UNIVERSIDAD DE SALAMANCA

Programa de Doctorado

Informática y Automática

Tesis Doctoral

Procedimientos de obtención de configuraciones de agarre en manipulación robótica

Autora:

Ángeles Mª Moreno Montero

Directores:

Dra. Belén Curto Diego

Dr. Vidal Moreno Rodilla

Salamanca, abril 2013

Dra. **Belén Curto Diego** y Dr. **Vidal Moreno Rodilla**, Profesores Titulares de Universidad del área de Ingeniería de Sistemas y Automática de la Universidad de Salamanca

CERTIFICAN:

Que el trabajo de investigación que se recoge en la presente memoria, titulado **Procedimientos de obtención de configuraciones de agarre en manipulación robótica** y presentada por Dª **Ángeles Mª Moreno Montero** para optar al grado de Doctora por la Universidad de Salamanca, ha sido realizado bajo su dirección en el Departamento de Informática y Automática de la Universidad de Salamanca.

Salamanca, 18 de abril de 2013

Dra. **Belén Curto Diego**
Profa. Titular de Universidad
Área de Ingeniería de Sistemas y Automática
Universidad de Salamanca

Dr. **Vidal Moreno Rodilla**
Prof. Titular de Universidad
Área de Ingeniería de Sistemas y Automática
Universidad de Salamanca

A mis padres y hermanos

Agradecimientos

En primer lugar quiero dar las gracias a mis directores, Belén y Vidal, por haber hecho posible la realización de este trabajo. No solo por su labor como directores, que ambos han desempeñado con absoluta entrega y rigor como acostumbran a hacer en todas las tareas de su vida profesional, sino por ser mis mentores, mis guías en el mundo universitario, mis maestros y sobre todo mis amigos.

A Javi Blanco por su ayuda en la simulación del movimiento de la carretilla, sin duda unas marcas que unen un origen con un destino, no son comparables con ver moverse a la carretilla aunque sea dibujada con unos cuantos pixeles.

A todos los miembros del Departamento de Informática y Automática por mantener un ambiente de trabajo agradable y cordial, pese a que algunos agoreros llevan tiempo pronosticando "nubes negras". Chicos, ¡no lo vamos a consentir!. En este amplio número de personas que integran el Departamento quiero destacar a aquellos con los que por determinadas circunstancias tengo o he tenido más contacto. A mis "compis" del café, María, Belén, Ana y en especial a Mª José con la que comparto las incidencias e inquietudes del día a día. Al profesor Quintales por "llevarnos de excursión" y por contar conmigo para formar parte de su equipo de gobierno como Director del Departamento. Aunque hubo momentos duros y sin duda este trabajo de tesis se resintió, la experiencia adquirida me ha permitido madurar como persona y conocer de primera mano la gestión Universitaria. Y al resto de "mi quinta": Iván, Fran y Guillermo.

A mi amiga Esther que, aunque la vida nos ha puesto a unos cuantos kilómetros de distancia, la siento muy cercana. A Lola, por saber escuchar, preocuparse por mi e interesarse siempre en el desarrollo de este reto que para mi ha supuesto la realización de la tesis doctoral. A mi familia salmantina, los González Pro, en especial a Marce por estar siempre ahí y soportarme después de tantos años.

A los integrantes de la peña "Coñote" por todos los buenos ratos pasado en el bar que permiten aliviar todo tipo de tensiones; aunque nunca seamos millonarios. En especial a su "alma mater": Vicente, siempre atento a que los momentos de ocio lo sean de verdad.

Y finalmente a mi familia; a mis padres que han volcado su vida para darle todo a sus hijos; a mi hermano que es al que tengo más cerca y a mi hermana y a Jaime que han traído la alegría a la familia con esos "locos bajitos": Carlos y Judith.

Contenido

1. Del manipulador industrial al robot móvil autónomo **1**

1.1. Motivación y objetivos 5

1.2. Estructura de la tesis 7

2. Fundamentos y antecedentes **9**

2.1. Robots autónomos 9

2.2. Planificación de caminos y el espacio de las configuraciones 11

2.3. Cálculo de los C-obstáculos 13

2.4. Planificación de manipulaciones 15

2.5. Generador de trayectorias 17

2.6. Agarres de piezas 19

2.6.1. Análisis de agarres 21

2.6.2. Medidas de calidad de agarres 25

2.6.3. Síntesis de agarres 28

2.6.4. Taxonomías de agarres 31

3. Síntesis de agarres **37**

3.1. Introducción 37

3.2. Formalismo propuesto 39

3.3. Robot rígido con zonas de agarre 42

3.4. Robot rígido con pinza de dos dedos paralelos 46

3.5. Robot con pinza de dos dedos paralelos que se desplaza y gira 49

3.6. Aportaciones 51

4. Propuesta de algoritmos de agarre, criterios de calidad y accesibilidad **53**

4.1. Discretización de los espacios W y C 54

4.1.1. Discretización de las funciones A, E y AE 54

4.1.2. Discretización de las funciones CE y ACE . . . 55
4.2. Algoritmo básico: pinza con apertura fija . . . 56
4.2.1. Funciones discretizadas y algoritmo . . . 56
4.2.2. Resultados del algoritmo básico . . . 58
4.2.3. Independencia de la forma de la pieza . . . 61
4.2.4. Independencia con la geometría de la pinza . . . 63
4.2.5. Criterio de calidad . . . 65
4.3. Algoritmo con criterios de calidad orientados a la tarea . . . 68
4.3.1. Objeto con zonas de agarre . . . 69
4.3.2. Pinza con zonas de agarre . . . 71
4.4. Algoritmo para una pinza que gira . . . 74
4.4.1. Funciones discretizadas y algoritmo . . . 74
4.4.2. Resultados de validación del algoritmo . . . 79
4.5. Algoritmo para un robot con pinza de apertura variable . . . 83
4.5.1. Funciones discretizadas y algoritmo . . . 83
4.5.2. Resultados de validación del algoritmo . . . 84
4.5.3. Criterio de calidad orientado a la tarea . . . 87
4.6. Accesibilidad: cálculo del C-espacio libre . . . 87
4.7. Análisis de la complejidad de los algoritmos propuestos . . . 96
4.8. Aportaciones . . . 97

5. Planificación de manipulaciones de un robot paletizador 99
5.1. Los vehículos guiados automáticamente . . . 99
5.1.1. Planteamiento propuesto para la planificación de manipulaciones . . . 100
5.2. El robot paletizador y su entorno . . . 102
5.3. Síntesis de agarre para un AGV . . . 103
5.3.1. Escenario básico con criterios de calidad en la horquilla . . . 103
5.3.2. Escenario básico con criterios de calidad orientados a la tarea . . . 105
5.3.3. Accesibilidad: cálculo del C-espacio libre . . . 109
5.4. Planificación de trayectorias . . . 114
5.4.1. Planificación de trayectorias en sistemas no holónomos . . . 114
5.4.2. Caminos más cortos para un robot tipo coche . . . 116
5.4.3. Algoritmo propuesto para el cálculo de las curvas de Dubins . . . 118
5.4.4. Cálculo geométrico de las curvas de Dubins . . . 122
5.4.5. Validación de los cálculos geométricos de la curva LSR . . . 127
5.5. Planificación de manipulaciones . . . 129

5.5.1. Algoritmo de manipulación propuesto 129
5.5.2. Resultados de validación del algoritmo 132
5.6. Aportaciones 139

6. Conclusiones y trabajos futuros **145**

Apéndices **149**

A. Algoritmos detallados para cálculo de las curvas de Dubins **151**

Índice de algoritmos

4.1. Algoritmo básico de obtención de agarres 58
4.2. Obtención de agarres con criterios de calidad 69
4.3. Obtención de agarres considerando la orientación 78
4.4. Obtención de agarres variando la orientación y apertura de la pinza 85
4.5. Algoritmo básico de obtención de agarres con FFT 98

5.1. Cálculo de las curvas de Dubins 120
5.2. Algoritmo de manipulación 130

A.1. Algoritmo detallado para la obtención de las curvas de Dubins 152
A.2. Cálculo de la curva de Dubins LSL (figura A.1) 153
A.3. Cálculo de la curva de Dubins RSR (figura A.2) 155
A.4. Cálculo de la curva de Dubins RSL (figura A.3) 156
A.5. Cálculo de la curva de Dubins LSR (figura A.4) 158

Índice de figuras

1.1. Robot KUKA KR 30 HA. 2
1.2. Robot paletizador P6 de la marca PMI 3
1.3. AGV Transcar de la empresa Swisslog-Telelift 4
1.4. Sistema quirúrgico Da Vinci de la empresa Intuitive Surgical 4
1.5. Robot de servicio Care-O-bot 3 5

2.1. Problema del *piano mover's*. Robot con forma de "L". 11
2.2. Entorno de trabajo y obstáculos [LaV06]. 12
2.3. Espacio de las configuraciones del entorno de la figura 2.2. 13
2.4. Entorno de trabajo y obstáculos. 14
2.5. Mano Barret. 19
2.6. Modelado de un agarre mediante un conjunto de fuerzas 22
2.7. No se cumple la condición de *force/torque closure*. 23
2.8. Se cumple la condición de *force closure* en un agarre con dos dedos. 24
2.9. Taxonomía de los agarres humanos [CW86] 32
2.10. Tipos de oposiciones básicas en los agarres humanos de Iberall: (a) Oposición de puntas. (b) Oposición de palma. (c) Oposición lateral. 33
2.11. Agarres de Lyons: (a) Agarre envolvente. (b) Agarre lateral. (c) Agarre de precisión. 34

3.1. Robot rígido A y entorno con objetos movibles M_j y obstáculos B_k. 42
3.2. Elección de los sistemas de referencia. 43
3.3. Robot rígido con pinzas paralelas. 47
3.4. Robot rígido con pinzas paralelas que se desplaza y gira libremente. 50

4.1. Sistemas de referencia para una pinza paralela. 57

4.2. Escenario básico con una pinza de dedos paralelos y un objeto (una tuerca) a manipular. . . . 58
4.3. Discretización de las funciones para el escenario básico con resolución $N = 128$. 59
4.4. Tuerca proyectada en el C-espacio. En z se muestra el logaritmo de la convolución, para mayor claridad. q_1 es la configuración de agarre. . . . 59
4.5. Perfil de la frontera de agarre. . . . 60
4.6. Pinza en la configuración de agarre con dos orientaciones. . . . 61
4.7. Pieza a agarrar con forma de "L" . . . 61
4.8. Configuración de agarre para pieza con forma de "L" . . . 62
4.9. Pieza con forma de "L" en C y su frontera de agarre para pinza con orientación $\theta = -\frac{\pi}{2}$ . . . 62
4.10. Configuración de agarre para pieza en "L" para pinza con orientación $\theta = -\frac{\pi}{2}$. 63
4.11. Discretización del escenario como matrices de $64x64$. . . . 63
4.12. Proyección en la pieza en C para el escenario de la figura 4.11. El eje z está en escala logaritmica. q_1 es la configuración de agarre. . . . 64
4.13. Perfil de la frontera de agarre. La configuración de agarre se alcanza en $q_1(x, y) = (21, 32)$ con el valor máximo de 18. . . . 64
4.14. Pinza en la configuración del mejor agarre $q_1(x, y, \partial_g) = (21, 32, 18)$. . . . 65
4.15. Pinza en otras configuraciones de agarre. . . . 66
4.16. Perfil de la frontera de agarre especificando zona de agarre en dedos y palma. 67
4.17. Escenario con zona de agarre marcada en color rojo en pinza y objeto movible. 69
4.18. Escenario discretizado. . . . 70
4.19. Pieza barra vertical proyectada en C. . . . 70
4.20. Perfil de la frontera de agarre con máximos locales seleccionado con umbral u. 71
4.21. Mejores agarres con zona de agarre en pinza y objeto. . . . 72
4.22. Escenario con zonas de agarre en la yema de los dedos de la pinza y en el objeto. . . . 73
4.23. Perfil de la frontera de agarre. . . . 74
4.24. Mejores agarres con zona de agarre en el objeto y en la yema de los dedos de la pinza. . . . 75
4.25. Pinza paralela A con zonas de agarre y un objeto movible M_1. . . . 76
4.26. Pinza y pieza discretizadas del escenario elegido. . . . 79
4.27. Valor máximo de la frontera de agarre para todas las orientaciones. . . . 79
4.28. Mejores agarres considerando la orientación. . . . 80
4.29. Valores de la frontera de agarre para la orientación $\theta = \frac{\pi}{2}$. . . . 81

4.30. Valores máximos de la frontera de agarre en todas las orientaciones. 81
4.31. Frontera de agarre en 2D en las orientaciones donde se obtienen las configuraciones de los mejores agarres. 82
4.32. Pinza agarrando la pieza en L. 82
4.33. Robot móvil con pinzas paralelas. 83
4.34. Robot con pinza con zona de agarres en la yema de los dedos y en la pieza. . 86
4.35. Robot con pinza de dedos paralelos en diferentes aperturas. 86
4.36. Robot en configuraciones de agarre (continua). 88
4.37. Agarrando por el mango. 89
4.38. Robot con zonas de agarre en un escenario con objetos movibles y obstáculos. 90
4.39. C-espacio en la orientación $\theta = -\pi$. 91
4.40. C-espacio en la orientación $\theta = -\frac{\pi}{2}$. 91
4.41. C-espacio en la orientación $\theta = 0$. 92
4.42. C-espacio en la orientación $\theta = \frac{\pi}{2}$. 92
4.43. Valores máximos de la frontera de agarre en todas las orientaciones. 93
4.44. Frontera de agarre en 2D del escenario de la figura 4.38 en las orientaciones donde se obtienen las configuraciones de los mejores agarres. 94
4.45. Escenario con el robot en las configuraciones de agarre. 95

5.1. Posible entorno de trabajo de un vehículo guiado automáticamente (AGV). . 101
5.2. Robot paletizador y palé. 104
5.3. Las matrices del robot A^* y del palé europeo ${M_1}^*$. 104
5.4. Valores máximos de la frontera de agarre para todas las orientaciones. 105
5.5. Perfil de la frontera de agarre en las orientaciones 0 y $\frac{\pi}{2}$. 106
5.6. Robot en las configuraciones de agarre cuando se especifica únicamente la zona de agarre en su horquilla. 107
5.7. Robot en las configuraciones de agarre con zona de agarre en la horquilla del robot y en el los tacos centrales del palé. 108
5.8. Frontera de agarre en todas las orientaciones. 109
5.9. Perfil de la frontera de agarre en las orientaciones 0 y $\frac{\pi}{2}$. 110
5.10. Escenario de trabajo del robot paletizador. 111
5.11. Escenario E^* y escenario aumentado AE^* con obstáculos y un objeto movible. 111
5.12. Robot en las configuraciones de agarre en un escenario con obstáculos y con objetos movibles. 112
5.13. Frontera de agarre en las orientaciones 0, $-\pi$ y $\frac{\pi}{2}$ donde se producen los agarres y en $-\frac{\pi}{2}$ donde no es posible agarrar el objeto. 113

5.14. El coche sencillo. . . . 114
5.15. Trayectorias de dos de las curvas de Dubins en $W = R^2$ . . . 118
5.16. Cálculo de las curvas de Dubins. . . . 121
5.17. Circunferencias con sentido positivo y negativo en la configuración inicial y final. . . . 121
5.18. Curva de Dubins LSL. . . . 122
5.19. Curva de Dubins RSR. . . . 123
5.20. Curva de Dubins RSL. . . . 123
5.21. Curva de Dubins LSR. . . . 124
5.22. Esquema del cálculo de la curva de Dubins LSR . . . 128
5.23. Escenario con un objeto movible y seis objetos fijos. . . . 133
5.24. Rutas del primer grupo de configuraciones de agarre en el orden en el que han sido evaluadas. El camino de tránsito seleccionado por el planificador es el representado en (a). . . . 135
5.25. Rutas para el segundo grupo de configuraciones de agarre. . . . 136
5.26. Rutas para la última configuración de agarre. . . . 137
5.27. Robot y Escenario para los caminos de transferencia. . . . 138
5.28. Caminos de transferencia desde el primer grupo de agarres en el orden en el que han sido evaluados. . . . 140
5.29. Caminos de transferencia desde el segundo grupo de agarres. . . . 141
5.30. Caminos de transferencia desde el último grupo de agarres. . . . 142

A.1. Esquema del cálculo de la curva de Dubins LSL. . . . 154
A.2. Esquema del cálculo de la curva de Dubins RSR. . . . 154
A.3. Esquema del cálculo de la curva de Dubins RSL. . . . 157
A.4. Esquema del cálculo de la curva de Dubins LSR. . . . 157

Índice de tablas

2.1. Mínimo número de contactos necesarios para obtener agarres FC ([Ngu86a], [Ngu87], [Ngu88]) 24
2.2. Condiciones para la existencia de un agarre FC 25
2.3. Medidas de calidad de agarres 26

4.1. Las 12 configuraciones de agarre con máximo local de la frontera de agarre. . 67
4.2. Configuraciones de agarre ordenadas de mayor a menor calidad con zonas de agarre en la yema de los dedos de la pinza y en el objeto. 73
4.3. Configuraciones de agarre ordenadas de mayor a menor calidad. 86

5.1. Las tres primitivas de movimiento con las que construir las curvas óptimas para el coche de Dubins 117
5.2. Rutas factibles (marcadas en rojo) para agarrar el palé europeo en el escenario de la figura 5.23 siendo $q_I = (-8.0, -8.0, \frac{\pi}{2})$. Las rutas marcadas en rojo no son realizables porque colisionan con un obstáculo en la configuración que se muestra en la celda. 133
5.3. Rutas factibles (marcadas en rojo) para trasladar el palé a la configuración $q_G = (6.0, 2.5, \frac{\pi}{2})$. Las rutas marcadas en rojo no son realizable porque colisionan con un obstáculo en la configuración mostrada. 138

1

Del manipulador industrial al robot móvil autónomo

Una de las ambiciones más importantes en robótica es la construcción de robots capaces de ayudar al hombre en casi cualquier tarea. Estos robots son generalmente concebidos imitando los comportamientos de los seres humanos y tratando de conseguir de forma artificial lo que el ser humano hace de manera absolutamente natural. Sin embargo, este concepto idealizado no concuerda con la definición de un robot como una máquina repetitiva en vez de un sistema versátil e inteligente. La definición adoptada por el Instituto Norteamericano de Robótica (RIA - *Robotic Industries Association*)[1] y aceptada internacionalmente es: *manipulador multifuncional y reprogramable, diseñado para mover materiales, piezas, herramientas o dispositivos especiales, mediante movimientos programados y variables que permiten llevar a cabo diversas tareas.* Esta definición del RIA, del año 1980, refleja el concepto de robot manipulador en una línea de montaje. Sin embargo, la robótica se ha expandido durante años en diferentes direcciones: la inclusión de plataformas móviles, la consideración del sec-

[1] http://www.robotics.org/

tor servicios al mismo nivel que el sector manufacturero y la interacción del hombre con la máquina. A continuación, analizamos el porqué de esta evolución de robot manipulador industrial a robot móvil autónomo.

A principios de los años sesenta se introducen en la industria, de modo significativo, los robots manipuladores como un elemento más del proceso productivo. Esta proliferación, motivada por la amplia gama de posibilidades que ofrecía, suscitó el interés de los investigadores para lograr manipuladores más rápidos, precisos y fáciles de programar. La consecuencia directa de este avance originó un nuevo paso en la automatización industrial, que flexibilizó la producción con el nacimiento de la noción de *célula de fabricación robotizada*. Los trabajos desarrollados por los robots manipuladores consistían frecuentemente en tareas repetitivas, como soldadura, pintura, etc. Para realizarlas, disponen de herramientas de proceso específicamente diseñadas para la tarea (pistolas de soldadura y pintura, taladros, etc.), que se sujetan mecánicamente en el extremo final del robot, por lo que el robot no tiene que cogerla si no que es parte de él. Un ejemplo se puede ver en el robot soldador de la figura 1.1.

Figura 1.1: Robot KUKA KR 30 HA[2].

Este trabajo de tesis está centrado en robots con un mayor grado de versatilidad que realizan tareas como coger-y-colocar (*pick-and-place*) piezas o coger una herramienta para realizar una tarea, para lo que necesitan disponer de elementos de agarre más genéricos.

En tareas de *pick-and-place*, para alimentar máquinas que componen la célula de fabricación robotizada, se utilizan efectores finales sencillos como los de tipo ventosa, pero sólo son adecuados para coger piezas muy ligeras. También aparecen garras expresamente diseñadas para coger un tipo determinado de piezas o herramientas y la forma de agarrar se programa también específicamente para la tarea a realizar. En la figura 1.2 podemos ver un robot paletizador manipulando sacos con una pinza formada por varias garras.

No obstante, el entorno está muy estructurado y cada robot sólo coge un tipo de pieza que siempre se encuentra en una posición y orientación conocidas. El manipulador industrial realiza de forma repetida la misma tarea hasta que es reprogramado para manipular otras piezas o en otras posiciones. Esta reprogramación la realizaba un operador basándose, ini-

[2] http://www.kuka-robotics.com
[3] http://www.tmipal.com

cialmente, en su experiencia, y posteriormente, contó con la ayuda de software para obtener automáticamente los puntos de contacto [SLG+99]. En ambos casos, como la reprogramación se realizaba *off-line*, aparece una limitación a la hora de cambiar la pieza o herramienta que debe coger, pues conduce a paradas en el proceso de fabricación. Una solución es diseñar elementos finales de propósito más general que sean capaces de manipular una variedad más amplia de objetos. Sin embargo, esta flexibilidad del efector final se logra con un mayor número de grados de libertad que, sin embargo, aumenta la complejidad en el software de ayuda.

Figura 1.2: Robot paletizador P6 de la marca PMI [3].

Cuando el robot coge y deja piezas para la alimentación de las distintas máquinas de la célula, éstas deben de estar ubicadas en el interior de un área accesible para el manipulador, caracterizada por la máxima extensión de sus articulaciones, lo cual podría resultar imposible a medida que la célula sufría progresivas ampliaciones. Una solución a este problema se logra al desarrollar un vehículo de guiado automático (AGV - *Automatic Guided Vehicle*) para proporcionar un transporte eficaz de los materiales entre las distintas zonas de la cadena de producción. En un intento de lograr una mayor flexibilidad, los sistemas de guiado han ido evolucionando desde los raíles, a hilos enterrados, las marcas en el suelo, hasta los radioguiados que utilizan *encoders* para conocer la trayectoria recorrida. En este último caso, si se necesita una alta precisión en su posicionamiento para realizar la tarea, los AGVs incorporan sus propios sistemas de ajuste de posicionamiento mediante elementos mecánicos, cámaras de visión y dispositivos láser para detección de balizas. Además disponen de sensores de rango o de contacto para la detección de obstáculos dentro de su sistema de seguridad. Es muy común que el vehículo se detenga hasta que el obstáculo desaparece, pues no disponen de la capacidad de evitar colisiones. En la figura 1.3 tenemos un ejemplo comercializado por la empresa Swisslog-Telelift de un AGV con detección de obstáculos por láser que puede ser utilizado por ejemplo en un hospital para el transporte de material médico, historiales, carros de comidas, etc.

La posibilidad de estructurar el entorno industrial permite la navegación del vehículo y la manipulación de objetos con una capacidad sensorial y de razonamiento mínimas. De

[4] http://www.swisslog.com/

este modo, la tarea se desglosa en una secuencia de acciones en la que a su término el robot supone que ha alcanzado el objetivo para el que está programado. Ante cualquier cambio inesperado en el área de trabajo que afecte el desarrollo normal de la tarea, el sistema de navegación y manipulación se encontrará imposibilitado para ejecutar acciones alternativas que le permitan reanudar su labor.

Figura 1.3: AGV Transcar de la empresa Swisslog-Telelift [4].

Actualmente, las aplicaciones donde el robot trabaja en entornos no estructurados y realizan operaciones de agarre de piezas y herramientas sin restricciones en su forma y características, se llevan a cabo mediante la teleoperación. Esta solución aparece como la más idónea en trabajos cuya complejidad implica disponer de la destreza de un trabajador humano. De hecho sólo tiene sentido sustituir a las personas porque el entorno sea muy adverso o peligroso para su salud, como en desactivación de explosivos, en instalaciones nucleares, etc., e incluso inaccesible, como en intervenciones quirúrgicas, donde ya son una realidad comercial. En la figura 1.4 vemos un ejemplo de este tipo de sistemas, el sistema quirúrgico Da Vinci comercializado por la empresa Intuitive Surgical.

Figura 1.4: Sistema quirúrgico Da Vinci de la empresa Intuitive Surgical [5].

[5] http://www.intuitivesurgical.com/

Además, por las potenciales aplicaciones fuera del ámbito industrial (robots de servicio), donde resulta costoso o imposible estructurar el entorno, se les intenta dotar, en la búsqueda de un vehículo manipulador de propósito general apto para desenvolverse en cualquier clase de ambiente, de un mayor grado de inteligencia y percepción, además de efectores finales más complejos. Según la Federación Internacional de Robótica (IFR - *Internacional Federation of Robotics*), la definición de un robot de servicio es: *un robot que opera de forma parcial o totalmente autónoma, para realizar servicios útiles para el bienestar de los humanos y del equipamiento, excluyendo operaciones de manufactura.* En esta definición se plantea la capacidad de movimiento y manipulación sobre entornos no estructurados, de los que se posee un conocimiento incierto, mediante la interpretación de la información suministrada a través de sus sensores y del estado actual del vehículo. Esta capacidad lleva implícita distintos niveles de autonomía: desde robots únicamente teleoperados a robots complemente autónomos, incluyendo niveles intermedios. La empresa alemana Care-O-bot comercializa el robot de servicio de la figura 1.5 capaz de moverse entre los humanos, detectar y agarrar los objetos mas frecuentes que existen en el hogar e intercambiarlos de forma segura con los humanos. En [FP05] se puede encontrar un estudio sobre la robótica de servicio.

Figura 1.5: Robot de servicio Care-O-bot 3 [6].

1.1. Motivación y objetivos

El trabajo de investigación que se presenta en esta tesis doctoral se enmarca dentro de una de las líneas de investigación más importantes del grupo de investigación Robótica y

[6] http://www.care-o-bot.de

Sociedad de la Universidad de Salamanca como es la búsqueda de métodos que permitan obtener la representación y evaluación de los obstáculos en el espacio de las configuraciones de forma rápida y con un consumo moderado de recursos.

Estos trabajos fueron iniciados en 1998 por Curto en [CVM98] donde se presenta un formalismo matemático que permite representar los obstáculos en el espacio de las configuraciones tanto para robots móviles como para los robots articulados. En este trabajo Curto recoge la propuesta inicial de [Kav95] solo para robots móviles, donde se parte del hecho de que cuando un robot es un objeto rígido que sólo se puede mover de forma translacional, el espacio de las configuraciones es la convolución del espacio de trabajo y del robot. Para una evaluación rápida de esta convolución se hace uso del algoritmo para la Transformada Rápida de Fourier (*Fast Fourier Transform*, FFT). Este método generalizado por Curto para robots articulados y móviles es especialmente importante para espacios de trabajo con muchos y/o complicados obstáculos, o cuando la forma del robot no es simple como sucede, por ejemplo, para un robot PUMA, SCARA, etc., puesto que se puede beneficiar del hardware y la experiencia existentes para el cálculo de la FFT.

Posteriormente en [The02] se propone un nuevo método que permite obtener una representación de los obstáculos en el espacio de las configuraciones para robots articulados redundantes. Los algoritmos propuestos para ser explotados en un entorno paralelo permiten obtener unos mejores resultados reduciendo el tiempo de cálculo y los recursos utilizados. En [Bla03] se aborda el problema de la capacidad de almacenamiento que la representación del C-espacio presenta cuando un robot móvil de se encuentra en un espacio de grandes dimensiones y se propone un método que permite el cálculo del C-espacio representado en estructuras de datos jerárquicas.

Es en este contexto donde surge este trabajo de investigación partiendo de la observación de que la proyección de los obstáculos en el C-espacio, además de representar todas las configuraciones en las que el robot colisiona o no con estos, también permite obtener información de las zonas de contacto entre el robot y un objeto del entorno. Los puntos de contacto entre ambos se encuentran en la frontera del C-Obstáculo.

Así, el objetivo principal de este trabajo es proponer un nuevo método de síntesis de agarres que permitirá automatizar los procesos de agarre cuando los objetos a agarrar son desconocidos a priori y el entorno no es estructurado. Además, el método debe estar bien fundamentado, de forma teórica, por un formalismo sólido y de carácter general. El método permitirá definir criterios de calidad de agarre como los orientados a alcanzar un alto grado de inmovilización, los orientados a la tarea específica a realizar con el objeto, etc.

Para validar el formalismo propuesto se diseñarán los algoritmos que lo implementen

para diferentes tipos de robot y para distintas piezas a manipular. De forma semejante a los trabajos previos del grupo, se buscará que la complejidad computacional de los algoritmos sea independiente de la geometría tanto de las partes que componen el robot como de los objetos situados en su entorno, sin ninguna contraprestación en cuanto a la exactitud de la representación obtenida.

El método de síntesis de agarres se integrará en un planificador de manipulaciones. El planificador de manipulaciones calculará las configuraciones donde se producirán los mejores agarres, planificará la trayectoria desde la configuración inicial del robot hasta la configuración de agarre (los denominados caminos de tránsito) evitando colisionar con los obstáculos y una vez agarrado el objeto planificará la ruta hasta la posición final deseada considerando que el robot transporta el objeto (caminos de transferencia) y evitando colisionar con ningún obstáculo.

Los resultados del planificador de manipulaciones se han de validar en un entorno no estructurado donde un robot móvil (una carretilla elevadora que está siendo robotizada grupo de investigación Robótica y Sociedad de la Universidad de Salamanca) manipula objetos de forma no preestablecida.

1.2. Estructura de la tesis

Una vez establecidos los puntos de partida en los que se enmarca el presente trabajo se desglosan a continuación los capítulos en los que se ha organizado la memoria.

En el capítulo segundo se estudian los fundamentos que nos permitan comprender las diferentes tareas que han de ser realizadas por un robot móvil autónomo y se analizarán los trabajos mas relevantes en aquellos aspectos directamente relacionados con este trabajo de tesis.

En el capítulo tercero se presenta el formalismo propuesto para la síntesis de agarres. Inicialmente de forma general y posteriormente particularizado para los tipos de robots que se tendrán en cuenta en este trabajo.

Una vez definido el formalismo, en el capítulo cuarto se presentan en dificultad creciente los algoritmos y su aplicación a los diferentes tipos de robots. Para cada uno de ellos se realizará un análisis de su complejidad computacional. Además se mostrará como considerar zonas de agarre en el efector final y criterios de calidad orientados a la tarea que se desea realizar con el objeto agarrado. Demostraremos también que los agarres obtenidos son accesibles; es decir, alcanzables por el robot pues no colisionan con los obstáculos.

En el capítulo quinto se describe como integrar el planificador de agarres en un planifi-

cador de manipulaciones capaz de determinar caminos de transferencia y tránsito para un robot tipo coche y sometido por tanto a restricciones no holónomas. Este capítulo finaliza mostrando los resultados mediante una simulación en Matlab del planificador de manipulaciones cuando nuestro robot paletizador ha de trasladar un palé a una determinada posición, por ejemplo, a una estantería de un almacén.

Para finalizar, en el capítulo sexto se exponen la principales conclusiones a las que ha llevado el desarrollo de este trabajo, así como las líneas de investigación que abren los resultados alcanzados.

2

Fundamentos y antecedentes

En este apartado se explicarán los conceptos fundamentales necesarios para situar este trabajo de tesis doctoral en su contexto. Así, se expondrán primero los subsistemas en que se puede dividir las tareas que proporcionarán al robot autónomo la capacidad de razonamiento como son: el planificador de tareas, el planificador de movimientos junto con el planificador de manipulaciones, y el generador de trayectorias. Para cada uno de estos subsistemas se revisarán las líneas y los trabajos de investigación en ese área directamente relacionados con este trabajo así como las principales aportaciones de cada uno de ellos.

2.1. Robots autónomos

Desde Shakey [Nil69], primer robot capaz de razonar sobre sus propias acciones, y Handey [LPJM+87], capaz de manipular objetos, el desarrollo de la tecnología junto con el desarrollo de técnicas y algoritmos en campos como la visión o en otro tipo de sensores, la inteligencia artificial, etc. han contribuido al desarrollo de los robots autónomos. Un robot autónomo se caracteriza por una conexión inteligente (planificación y razonamiento) entre su percepción

y acción (movimiento y manipulación), que define su comportamiento y le permite llegar a la consecución de los objetivos programados sobre entornos con cierta incertidumbre.

El grado de autonomía depende en gran medida de la facultad del robot para abstraer el entorno y convertir la información obtenida en órdenes, de tal modo que, aplicadas sobre los actuadores del sistema, garantice la realización eficaz de su tarea. En el caso de un robot móvil, las dos grandes capacidades que lo alejan de cualquier otro tipo de vehículo se relacionan a continuación [LP90]:

- Percepción: Determina la relación del robot con su entorno de trabajo, mediante el uso de los sensores del robot, y le permiten localizar objetos de interés.

- Razonamiento: Determina las acciones que se han de realizar en cada momento, según el estado del robot y su entorno, para alcanzar las metas asignadas.

La capacidad de razonamiento del robot autónomo ha impulsado el desarrollo de subsistemas capaces de especificar una tarea en lenguaje de alto nivel, próximo al usuario, y automáticamente compilar esta especificación en un conjunto de primitivas de bajo nivel, o controladores realimentados, para realizar la tarea. Estos subsistemas, que normalmente aparecen interrelacionados de forma jerárquica, son: el planificador de tareas, el planificador de movimientos junto con el planificador de manipulaciones, y el generador de trayectorias [Lat91b].

El planificador de tareas, partiendo de una tarea especificada por el usuario, trata de obtener un conjunto de operaciones elementales de desplazamiento, ensamblado, manipulación, etc. El planificador de movimientos, para realizar cada una de estas operaciones elementales, ha de generar el camino entre el estado inicial y el final sin colisiones del robot con objetos del entorno. El planificador ha de ser eficiente temporalmente y completo, en sentido de que si existe una solución la encuentra en el menor tiempo posible, y si no existe devuelva un fallo.

Cuando el robot debe manipular objetos de forma autónoma se necesita un planificador de manipulaciones que integre la planificación de caminos junto con la dificultad adicional de las operaciones de agarre y reagarre. El resultado son caminos donde el robot se dirige a coger el objeto y otros donde lo transporta.

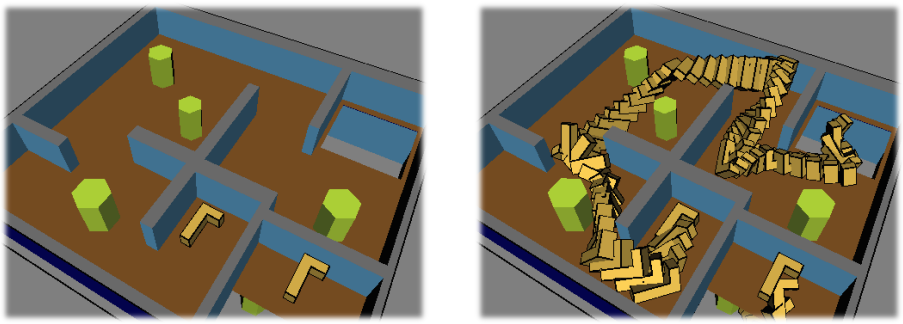
Ambos tipos de caminos son la entrada al generador de trayectoria que determina la dependencia temporal de los parámetros de configuración: la velocidad a lo largo del camino planificado, teniendo en cuenta las restricciones cinemáticas del robot. Los controladores de bajo nivel se encargan de que la trayectoria planificada se ejecute exactamente.

2.2. Planificación de caminos y el espacio de las configuraciones

El clásico problema de planificación de caminos es el problema del *piano mover's*. Dado un cuerpo rígido tridimensional, por ejemplo un poliedro, y un conjunto conocido de obstáculos, el problema es encontrar un camino libre de colisiones para el objeto (un piano) desde un estado inicial a uno objetivo. Se supone que los obstáculos permanecen estacionarios y son perfectamente conocidos, y el camino planificado se ejecuta exactamente. Este enfoque se denomina planificación *offline* o *global* porque, basándose en un modelo del entorno, el planificador construye un plan por adelantado y posteriormente se lo envía al ejecutor del plan.

Esta idea se diferencia de la planificación *on-line* donde se construye un plan mientras el robot se encuentra en ejecución. En este caso el planificador puede estar *basado en sensores* o *local*, donde se abordan las incertidumbres y cambios en el entorno mediante la combinación entremezclada de percepción (*sensing*), planificación y actuación. Un planificador basado en sensores muy simple es el algoritmo tipo *Bug* [LS87b][LS90].

Algunas veces distinguir entre ambas resulta difícil, especialmente cuando el planificador se ejecuta muy rápidamente. En este caso se puede volver a planificar con un nuevo modelo del entorno actualizado por los sensores.



(a) Problema (b) Solución

Figura 2.1: Problema del *piano mover's*. Robot con forma de "L".

En general, un planificador calcula un camino solución que contiene una secuencia de localizaciones, donde el robot no colisiona con los objetos del entorno, y que el robot debe seguir para alcanzar el objetivo. Cuando el robot no es puntual (un manipulador articulado o un móvil), una forma de representar la localización de todos sus puntos es mediante la noción de *configuración* de un robot. El espacio de las configuraciones (*C-espacio*) es el espacio de

todas las configuraciones que el robot puede alcanzar. Un ejemplo de una configuración es el conjunto de todos los ángulos de las articulaciones para un brazo robótico; en el caso de un sofá que se desplaza sobre un plano se tendrían dos variables de posición (por ejemplo (x, y)) y una variable de orientación; si se trata de un piano que puede moverse y girar en 3D entonces serían tres variables de posición (por ejemplo (x, y, z))) y tres de orientación (por ejemplo, *pitch*, *yaw*, *roll*).

La planificación de los movimientos del robot se realiza en el espacio de las configuraciones, donde el robot en una configuración se representa como un punto, pudiendo ser una configuración libre o de colisión. Esto supone una importante simplificación del problema porque supone planificar los movimientos de un punto y no los de todos los puntos del robot.

Además se necesita incorporar un subsistema de *detección de colisiones* que se encarga de validar el estado del robot, es decir, si el robot puede adoptar cierta configuración (estado) sin chocar con los objetos presentes en su espacio de trabajo o con él mismo. En el caso de trabajar en el espacio de las configuraciones, esta validación se reduciría a comprobar si la configuración es libre o de colisión, suponiendo una drástica minimización del tiempo de cálculo asociado a la validación. Sin embargo, es necesario incluir un paso preliminar dedicado a la proyección de los objetos desde el espacio de trabajo (Figura 2.2) al espacio de las configuraciones (Figura 2.3), que se denominan *C-obstáculos*. Este paso solamente es necesario realizarlo una vez para un robot y un entorno de trabajo determinado. Esto supone que para sucesivas ejecuciones de la planificación, por cambios en el estado inicial o final, no se requiere volver a realizar el cálculo.

Figura 2.2: Entorno de trabajo y obstáculos [LaV06].

La completitud es una propiedad muy relevante en los algoritmos de planificación, que hace referencia a que si existe una solución al problema de planificación, el algoritmo la encuentra y sino devuelve un fallo, en un tiempo de cálculo razonable. Basándose en esta propiedad, los métodos de planificación se han agrupado en dos grandes categorías: métodos completos y métodos basados en muestreo.

Los métodos completos cumplen esta propiedad con un bajo coste computacional si el número de grados de libertad es reducido. Entre ellos se encuentran los mapas de carretera, la descomposición en celdas y los campos de potencial, según la clasificación que aparece en [Lat91b].

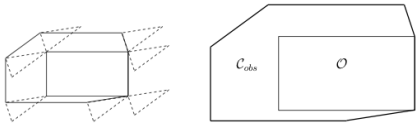


Figura 2.3: Espacio de las configuraciones del entorno de la figura 2.2.

La principal ventaja de los planificadores basados en muestreo (*Sampling-Based Planner*) es que son aplicables en la resolución de problemas con muchos grados de libertad. Entre ellos se encuentran el planificador PRM (*Probabilistic Roadmap*) y el RRT (*Rapidly Exploring Random Trees*).

A partir del problema clásico, del *piano mover's*, la planificación de caminos ha evolucionado para dedicarse a variaciones de este problema, que permiten su aplicación en otras áreas como la animación de actores digitales, verificación automática de diseños industriales, diseño de fármacos, etc. Nuevas aplicaciones conllevan nuevas consideraciones en el diseño de algoritmos de planificación [CLH+05]. El avance en el campo de la planificación de movimientos se ha plasmado recientemente en la biblioteca OMPL (*Open Motion Planning Library*), desarrollada en (`http://ompl.kavrakilab.org/`), que incluye principalmente algoritmos de planificación de caminos basados en muestreo.

2.3. Cálculo de los C-obstáculos

Como se ha comentado, la planificación de caminos global se lleva a cabo en dos pasos ([LP83]):

- *findspace*, donde se encuentran las configuraciones libres de colisión
- *findpath*, donde se calcula una secuencia de configuraciones para que el robot se mueva de una configuración inicial a una objetivo

El primer paso se refiere a la proyección de los objetos del espacio de trabajo W del robot a su espacio de las configuraciones C. A esta representación de los objetos en C se denomina *C-obstáculos* CO, que representan las configuraciones en las que el robot colisiona con los objetos.

La mayoría de los trabajos que calculan explícitamente el C-espacio [LP87] [MF93] [NB91] [Bro89] consideran únicamente un objeto móvil poligonal o poliédrico que se mueve en R^2 o R^3, mientras que un número muy escaso se centra en manipuladores donde sus elementos son polígonos o poliedros. En ambos casos, se considera la presencia de obstáculos poligonales o poliédrico. Esto, además de ser una restricción en las formas de los objetos y elementos, impone que el tiempo de cálculo dependa del número de vértices.

En [LP83] se propone el cálculo de la intersección entre el robot y los obstáculos mediante la expresión

$$CO_A(B) \equiv \{x \in Cspace_A / (A)_x \cap B \neq 0\}$$

donde A es el robot, y B es el obstáculo y $CO_A(B)$ es el C-obstáculo (el obstáculo representado en el C-espacio).

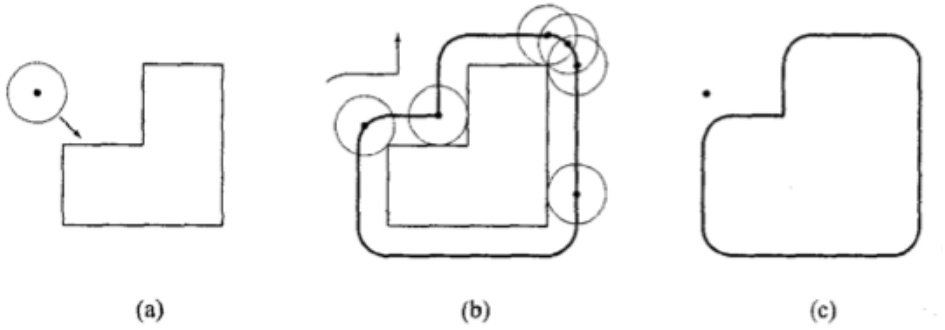


Figura 2.4: Entorno de trabajo y obstáculos.

En la figura 2.4 (a) aparece un robot circular A y un obstáculo B en el espacio de trabajo. En (c) aparece el obstáculo representado en el espacio de las configuraciones y el robot representado como un punto en C.

En [GRS83] se establece que esta operación se puede ver como la convolución de A con B. Esta idea está reflejada en la figura 2.4 (b), donde el robot circular se desplaza alrededor del obstáculo, y registrando la curva trazada por el punto de referencia (situado en el centro del robot) se obtiene el C-obstáculo.

En [Kav95] se propone realizar esta operación a través del Teorema de Convolución. Utilizando bitmaps del robot y del entorno de trabajo se puede obtener un bitmap del C-espacio, lo que permite que el cálculo sea independiente de la forma del robot y de los obstáculos, y por tanto el tiempo de cálculo no depende del número de vértices, sino únicamente de la discretización del espacio. La idea del cálculo mediante convolución es prácticamente intuitiva para robots móviles, aunque no así para manipuladores articulados.

En [CM97] se propone un formalismo matemático para el cálculo del C-espacio por medio de la convolución de una función que representa al robot y otra al espacio de trabajo. Este formalismo es aplicado a robots manipuladores en R^2 y R^3 con 2 y 3 grados de libertad. En [CMB02] se extiende el método para considerar manipuladores redundantes.

2.4. Planificación de manipulaciones

Entre las habilidades que necesita un robot autónomo para interaccionar con su entorno destaca la manipulación de objetos dentro de un entorno donde también existen objetos con los que no tiene que interaccionar. El problema de la planificación de manipulaciones es una forma sofisticada [Lat91b] de la planificación de movimientos donde el robot puede mover objetos (denominados *objetos movibles*). Se puede definir como: *planificar los movimientos de un robot para que pueda mover un objeto determinado desde una configuración inicial a una final mientras evita colisiones con los objetos estáticos y otros objetos movibles en su entorno, con operaciones de reagarre* [AGM98].

La investigación en planificación de movimientos y en planificación de manipulación ha tenido una evolución similar: desde los primeros métodos completos y exactos de descomposición en celdas, que en la práctica son sólo adecuados para casos sencillos del problema, hasta los algoritmos basados en muestreo, capaces de tratar con más grados de libertad y entornos más realistas. A continuación se comentarán algunos de los trabajos que reflejan esta evolución [SCLS04].

De la misma forma que el método para resolver el problema del *piano mover's* ([SS83a] [SS83b] [Lat91b]) surgió dentro de la comunidad de geometría computacional, también el problema de la planificación de manipulaciones encontró sus primeras expresiones formales [Wil88] dentro de esta comunidad, también en entornos 2D. Además los primeros trabajos proponían utilizar la descomposición exacta en celdas en espacios de trabajos poligonales. Wilfong propone un algoritmo de descomposición exacta de celdas para el caso particular de un robot poligonal convexo que traslada (no puede girar) un único objeto movible poligonal convexo en un espacio de trabajo 2D poligonal. Se supone un conjunto finito de agarres posibles del objeto movible. El robot puede agarrar un objeto movible haciendo coincidir uno de sus lados con otro del objeto; después el robot y el objeto se mueven juntos como un único objeto rígido hasta que el robot suelta el objeto movible. Un problema similar se considera en [LA88] donde el robot y el objeto son discos en un entorno poligonal.

También hay que resaltar que en el trabajo de Wilfrong ([Wil88]), se encuentra implícita la idea de combinar el espacio de las configuraciones del robot y del objeto movible, que se hizo explícita en trabajos posteriores como [ASL90][ALS95] [Lat91a]. Junto con los algoritmos de planificación, la principal contribución de estos trabajos reside en la formalización de diferentes aspectos del problema de manipulación, comenzando con la definición de espacio: la manipulación se realiza en el espacio de configuración compuesto (CS), que es el producto cartesiano de los C-espacios individuales de los robots (uno o varios) y de los objetos

movibles. En el cierre de la parte libre de este espacio compuesto ($cl(CS_{free})$ (*configuraciones libres*)) se consideran dos importantes subconjuntos: CP, donde los objetos movibles están en posiciones estables, y CG que es la unión de los diferentes CG_j correspondientes a las configuraciones donde cada objeto movible M_j es agarrado por uno o más robots. Un *camino de manipulación* se define como una secuencia alternativa de caminos de *tránsito* y *transferencia*, donde los primeros corresponden con los movimientos del robot sin transportar objetos, mientras que los de transferencia corresponden con movimientos del robot cuando está desplazando objetos. Estos dos caminos pertenecen a subespacios diferentes y el problema es encontrar cuándo cambiar de uno a otro.

El concepto de *grafo de manipulación* se introduce en [ASL90] para el caso de un robot y varios objetos movibles, donde el número de agarres CG_j y de posiciones estables CP de cada objeto son finitos. En un grafo de manipulación, los nodos se corresponden con configuraciones discretas y los arcos se construyen buscando caminos de transferencia (o tránsito) entre nodos que comparten el mismo agarre o posiciones estables del objeto(s) movible(s). Siguiendo esta estructura general, el planteamiento fue implementado para un polígono que se traslada [ASL90] y un manipulador planar de 3 grados de libertad [ALS95]. En [ALS95] se propone también un algoritmo de descomposición exacta en celdas para el caso específico de un robot poligonal que sólo puede trasladarse (o girar) capaz de manipular un polígono movible con un conjunto infinito de agarres.

Los trabajos que hemos comentado previamente sólo son aplicables a problemas sencillos (de pocos grados de libertad) debido sobre todo a su elevada carga computacional que los hace inviables para entornos de más alta dimensionalidad. Escenarios con manipuladores multibrazo o la manipulación de objetos tridimensionales con múltiples operaciones de reagarre, necesitan de algoritmos más potentes. La aparición de planificadores de caminos basados en muestreo tiene una clara relevancia en la planificación de manipulación. Planificadores como RPP - *Randomized Path Planner*, PRM - *Probabilistic RoadMap* o el algoritmo de Ariadna Clew tienen sus homólogos en el ámbito de la planificación de la manipulación.

Los métodos del tipo PRM ([Kav94], [OS95]) se encuentran en el núcleo de muchos planificadores de manipulación ([NK00], [SCLS04]), debido a que son adecuados para problemas de alta dimensionalidad. En la planificación de caminos se generan al azar configuraciones sin colisión (*landmark*) y se conectan mediante un planificador local. De forma similar, la estrategia en la planificación de manipulación es generar al azar configuraciones en $CP \cap CG$, preferentemente representando a todas las componentes conectadas de esta subvariedad, y conectarlas con los caminos de tránsito y de transferencia.

En [AGM98] se aplica el algoritmo de Ariadne Clew ([BATM93]) a un robot manipulador

redundante (7 grados de libertad) y un único objeto en un espacio 3D. La planificación se realiza en dos niveles siguiendo en ambos una estrategia de exploración y búsqueda. El método supone un conjunto de agarres discretos predefinidos del objeto movible. No obstante, es capaz de tratar en situaciones realistas con manipuladores redundantes, para los cuales cada agarre se corresponda posiblemente con un número infinito de configuraciones del robot.

Para estos nuevos escenarios no se pretende caracterizar de forma explícita y completa el subespacio $CP \cap CG$. En cambio, los algoritmos intentan identificar configuraciones libres de colisión dentro de las componentes conectadas de esta subvariedad. Para evitar este problema, en [NK00] el usuario (o software cliente) tiene que proporcionar un conjunto de posiciones estables y de agarres para el objeto manipulado y conectarlas a través de caminos de tránsito y de transferencia. Tanto para el cálculo del grafo de manipulación como para el planificador punto a punto se utiliza una técnica Fuzzy PRM.

2.5. Generador de trayectorias

Una vez obtenido por los planificadores de caminos y de manipulación un camino libre de obstáculos (realizable) entre dos configuraciones, el camino ha de ser seguido por el robot utilizando una trayectoria generada por el planificador. El generador de trayectorias determina la dependencia temporal de los parámetros de configuración: la velocidad a lo largo del camino planificado, teniendo en cuenta las restricciones cinemáticas del robot.

Los robots autónomos son generalmente vehículos con ruedas que no se pueden desplazar y girar libremente en el espacio de trabajo puesto que están sujetos a restricciones en sus movimientos. Un coche se mueve perpendicular al plano de las ruedas, adelante o atrás, pero no lateralmente con un ángulo de giró limitado por el máximo ángulo de giro de las ruedas directrices. Los sistemas que presentan este tipo de restricciones cinemáticas, que cuando se toman en consideración no reducen la dimensión del espacio de las configuraciones, se conocen como sistemas no holonomos. Los sistemas holónomos son aquellos que al tenerlos en consideración eliminan algunos de los parámetros de las ecuaciones cinemáticas y reducen por tanto la dimensión del espacio de las configuraciones. Un ejemplo de restricciones holónomas son las restricciones que tienen los robots articulados sujetos a movimientos restringidos de sus articulaciones. Las restricciones holonomas no cambian sustancialmente el problema de la planificación de caminos, pero no sucede lo mismo con las no holónomas que añaden mas complejidad al problema.

Así, los sistemas no holónomos se caracterizan por ecuaciones de restricción que contienen

derivadas en el tiempo de las variables de configuración del sistema. Estas ecuaciones no son integrables, las restricciones no se pueden expresar como derivadas en el tiempo de alguna función de las coordenadas generalizadas por lo que son necesarias técnicas de control avanzadas. Esto se produce típicamente cuando el sistema tiene menos variable de control que variables de acción. Por ejemplo, un robot tipo coche tiene dos controles (velocidad lineal y velocidad angular) mientras que se mueve en un espacio de las configuraciones tridimensional (posición en el plano (x, y) y orientación θ). Estas restricciones no eliminan grados de libertad del sólido al que aplican y sin embargo su presencia impone condiciones a su movimiento.

La principal consecuencia de las restricciones no holonómicas es que un camino arbitrario en el espacio de la configuraciones admisible no se corresponde necesariamente con una trayectoria factible o realizable por el robot. Este es básicamente el porqué las técnicas puramente geométricas desarrolladas en planificación de movimientos para sistemas holónomos no son de aplicación directa a los no holónomos.

La planificación de movimientos con restricciones no holónomas es un área de investigación relativamente nueva. Su aplicación a los robots autónomos sigue suscitado el interés de la comunidad científica puesto que la búsqueda de un método general que permita obtener rutas optimas con un coste computacional razonable en sistemas con restricciones no holónomas sigue siendo aún un tema abierto. La mayoría de los modelos cinemáticos de los robots se simplifican para facilitar la tarea de convertir un camino en una trayectoria realizable por el robot.

Una de las líneas de investigación conducentes a simplificar el problema de la planificación de trayectorias son las *maniobras restringidas*. La idea de este método de planificación en escenarios sin obstáculos se basa en la sucesión de trayectorias con ciertas propiedades. Estas trayectorias serán movimientos sujetos a las restricciones de giro del vehículo. Esta idea se inicia con las aportaciones publicadas en [Dub57], [RS90], [Lat91b], [LJTM94], y [FW91] entre otras. En este trabajo de tesis se utilizará uno de estos modelos, el robot de Dubins integrado en un planificador de manipulación para escenarios con obstáculos fijos y obstáculos movibles (objetos manipulables por el robot). [Lau98] constituye una buena referencia para conocer el estado del arte de la planificación y control de movimientos en general y de los sistemas no holónomos en particular.

2.6. Agarres de piezas

Una de las formas primarias que un robot tiene de interaccionar con los objetos de su entorno es agarrándolos. En entornos de fabricación, que generalmente están muy estructurados, cada robot sólo agarra un tipo de objeto o herramienta hasta que es reprogramado para manipular otros. Por tanto, disponen de efectores finales diseñados específicamente para una tarea particular en operaciones de agarre, aunque también existen efectores más flexibles. Para los robots que trabajan en entornos menos estructurados, como los robots de servicio, se diseñan efectores finales de propósito más general y más versátiles que son capaces de agarrar y manipular una variedad más amplia de objetos.

Un efector final versátil [FP05] puede tener una complejidad muy variable: la clásica pinza paralela de dos dedos se puede considerar muy versátil si el recorrido de sus dedos es suficientemente grande para poder agarrar una variedad de objetos; en el extremo contrario, existen dispositivos extremadamente complejos, como las manos antropomórficas, que actualmente se utilizan con fines de investigación o bien trabajan mediante teleoperación. Muy pocos diseños de manos mecánicas (sólo la mano Barret - Figura 2.5) han sido comercializados con fines industriales.

Figura 2.5: Mano Barret[1].

En la sección 2.4, se ha simplificado considerablemente la planificación de manipulaciones, pues se ha definido de antemano un conjunto finito de posibles agarres para cada objeto. Sería muy conveniente disponer de métodos que permitan generar de forma automática este conjunto de posibles agarres, tanto para entornos estructurados como no estructurados. Es claro que para lograr mayor versatilidad de los efectores, desde una clásica pinza paralela hasta una mano artificial, se necesita un mayor número de grados de libertad. Esto supone

[1] http://www.robotnik.es

una complejidad adicional a la hora de decidir cuál es el agarre más apropiado para la tarea. Esta complejidad proviene tanto por la complejidad del efector como por la necesidad de reconocer el tipo de objeto y seleccionar la mejor forma de agarrarlo.

En robótica, el problema de seleccionar un conjunto de contactos sobre un objeto para agarrarlo con un efector (una pinza o una mano artificial) se conoce como *síntesis de agarres*, mientras que el problema recíproco, *análisis de agarres*, comprende el estudio y evaluación de un determinado agarre [BK00]. Un agarre se define como *el conjunto de lugares (puntos o regiones) en la superficie del objeto donde se han de colocar los dedos del efector para agarrarlo*. La definición de un agarre incluirá la configuración del efector, que depende de su cinemática y sus grados de libertad.

La calidad de un agarre viene determinada por múltiples características y, en función de la tarea concreta a realizar, se establecerá una ponderación de las mismas. Siguiendo los trabajos de [Cut84] y [Ngu86b], se resumen las principales características que se suelen imponer a un agarre:

- La *viabilidad* de un agarre se refiere a la existencia de una configuración para los dedos de la pinza capaz de lograr que los extremos de los dedos se sitúen sobre los puntos de contacto.

- La *alcanzabilidad* se refiere a la posibilidad de encontrar un camino libre de colisiones que permita alcanzar la configuración de agarre partiendo de una configuración inicial.

- El *equilibrio* de un agarre indica que la suma de las fuerzas y los pares que actúan sobre la pieza es nula. Esto quiere decir que las fuerzas ejercidas por la pinza a través de los puntos de contacto contrarrestan las debidas a la gravedad (supuesta una situación estática).

- La *estabilidad* de un agarre indica si el objeto volvería a su posición de inicial después de un desplazamiento infinitesimal; esto es, si las fuerzas y pares generados como resultado del desplazamiento se oponen o se suman a las que ocasionaron el mismo. La estabilidad depende de la disposición concreta de los puntos de contacto sobre la superficie del objeto.

Otras características adicionales de un agarre son la *rigidez*, la *resistencia al deslizamiento*, la *manipulabilidad* y la *flexibilidad*, cuya definición se puede encontrar en [FP05].

Se distinguen dos tipos de agarres, en función de maximizar alguna de las características relacionadas previamente: agarre dactilar (*pinch grasp*) cuando el contacto con el objeto se establece únicamente con los extremos o las yemas de los dedos; agarre envolvente (*enveloping*

grasp, *power grasp*) cuando el contacto con el objeto se estable a través de la parte interna de los dedos o la palma de la mano.

Durante las dos últimas décadas, los agarres y la manipulación robótica se ha convertido en un área de investigación muy activa, con trabajos que abordan la problemática crucial de este ámbito y donde se ha establecido una extensa base teórica. En [MLS94] se revisaron las bases teóricas de la manipulación robótica; Mishra [Mis95] estudió los criterios de calidad útiles para seleccionar un buen agarre; Shimoga [Shi96] examinó los algoritmos de síntesis de agarres basados en sus propiedades mecánicas; en [BK00] se describen los problemas relacionados con los puntos de contactos; en [OSC00] se clasifican los problemas relacionados con la manipulación en tres diferentes niveles de control; Bicchi [Bic00] presenta las tendencias en este ámbito, teniendo en cuenta las necesidades funcionales de las manos mecánicas; y en [MA04] han desarrollado un simulador versátil que incluye varios modelos de manos mecánicas y que es útil para evaluar algunos de los conceptos teóricos implicados en agarres.

Por tanto, los trabajos de investigación sobre agarres se han clasificado en las siguientes categorías: el análisis de agarres, las medidas de calidad del agarre y la síntesis de agarres. Por ello, la revisión que se realiza en las siguientes secciones también se ha hecho siguiendo esta clasificación. Sin embargo, la frontera entre las obras relacionadas con estas tres categorías es a menudo muy estrecha y algunas de ellas podrían incluirse en más de una categoría.

2.6.1. Análisis de agarres

El análisis de agarres se refiere al estudio de las propiedades físicas, estáticas y dinámicas, implicadas en un determinado agarre. El objetivo de este análisis es obtener modelos matemáticos de las principales propiedades de un agarre que se utilizarán posteriormente en la síntesis.

En un estudio estático el agarre de una pieza mediante un robot se modela a través de las fuerzas ejercidas por el robot en los contactos con la pieza. El modelado del agarre se plantea respecto a un sólido rígido sobre el que actúan un conjunto de fuerzas y pares. De esta forma el agarre se representa con independencia de la pinza robótica por un conjunto de fuerzas aplicadas sobre la superficie del objeto (en la figura 2.6 se observan sólo fuerzas en dirección normal).

El modelo del contacto va a influir en las fuerzas que el robot va a transmitir a la pieza. Así, se consideran dos factores fundamentales: la dimensión del contacto y el rozamiento existente entre efector y pieza. Respecto a la dimensión, se consideran: contacto puntual, contacto lineal y superficial, según sea un punto, una línea o una superficie la zona de contacto entre el dedo y la pieza. Esto lleva implícito que los dedos puedan ejercer sólo

fuerzas en contacto puntual, y fuerzas y pares en contactos lineal y superficial. Respecto a considerar o no la fricción, implica que las fuerzas ejercidas son perpendiculares a la superficie o están contenidas dentro de un cono de fricción. Combinando ambos factores se pueden tener seis posibles tipos de contacto, cuyas características más destacadas se pueden encontrar en [FP05]. Un trabajo muy relevante y que se puede considerar pionero en esta línea fue realizado por Nguyen ([Ngu86a], [Ngu87], [Ngu88]), quien propuso varios modelos de contactos. Tres de ellos están ampliamente adoptados en la literatura: contacto puntual sin fricción, contacto puntual con fricción (*hard-finger*) y *soft-finger contact*. Para facilitar el análisis de agarres con múltiples puntos de contacto, es habitual descomponer los contactos lineales o superficiales por conjuntos de contactos puntuales, situados en los extremos.

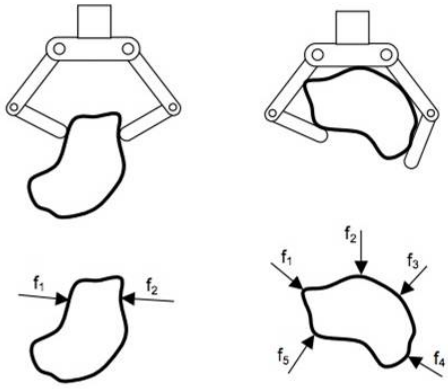


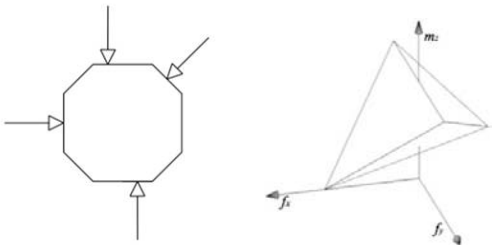
Figura 2.6: Modelado de un agarre mediante un conjunto de fuerzas

Los agarres capaces de resistir perturbaciones externas, es decir, inmovilizar el objeto se clasifican como agarres *form-closure* y *force-closure* [Tri92]. Las diferencias entre los agarres *form-closure* y *force-closure* son muy sutiles y su significado puede variar ligeramente en función del autor. Según las definiciones propuestas en [Tri92] y [Bic95], entre otros, *form-closure* se considera una propiedad puramente geométrica, ya que sólo la situación de los puntos de contacto restringen completamente todos los movimientos (grados de libertad) de la pieza, mientras que *force-closure* tiene en cuenta las fuerzas aplicadas en los puntos de contacto para inmovilizar el objeto. Estos autores establecen que la principal diferencia entre ambos se basa en considerar o no puntos de contacto con o sin fricción en los modelos, siendo el agarre *force-closure* o *form-closure*, respectivamente. Esta diferencia tiene implicaciones en cuanto a las aplicaciones donde se utilizan estos agarres: mientras que los agarres *form-closure* se utilizan para determinar agarres robustos que no se basan en la fricción (por ejemplo, el soporte de objetos para su fabricación o montaje en los procesos industriales), los agarres *force-closure* dependen de la fricción y se utilizan para manipular objetos con

pocos puntos de contacto. En [Bic95] se puede encontrar un amplio estudio de estas dos propiedades. En muchos casos, los aspectos teóricos involucrados en los agarres *form-closure* y *force-closure* son semejantes. Así, cuando las diferencias entre ellos son irrelevantes o el tipo de agarre es evidente por el contexto, se utiliza el término agarre FC para referirse genéricamente a un agarre capaz de resistir perturbaciones externas.

Una de las cuestiones fundamentales en el análisis de agarres es determinar cuándo un conjunto de puntos de contacto produce un agarre FC o no. [Sal82] demostró que una condición necesaria y suficiente para la existencia de un agarre FC es que el conjunto de posibles *wrenches* (es decir, el vector formado por las componentes de la fuerza y del par) producidos por cada una de las fuerzas en los puntos de contacto sobre el objeto recubren positivamente todo el espacio del *wrench*. Entonces, en un agarre FC cualquier *wrench* se puede expresar como una combinación lineal con coeficiente positivos de los *wrenchs* producidos por las fuerzas unitarias de los dedos. En [MSS87] modelaron matemáticamente el conjunto de posibles *wrenches* como una envolvente convexa (*convex hull*), afirmando que la condición necesaria y suficiente propuesta por [Sal82], es equivalente a decir que el *convex hull* contiene el origen del espacio del *wrench*.

En la figura 2.7(a) se muestra un polígono agarrado por cuatro dedos, suponiendo contactos sin fricción. En la figura 2.7(b) se representa el resultado de calcular el recubrimiento convexo definido por el conjunto de los cuatro *wrench* realizados por los dedos. Aparecen las componentes x e y de las fuerzas y el par de rotación. Se puede observar cómo la envolvente convexa (*convex hull*) no contiene al origen de coordenadas, con lo cual se concluye que el agarre propuesto no cumple la condición de *force closure*.



(a) (b)

Figura 2.7: No se cumple la condición de *force/torque closure*.

Para una pinza con dos dedos, que se modela como un agarre con dos contactos puntuales con fricción, se puede demostrar que un agarre cumple la condición de *force closure* cuando la línea que une los dos puntos de contacto se encuentra en el interior de los conos de fricción asociados a cada uno de los contactos. La Figura 2.8 cumple la condición de *force closure*.

Para conseguir mejores agarres hay que elegir el material con el que se recubren los dedos (mayor apertura del cono de fricción) y además que la línea que une los puntos de contacto sea perpendicular a la superficie.

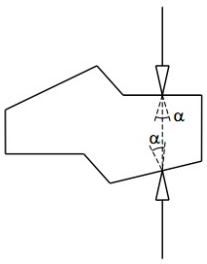


Figura 2.8: Se cumple la condición de *force closure* en un agarre con dos dedos.

En los trabajos ([Ngu86a], [Ngu87], [Ngu88]) Nguyen determinó el número mínimo de puntos de contacto entre el objeto y los dedos que se necesita para obtener un agarre FC, teniendo en cuenta la dimensión del espacio del objeto y el modelo de contacto (Tabla 2.1), y se propuso una condición necesaria y suficiente para cada uno de estos agarres mínimos.

Modelo de contacto	**Espacio de trabajo**	**Mínimo número de contactos**
Contacto puntual sin fricción	2D	4
	3D	7
Punto de contacto con fricción	2D	2
	3D	3
soft-finger contact	3D	2

Tabla 2.1: Mínimo número de contactos necesarios para obtener agarres FC ([Ngu86a], [Ngu87], [Ngu88])

En [MNP90] también se aborda el problema de determinar el número mínimo de puntos de contacto sin fricción necesarios para agarrar un objeto, y demostraron que no hay solución cuando el objeto es una circunferencia o una esfera. Rimon y Burdick [RB96] introdujeron el concepto de agarre *form-closure* y *force-closure* de 2º orden, para tener en cuenta los efectos que la curvatura del objeto y la punta de los dedos producen en un agarre. Basándose en este enfoque, se puede reducir el número mínimo de dedos para inmovilizar un objeto, obteniendo por ejemplo, que sólo son necesarios tres contactos sin fricción para inmovilizar un objeto poligonal.

Siguiendo el planteamiento de Nguyen [Ngu88], varios autores han desarrollado las condiciones específicas para la existencia de un agarre FC, dado el número de contactos y el tipo de objeto. En la Tabla 2.2 se resumen los principales trabajos que han contribuido a definir las condiciones para la existencia de un agarre FC.

Tipo de condición	Espacio del objeto	Número de Contactos	Referencias
Condición necesaria y suficiente	2D/3D	Cualesquiera	[Sal82]
	2D/3D	Cualesquiera	[MSS87]
	2D/3D	Número mínimo dependiendo del modelo de contacto	[Ngu86a], [Ngu87], [Ngu88]
	2D	4 sin fricción	[BM90]
	2D	3 con fricción	[SP05]
	2D/3D	3 con fricción	[LLC03]
Condición suficiente	2D	3 con fricción	[PF91], [PF95]
	3D	3 y 4 con fricción	[PSBM93], [PSS+96]

Tabla 2.2: Condiciones para la existencia de un agarre FC

Otros trabajos en análisis de agarre ([Mon88], [Mon95]) tienen en cuenta la cinemática de los puntos de contacto, describiendo el movimiento de un punto de contacto sobre las superficies de los dos objetos que contactan, y la dinámica del agarre ([HA77], [Ngu89]), respecto a su estabilidad. Estos análisis son fundamentales para controlar los movimientos de deslizamiento o de giro de los dedos sobre la superficie del objeto ([KC92], [MB00]).

2.6.2. Medidas de calidad de agarres

Normalmente, para un determinado objeto, existen varios agarres FC posibles y se utilizan diferentes medidas de calidad para evaluarlos y seleccionar el mejor. En estas medidas de calidad se tienen en cuenta las propiedades de los objetos, los modelos de contacto y las condiciones *form-closure* y *force-closure* para cuantificar la calidad del agarre. En esta sección, se describen y clasifican en cuatro grupos las medidas más conocidas [SCR06], como se resumen en la Tabla 2.3.

El primer grupo de medidas se basa en las propiedades algebraicas de la matriz de agarre G. Esta matriz relaciona las fuerzas aplicadas por los dedos en cada uno de los puntos de contacto con el *wrench* neto, *wrench* formado por los *wrenchs* individuales en cada uno de los puntos de contacto, que actúa sobre el objeto. Entre las propiedades algebraicas de G que se consideran están:

- El valor singular más pequeño de G [LS87a], que indica lo lejos que la configuración de agarre está de caer en una configuración singular.
- El volumen del elipsoide de manipulación [LS87a], para considerar todos los valores singulares de G.
- El número de condición de G [KOYS01], que busca un agarre bien distribuido, por lo que se le denomina *Índice isotrópico de agarre*.

Grupo	Medida de calidad	Referencia
Medidas basadas en las propiedades algebraicas de la matriz de agarre G	Menor valor singular	[LS87a]
	Volumen del elipsoide de manipulación	[LS87a]
	Índice isotrópico de agarre	[KOYS01]
Medidas basadas en las relaciones geométricas de los puntos de contacto	Índice de estabilidad de agarre	[PS90]
	Área del politopo de contacto	[MC94]
	Distancia entre el centroide y el centro de masas	[PF95]
Medidas basadas en la robustez de los errores de posición de los dedos enfrentados	Distancia desde el agarre hasta los límites del espacio FC	[LYT02]
	Longitud de la región de agarre independiente más corta	[Ngu88]
	Índice de incertidumbre de agarre	[KOYS01]
Medidas considerando limitaciones en las fuerzas de los dedos	Bola más grande contenida en la cobertura convexa	[KMY92]
	Wrenches desacoplados	[MC94]
	Bola más grande con respecto a todos los sistemas de referencia	[Tei96]
	Volumen de la cobertura convexa	[MA99]
	Q distancia	[ZDW03]
	Suma de los componentes normales de las fuerzas de los dedos	[LXWL04]
	Distancia de las fuerzas a los límites del cono de fricción	[LXWL04]
	Medidas orientadas a tareas	[LS87a]

Tabla 2.3: Medidas de calidad de agarres

El segundo grupo de medidas de calidad tiene en cuenta determinada relaciones geométricas de los puntos de contacto para evaluar los agarres. Para ello se define el *politopo de contacto*, como el politopo cuyos vértices son los puntos de contacto en la frontera del objeto (así, es un polígono para objetos 2D y un poliedro para objetos 3D). El politopo de contacto se utilizó para establecer las siguientes medidas de calidad:

- La diferencia entre los ángulos internos del politopo de agarre y los del politopo regular correspondiente ([PS90], [KOYS01]). El objetivo de esta medida (denominada *Índice de estabilidad de agarre*) es cuantificar la distribución uniforme de los dedos en la frontera del objeto.

- El área del polítopo de contacto ([MC94], [CFMdP03]) para cuantificar también la distribución uniforme de los dedos sobre el objeto.

- La distancia entre el centro de masas del objeto y el centroide del polígono ([PF95], [CMFdP05]) o poliedro [PSS+96] de contacto, cuyo objetivo es reducir el efecto de las fuerzas gravitacionales e inerciales en el agarre, que se minimiza cuando esta distancia es mínima.

El tercer grupo se basa en la solidez del agarre frente los errores en el posicionamiento de los dedos. Estos se producen porque la posición calculada teóricamente difiere de la posición real de los dedos cuando se lleva a cabo el agarre. [Liu98] define el *espacio de contacto* como el espacio n-dimensional que representa a todos los posibles puntos de contacto de los n dedos en la frontera del objeto. Así, un agarre se representa como un punto en este espacio. En [LYT02] proponen como una medida de calidad, la distancia mínima desde el punto de agarre calculado al límite del espacio FC. También se utilizan distintos índices ([PF95], [KOYS01], [CFMdP03]) basados en distancias, considerando las **regiones de agarre independientes**. Este concepto, introducido por [Ngu88], define a las regiones en la frontera del objeto tales que si cada dedo de la mano se posiciona dentro de una de estas regiones se obtiene un agarre FC, independientemente del punto exacto de contacto dentro de dichas regiones.

Los grupos de medidas de calidad comentados previamente incluyen las relacionadas con la localización geométrica de los puntos de contacto y con la incertidumbre en el posicionamiento de los dedos, pero no consideran ningún límite en la magnitud de las fuerzas aplicadas por los dedos. La condición de *force closure* no indica nada respecto a la magnitud que deberían tener las fuerzas a aplicar sobre los puntos de contacto (las fuerzas a ejercer por los dedos) para contrarrestar las fuerzas o pares (*wrenches*) externos que puedan actuar sobre la pieza (gravedad, inercia, colisiones, etc.). Es posible que un agarre que cumpla la condición de *force closure* sea un mal agarre, dado que precisa aplicar mediante los dedos fuerzas extremadamente altas para compensar pequeñas perturbaciones exteriores. Por esta razón, se utilizan índices adicionales que, en general, proporcionan información relacionada con la máxima fuerza o par externos que el agarre podría soportar, suponiendo que existen determinados límites en las fuerzas que pueden aplicar los dedos. Por tanto, este grupo de medidas está basado en la robustez frente a fuerzas de perturbación externas.

Los límites en las fuerzas aplicadas por los dedos se pueden considerar de muy diversas formas; los criterios más comunes ([Mis95]) son considerar una fuerza máxima para cada uno de los dedos (equivaldría a limitaciones mecánicas de los mismos o una fuente de alimentación individual para cada dedo) o bien un valor máximo para la suma de las fuerzas a ejercer por todos los dedos (equivaldría a limitar la máxima potencia que puede desarrollar la mano o pinza completa).

Si se limita la suma de las fuerzas que deben ejercer los dedos, todos los *wrenches* producidos por los dedos están contenidos en el *convex hull* P_w, denominado *Espacio Wrench de Agarre* ([Pol96], [BFH04]). En [KMY92] y [FC92] proponen como medida de calidad el mayor *wrench* de perturbación que el agarre puede resistir, independientemente de su dirección. Geométricamente, esta medida de calidad es igual al radio de la bola más grande, que

centrada en el origen del espacio del *wrench*, está completamente contenida en P_w. Debido a la interpretación geométrica de esta medida se la denