AAAI)*, pages 343–349, 1999.

[48] Thierry Fraichard and J-M Ahuactzin. Smooth path planning for cars. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (ICRA)*, pages 3722–3727, 2001.

[49] Thierry Fraichard and Alexis Scheuer. From reeds and shepp's to continuous-curvature paths. *IEEE Transactions on Robotics*, 20:1025–1035, 2004.

[50] E. Frazzoli, M. A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. In *American Control Conference*, pages 43–49, 2001.

[51] Emilio Frazzoli, Munther A Dahleh, and Eric Feron. Real-time motion planning for agile autonomous vehicles. *Journal of guidance, control, and dynamics*, 25(1):116–129, 2002.

[52] Jorge Fuentes-Pacheco, José Ruíz Ascencio, and Juan Manuel Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review.*, 43(1):55–81, 2015.

[53] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2997–3004, 2014.

[54] Santiago Garrido, Luis Moreno, Mohamed Abderrahim, and Dolores Blanco. Fm2: a real-time sensor-based feedback controller for mobile robots. *International Journal of Robotics & Automation*, 24(1):48, 2009.

[55] Santiago Garrido, Luis Moreno, Mohamed Abderrahim, and Fernando Martín Monar. Path planning for mobile robot navigation using voronoi diagram and fast marching. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2376–2381, 2006.

[56] Santiago Garrido, Luis Moreno, and Dolores Blanco. Voronoi diagram and fast marching applied to path planning. In *IEEE International Conference on Robotics and Automation, (ICRA)*, pages 3049–3054, 2006.

[57] DHL International GmbH. DHL Parcelcopter. Available at: `https://www.dpdhl.com/en/media-relations/specials/dhl-parcelcopter.html`, Apr. 2018.

[58] Chad Goerzen, Zhaodan Kong, and Bernard Mettler. A survey of motion planning algorithms from the perspective of autonomous uav guidance. *Journal of Intelligent and Robotic Systems*, 57(1-4):65, 2010.

[59] David González, Joshué Pérez, Vicente Milanés, and Fawzi Nashashibi. A review of motion planning techniques for automated vehicles. *IEEE Trans. Intelligent Transportation Systems*, 17(4):1135–1145, 2016.

[60] V. González, Concepción Alicia Monje Micharet, Luis Moreno, and Carlos Balaguer. Uavs mission planning with flight level constraint using fast marching square method. *Robotics and Autonomous Systems*, 94:162–171, 2017.

[61] Adrián González-Sieira, Manuel Mucientes, and Alberto Bugarín. Motion planning under uncertainty in graduated fidelity lattices. *Robotics and Autonomous Systems*, 109:168 – 182, 2018.

[62] Colin J. Green and Alonzo Kelly. Toward optimal sampling in the space of paths. In *International Symposium on Robotics Research (ISSR)*, pages 281–292, 2007.

[63] Giorgio Grisettiyz, Cyrill Stachniss, and Wolfram Burgard. Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2432–2437, 2005.

[64] Jürgen Guldner and Vadim I Utkin. Sliding mode control for gradient tracking and robot navigation using artificial potential fields. *IEEE Transactions on Robotics and Automation*, 11(2):247–254, 1995.

[65] E. Hall. *The hidden dimension*. Anchor Books, 1966.

[66] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[67] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. Correction to "a formal basis for the heuristic determination of minimum cost paths". *SIGART Newsletter*, 37:28–29, 1972.

[68] Kris K. Hauser. Randomized belief-space replanning in partially-observable continuous spaces. In *Ninth International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, pages 193–209, 2010.

[69] Armin Hornung, Kai M Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206, 2013.

[70] Thomas M Howard and Alonzo Kelly. Optimal rough terrain trajectory generation for wheeled mobile robots. *The International Journal of Robotics Research*, 26(2):141–166, 2007.

[71] David Hsu. *Randomized Single-query Motion Planning in Expansive Spaces*. PhD thesis, 2000.

[72] David Hsu, Lydia E Kavraki, Jean-Claude Latombe, Rajeev Motwani, Stephen Sorkin, et al. On finding narrow passages with probabilistic roadmap planners. In *Workshop on the Algorithmic Foundations of Robotics*, pages 141–154, 1998.

[73] David Hsu, Robert Kindel, Jean-Claude Latombe, and Stephen M. Rock. Randomized kinodynamic motion planning with moving obstacles. *International Journal of Robotics Research*, 21(3):233–256, 2002.

[74] David Hsu, Jean-Claude Latombe, and Hanna Kurniawati. On the probabilistic foundations of probabilistic roadmap planning. *International Journal of Robotics Research*, 25(7):627–643, 2006.

[75] David Hsu, Jean-Claude Latombe, and Rajeev Motwani. Path planning in expansive configuration spaces. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2719–2726, 1997.

[76] David Hsu, Jean-Claude Latombe, and Rajeev Motwani. Path planning in expansive configuration spaces. *International Journal of Computational Geometry & Applications*, 9(04):495–512, 1999.

[77] Vu Anh Huynh and Nicholas Roy. iclqg: Combining local and global optimization for control in information space. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2851–2858, 2009.

[78] Yong K Hwang and Narendra Ahuja. Gross motion planning—a survey. *ACM Computing Surveys (CSUR)*, 24(3):219–291, 1992.

[79] Yong Koo Hwang and Narendra Ahuja. A potential field approach to path planning. *IEEE Transactions on Robotics and Automation*, 8(1):23–32, 1992.

[80] Massimiliano Iacono and Antonio Sgorbissa. Path following and obstacle avoidance for an autonomous UAV using a depth camera. *Robotics and Autonomous Systems*, 106:38–46, 2018.

[81] Alphabet Inc. Wing, Transforming the way goods are transported. Available at: `https://x.company/projects/wing`, Jul. 2018. (Online; Accessed on Sept. 18, 2019).

[82] Amazon.com Inc. Amazon prime Air. Available at: `https://www.amazon.com/primeair`. (Online; Accessed on Sept. 18, 2019).

[83] Leonard Jaillet, Juan Cortés, and Thierry Siméon. Sampling-based path planning on configuration-space costmaps. *IEEE Transactions on Robotics*, 26(4):635–646, 2010.

[84] Lucas Janson, Brian Ichter, and Marco Pavone. *Deterministic Sampling-Based Motion Planning: Optimality, Complexity, and Performance*, volume 2, pages 507–525. Springer International Publishing, 2018.

[85] Lucas Janson, Brian Ichter, and Marco Pavone. Deterministic sampling-based motion planning: Optimality, complexity, and performance. *International Journal of Robotics Research*, 37(1):46–61, 2018.

[86] James J. Kuffner Jr. and Steven M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 995–1001, 2000.

[87] Robert Platt Jr., Russ Tedrake, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Belief space planning assuming maximum likelihood observations. In *Robotics: Science and Systems VI*, 2010.

[88] Simon J Julier, Jeffrey K Uhlmann, and Hugh F Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proceedings of the American Control Conference*, volume 3, pages 1628–1632, 1995.

[89] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134, 1998.

[90] Sertac Karaman and Emilio Frazzoli. Incremental sampling-based algorithms for optimal motion planning. *Robotics Science and Systems VI*, 104, 2010.

[91] L. E. Kavraki. *Random Networks in Configuration Space for Fast Path Planning*. PhD thesis, 1994.

[92] L. E. Kavraki, P. Svestka, J. . Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12:566–580, 1996.

[93] Lydia E. Kavraki, Mihail N. Kolountzakis, and Jean-Claude Latombe. Analysis of probabilistic roadmaps for path planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3020–3025, 1996.

[94] Lydia E. Kavraki, Mihail N. Kolountzakis, and Jean-Claude Latombe. Analysis of probabilistic roadmaps for path planning. *IEEE Transactions on Robotics and Automation*, 14(1):166–171, 1998.

[95] Lydia E. Kavraki, Jean-Claude Latombe, Rajeev Motwani, and Prabhakar Raghavan. Randomized query processing in robot path planning. *Journal of Computer and System Sciences*, 57(1):50–66, 1998.

[96] Alonzo Kelly and Bryan Nagy. Reactive nonholonomic trajectory generation via parametric optimal control. *The International Journal of Robotics Research*, 22:583 – 601, 2003.

[97] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 500–505, 1985.

[98] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1):90–98, 1986.

[99] H Jin Kim, Michael I Jordan, Shankar Sastry, and Andrew Y Ng. Autonomous helicopter flight via reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 799–806, 2004.

[100] Ross Knepper and Alonzo Kelly. High Performance State Lattice Planning Using Heuristic Look-Up Tables. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3375–3380, 2006.

[101] Sven Koenig and Maxim Likhachev. Fast replanning for navigation in unknown terrain. *IEEE Transactions on Robotics*, 21(3):354–363, 2005.

[102] Yoram Koren and Johann Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1398–1404, 1991.

[103] Richard E. Korf and Michael Reid. Complexity analysis of admissible heuristic search. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference, (AAAI)*, pages 305–310, 1998.

[104] Thibault Kruse, Patrizia Basili, Stefan Glasauer, and Alexandra Kirsch. Legible robot navigation in the proximity of moving humans. In *IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*, pages 83–88, 2012.

[105] Thibault Kruse, Alexandra Kirsch, Emrah Akin Sisbot, and Rachid Alami. Exploiting human cooperation in human-centered robot navigation. In *19th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 192–197, 2010.

[106] Thibault Kruse, Amit Kumar Pandey, Rachid Alami, and Alexandra Kirsch. Human-aware robot navigation: A survey. *Robotics and Autonomous Systems*, 61(12):1726–1743, 2013.

[107] Hanna Kurniawati, David Hsu, and Wee Sun Lee. SARSOP: efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems*, volume IV, 2008.

[108] Andrew M. Ladd and Lydia E. Kavraki. Measure theoretic analysis of probabilistic path planning. *IEEE Transactions on Robotics and Automation*, 20(2):229–242, 2004.

[109] Hamid Laga and Toshitaka Amaoka. Modeling the spatial behavior of virtual agents in groups for non-verbal communication in virtual worlds. In *Proceedings of the 3rd International Universal Communication Symposium, IUCS*, pages 154–159, 2009.

[110] Florent Lamiraux and J-P Lammond. Smooth motion planning for car-like vehicles. *IEEE Transactions on Robotics and Automation*, 17:498–501, 2001.

[111] Jean-Claude Latombe. *Robot motion planning*, volume 124 of *The Springer International Series in Engineering and Computer Science*. 1991.

[112] Jean-Claude Latombe, Anthony Lazanas, and Shashank Shekhar. Robot motion planning with uncertainty in control and sensing. *Artificial Intelligence*, 52(1):1–47, 1991.

[113] J. . Laumond, P. E. Jacobs, M. Taix, and R. M. Murray. A motion planner for nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 10:577–593, 1994.

[114] J. P. Laumond. *Robot Motion Planning and Control*, volume 229 of *Lecture Notes in Control and Information Sciences*. Springer, 1998.

[115] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006.

[116] Steven M. Lavalle. Rapidly-exploring random trees: A new tool for path planning. Technical report, Department of Computer Science. Iowa State University, 1998.

[117] Steven M. LaValle, Michael S. Branicky, and Stephen R. Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *International Journal of Robotics Research*, 23(7-8):673–692, 2004.

[118] Steven M. LaValle and James J. Kuffner Jr. Randomized kinodynamic planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 473–479, 1999.

[119] Steven M LaValle and James J Kuffner. Randomized kinodynamic planning. *The International Journal of Robotics Research*, 20(5):378–400, 2001.

[120] Sang Uk Lee, Ramon Gonzalez, and Karl Iagnemma. Robust sampling-based motion planning for autonomous tracked vehicles in deformable high slip terrain. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2569–2574, 2016.

[121] Robert C Leishman, Timothy W McLain, and Randal W Beard. Relative navigation approach for vision-based aerial GPS-denied navigation. *Journal of Intelligent & Robotic Systems*, 74(1-2):97–111, 2014.

[122] Qing Li, Da-Chuan Li, Qin-fan Wu, Liang-wen Tang, Yan Huo, Yi-xuan Zhang, and Nong Cheng. Autonomous navigation and environment modeling for MAVs in 3-D enclosed industrial environments. *Computers in Industry*, 64(9):1161–1177, 2013.

[123] M. Likhachev and D. Ferguson. Planning Long Dynamically Feasible Maneuvers for Autonomous Vehicles. *The International Journal of Robotics Research*, 28(8):933–945, June 2009.

[124] Maxim Likhachev and Dave Ferguson. Planning long dynamically-feasible maneuvers for autonomous vehicles. In *Robotics: Science and Systems*, 2008.

[125] Maxim Likhachev, Dave Ferguson, Geoff Gordon, Anthony Stentz, and Sebastian Thrun. Anytime dynamic A*: An anytime, replanning algorithm. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, pages 262–271, 2005.

[126] Maxim Likhachev, Dave Ferguson, Geoff Gordon, Anthony Stentz, and Sebastian Thrun. Anytime search in dynamic graphs. *Artificial Intelligence*, 172(14):1613–1643, 2008.

[127] Stephen R. Lindemann and Steven M. LaValle. Incrementally reducing dispersion by increasing voronoi bias in rrts. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3251–3257, 2004.

[128] Stephen R. Lindemann and Steven M. LaValle. Steps toward derandomizing rrts. In *Proceedings of the Fourth International Workshop on Robot Motion and Control (RoMoCo)*, pages 271–277, 2004.

[129] Tomás Lozano-Pérez and Michael A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Commun. ACM*, 22(10):560–570, 1979.

[130] Brandon D Luders, Sertac Karaman, and Jonathan P How. Robust sampling-based motion planning with asymptotic optimality guarantees. In *AIAA Guidance, Navigation, and Control (GNC) Conference*, page 5097, 2013.

[131] Brian MacAllister, Jonathan Butzke, Alex Kushleyev, Harsh Pandey, and Maxim Likhachev. Path planning for non-circular micro aerial vehicles in constrained environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3933–3940, 2013.

[132] Emmanuel Mazer, Juan Manuel Ahuactzin, and Pierre Bessiére. The ariadne's clew algorithm. *Journal of Artificial Intelligence Research*, 9:295–316, 1998.

[133] Ross Mead, Amin Atrash, and M Mataric. Automated analysis of proxemic behavior: Leveraging metrics from the social sciences. In *Proceedings of the Robotics: Science and systems workshop on human-robot interaction: Perspectives and contributions to robotics from the human sciences*, 2011.

[134] Donald Meagher. Geometric modeling using octree encoding. *Computer graphics and image processing*, 19(2):129–147, 1982.

[135] Dhanvin Mehta, Gonzalo Ferrer, and Edwin Olson. Autonomous navigation in dynamic social environments using multi-policy decision making. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1190–1197, 2016.

[136] Bernard Mettler, Takeo Kanade, and Mark Brian Tischler. *System identification modeling of a model-scale helicopter*, pages 1–25. Carnegie Mellon University, The Robotics Institute Archive, 2000.

[137] Bernard Mettler, Mark B. Tischler, and Takeo Kanade. System identification of small-size unmanned helicopter dynamics. In *55th Forum of the American Helicopter Society*, 1999.

[138] Bernard Mettler, Mark B Tischler, and Takeo Kanade. System identification of small-size unmanned helicopter dynamics. In *Proceedings of the American Helicopter Society Annual Forum*, volume 2, pages 1706–1717, 1999.

[139] Scott A. Miller, Zachary A. Harris, and Edwin K. P. Chong. A POMDP framework for coordinated guidance of autonomous uavs for multitarget tracking. *EURASIP Journal on Advances in Signal Processing*, 2009.

[140] Raul Mur-Artal and Juan D Tardós. ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017.

[141] Richard M Murray and S Shankar Sastry. Steering nonholonomic systems in chained form. In *30th IEEE Conference on Decision and Control*, pages 1121–1126, 1991.

[142] Tobias Nägeli, Lukas Meier, Alexander Domahidi, Javier Alonso-Mora, and Otmar Hilliges. Real-time planning for automated multi-view drone cinematography. *ACM Transactions on Graphics (TOG)*, 36(4):132, 2017.

[143] Alex Nash, Kenny Daniel, Sven Koenig, and Ariel Felner. Theta*: Any-angle path planning on grids. In *Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI)*, pages 1177–1183, 2007.

[144] Alex Nash, Sven Koenig, and Craig A. Tovey. Lazy theta*: Any-angle path planning and path length analysis in 3d. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI)*, 2010.

[145] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *10th IEEE international symposium on Mixed and augmented reality (ISMAR)*, pages 127–136, 2011.

[146] Francesco Nex and Fabio Remondino. UAV for 3D mapping applications: a review. *Applied Geomatics*, 6(1):1–15, 2014.

[147] Harald Niederreiter. *Random number generation and quasi-Monte Carlo methods*, volume 63. Siam, 1992.

[148] Matthias Nieuwenhuisen, David Droeschel, Marius Beul, and Sven Behnke. Obstacle detection and navigation planning for autonomous micro aerial vehicles. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1040–1047, 2014.

[149] Helen Oleynikova, Michael Burri, Zachary Taylor, Juan Nieto, Roland Siegwart, and Enric Galceran. Continuous-time trajectory optimization for online UAV replanning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5332–5339, 2016.

[150] Miguel A. Olivares-Méndez, Iván Fernando Mondragón, Pascual Campoy Cervera, and Carol Chamorro-Martínez. Fuzzy controller for uav-landing task using 3d-position visual estimation. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–8, 2010.

[151] Sylvie C. W. Ong, Shao Wei Png, David Hsu, and Wee Sun Lee. Planning under uncertainty for robotic tasks with mixed observability. *International Journal of Robotics Research*, 29(8):1053–1068, 2010.

[152] Mark H. Overmars and Petr Svestka. A probabilistic learning approach to motion planning. technical report uu-cs-1994-03. Technical report, Department of Computer Science. Utrecht University, Jan 1994.

[153] Brian Paden, Michal Cáp, Sze Zheng Yong, Dmitry S. Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1):33–55, 2016.

[154] Christos H Papadimitriou and John N Tsitsiklis. The complexity of Markov decision processes. *Mathematics of operations research*, 12(3):441–450, 1987.

[155] Judea Pearl. *Heuristics - Intelligent Search Strategies for Computer Problem Solving*. Series in Artificial Intelligence. Addison-Wesley, 1984.

[156] Francisco J Perez-Grau, Ricardo Ragel, Fernando Caballero, Antidio Viguria, and Anibal Ollero. An architecture for robust UAV navigation in GPS-denied areas. *Journal of Field Robotics*, 35(1):121–145, 2018.

[157] Clment Petres, Yan Pailhas, Pedro Patron, Yvan Petillot, Jonathan Evans, and David Lane. Path planning for autonomous underwater vehicles. *IEEE Transactions on Robotics*, 23(2):331–341, 2007.

[158] Mihail Pivtoraiko and Alonzo Kelly. Efficient constrained path planning via search in state lattices. In *8th International Symposium on Artificial Intelligence, Robotics and Automation (I-SAIRAS)*, 2005.

[159] Mihail Pivtoraiko and Alonzo Kelly. Differentially constrained motion replanning using state lattices with graduated fidelity. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2611–2616, 2008.

[160] Mihail Pivtoraiko and Alonzo Kelly. Kinodynamic motion planning with state lattice motion primitives. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2172–2179, 2011.

[161] Mihail Pivtoraiko, Ross A Knepper, and Alonzo Kelly. Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics*, 26(3):308–333, 2009.

[162] Mihail Pivtoraiko, Daniel Mellinger, and Vijay Kumar. Incremental micro-UAV motion replanning for exploring unknown environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2452–2458, 2013.

[163] Samuel Prentice and Nicholas Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *The International Journal of Robotics Research*, 28(11-12):1448–1465, 2009.

[164] Anuj Puri. A survey of unmanned aerial vehicles (UAV) for traffic surveillance. *Department of computer science and engineering, University of South Florida*, pages 1–29, 2005.

[165] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5, 2009.

[166] Shankarachary Ragi and Edwin K. P. Chong. UAV path planning in a dynamic environment via partially observable markov decision process. *IEEE Transactions on Aerospace and Electronic Systems*, 49(4):2397–2412, 2013.

[167] James Reeds and Lawrence Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145:367–393, 1990.

[168] John H. Reif. Complexity of the mover's problem and generalizations (extended abstract). In *20th Annual Symposium on Foundations of Computer Science*, pages 421–427, 1979.

[169] Elon Rimon and Daniel E. Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, 1992.

[170] Jorge Rios-Martinez, Anne Spalanzani, and Christian Laugier. Understanding human interaction for probabilistic autonomous navigation using risk-rrt approach. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2014–2019, 2011.

[171] P Rodriguez-Mier, A Gonzalez-Sieira, Manuel Mucientes, Manuel Lama, and A Bugarin. Hipster: An open source java library for heuristic search. In *9th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6, 2014.

[172] Fernando Ropero, Pablo Muñoz, and María D. R-Moreno. TERRA: A path planning algorithm for cooperative UGV–UAV exploration. *Engineering Applications of Artificial Intelligence*, 78:260 – 272, 2019.

[173] Maurizio Rossi, Davide Brunelli, Andrea Adami, Leandro Lorenzelli, Fabio Menna, and Fabio Remondino. Gas-drone: Portable gas sensing system on uavs for gas leakage localization. In *IEEE SENSORS*, pages 1431–1434, 2014.

[174] Nicholas Roy and Sebastian Thrun. Coastal navigation with mobile robots. In *Advances in Neural Information Processing Systems*, pages 1043–1049, 2000.

[175] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson Education, 2016.

[176] Inkyu Sa, Hu He, Van Huynh, and Peter Corke. Monocular vision based autonomous navigation for a cost-effective MAV in GPS-denied environments. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 1355–1360, 2013.

[177] Gildardo Sánchez-Ante and Jean-Claude Latombe. A single-query bi-directional probabilistic roadmap planner with lazy collision checking. In *International Symposium on Robotics Research (ISRR)*, pages 403–417, 2001.

[178] Gildardo Sánchez-Ante and Jean-Claude Latombe. On delaying collision checking in PRM planning. *International Journal of Robotics Research*, 21(1):5–26, 2002.

[179] Alexis Scheuer and Th Fraichard. Continuous-curvature path planning for car-like vehicles. In *IEEE/RSJ International Conference on Intelligent Robot and Systems (IROS). Innovative Robotics for Real-World Applications.*, pages 997–1003, 1997.

[180] Alexis Scheuer and Christian Laugier. Planning sub-optimal and continuous-curvature paths for car-like robots. In *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Innovations in Theory, Practice and Applications.*, pages 25–31, 1998.

[181] Jacob T Schwartz and Micha Sharir. On the "piano movers" problem. ii. general techniques for computing topological properties of real algebraic manifolds. *Advances in applied Mathematics*, 4(3):298–351, 1983.

[182] Jacob T Schwartz and Chee-Keng Yap. *Advances in Robotics: Algorithmic and geometric aspects of robotics*. Lawrence Erlbaum Assoc., Hillsdale, New Jersey, 1986.

[183] Stephan Sehestedt, Sarath Kodagoda, and Gamini Dissanayake. Robot path planning in a social context. In *IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 206–211, 2010.

[184] Eduard Semsch, Michal Jakob, Dušan Pavlicek, and Michal Pechoucek. Autonomous UAV surveillance in complex urban environments. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 02*, pages 82–85, 2009.

[185] James A Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[186] Liang Shi, Xin Wang, Tao Zhang, Chunrong Hu, Kaixin Luo, and Bing Bai. Hazardous gas detection four-rotor uav system development. In *IEEE International Conference on Mechatronics and Automation*, pages 2461–2465, 2016.

[187] Anthony Stentz. Optimal and efficient path planning for partially-known environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3310–3317, 1994.

[188] Anthony Stentz. The focussed d* algorithm for real-time replanning. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1652–1659, 1995.

[189] Martin Stokkeland, Kristian Klausen, and Tor A Johansen. Autonomous visual navigation of unmanned aerial vehicle for wind turbine inspection. In *IEEE International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 998–1007, 2015.

[190] Mikael Svenstrup, Søren Tranberg Hansen, Hans Jørgen Andersen, and Thomas Bak. Pose estimation and adaptive robot behaviour for human-robot interaction. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3571–3576, 2009.

[191] Shu Tezuka. *Uniform random numbers: Theory and practice*. The Springer International Series in Engineering and Computer Science.

[192] Cornelius A Thiels, Johnathon M Aho, Scott P Zietlow, and Donald H Jenkins. Use of unmanned aerial vehicles for medical product transport. *Air Medical Journal*, 34(2):104–108, 2015.

[193] Mark B Tischler and Mavis G Cauffman. Frequency-response method for rotorcraft system identification: Flight applications to BO 105 coupled rotor/fuselage dynamics. *Journal of the American Helicopter Society*, 37(3):3–17, 1992.

[194] Emanuel Todorov and Weiwei Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *IEEE American Control Conference*, pages 300–306, 2005.

[195] Noel E. Du Toit and Joel W. Burdick. Robotic motion planning in dynamic, cluttered, uncertain environments. In *IEEE International Conference on Robotics and Automation, (ICRA)*, pages 966–973, 2010.

[196] Chris Urmson and Reid G. Simmons. Approaches for heuristically biasing RRT growth. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1178–1183, 2003.

[197] Jur Van den Berg, Pieter Abbeel, and Ken Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *The International Journal of Robotics Research*, 30(7):895–913, 2011.

[198] Jur van den Berg, Sachin Patil, and Ron Alterovitz. Efficient approximate value iteration for continuous gaussian pomdps. In *Twenty-Sixth Conference on Artificial Intelligence (AAAI)*, 2012.

[199] Jur Van den Berg, Sachin Patil, and Ron Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, 31(11):1263–1278, 2012.

[200] Anurag Sai Vempati, Mina Kamel, Nikola Stilinovic, Qixuan Zhang, Dorothea Reusser, Inkyu Sa, Juan I. Nieto, Roland Siegwart, and Paul A. Beardsley. Paintcopter: An autonomous UAV for spray painting on three-dimensional surfaces. *IEEE Robotics and Automation Letters*, 3(4):2862–2869, 2018.

[201] Jered Vroon, Michiel Joosse, Manja Lohse, Jan Kolkmeier, Jaebok Kim, Khiet P. Truong, Gwenn Englebienne, Dirk Heylen, and Vanessa Evers. Dynamics of social positioning patterns in group-robot interactions. In *24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 394–399, 2015.

[202] Michael L Walters, Kerstin Dautenhahn, René Te Boekhorst, Kheng Lee Koay, Dag Sverre Syrdal, and Chrystopher L Nehaniv. An empirical framework for human-robot proxemics. 2009.

[203] Michael L. Walters, Kheng Lee Koay, Sarah N. Woods, Dag Sverre Syrdal, and Kerstin Dautenhahn. Robot to human approaches: Preliminary results on comfortable distances and preferences. In *AAAI Spring Symposium on Multidisciplinary Collaboration for Socially Assistive Robotics*, pages 103–109, 2007.

[204] Long Wang and Zijun Zhang. Automatic detection of wind turbine blade surface cracks based on uav-taken images. *IEEE Transactions on Industrial Electronics*, 64(9):7293–7303, 2017.

[205] Nathan A Wedge and Michael S Branicky. On heavy-tailed runtimes and restarts in rapidly-exploring random trees. In *AAAI Conference on Artificial Intelligence*, pages 127–133, 2008.

[206] Anna Yershova, Leonard Jaillet, Thierry Siméon, and Steven M. LaValle. Dynamic-domain rrts: Efficient exploration by controlling the sampling domain. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3856–3861, 2005.

[207] Chul Min Yeum and Shirley J Dyke. Vision-based automated crack detection for bridge inspection. *Computer-Aided Civil and Infrastructure Engineering*, 30(10):759–770, 2015.

[208] Huili Yu and Randy Beard. A vision-based collision avoidance technique for micro air vehicles using local-level frame mapping and path planning. *Autonomous Robots*, 34(1-2):93–109, 2013.

[209] Chi Yuan, Zhixiang Liu, and Youmin Zhang. Uav-based forest fire detection and tracking using image processing techniques. In *IEEE International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 639–643, 2015.

[210] Kemin Zhou, John Comstock Doyle, Keith Glover, et al. *Robust and optimal control*. Prentice Hall, 1996.

[211] Simon Zingg, Davide Scaramuzza, Stephan Weiss, and Roland Siegwart. MAV navigation through indoor corridors using optical flow. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3361–3368, 2010.

List of Figures

List of Tables

List of Acronyms

AD*	Anytime Dynamic A*
BFS	Breadth First Search
BRM	Belief Roadmap
EKF	Extended Kalman Filter
FM	Fast Marching
FM^2	Fast Marching Square
FSH	Free Space Heuristic
GF	Graduated Fidelity
GPS	Global Positioning System
HLUT	Heuristic Look-Up Table
hRRT	Heuristically-guided Rapidly-exploring Random Tree
LQG	Linear Quadratic Gaussian
LQG-MP	Linear Quadratic Gaussian Motion Planning
LRM	Lattice Roadmap
MDP	Markov Decision Process
MOMDP	Mixed Observable Markov Decision Process

MPC	Model Predictive Control
NBO	Nominal Belief-state optimization
PD	Proportional-Derivative
PDF	Probability Density Function
POMDP	Partially Observable Markov Decision Process
PRM	Probabilistic Roadmap
PRM*	Optimal Probabilistic Roadmap
RRG	Rapidly-exploring Random Graph
RRT	Rapidly-exploring Random Tree
RRT*	Optimal Rapidly-exploring Random Tree
SLAM	Simultaneous Localization and Mapping
SMRM	Stochastic Motion Roadmap
sPRM	Simple Probabilistic Roadmap
T-RRT	Transition-based Rapidly-exploring Random Tree
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter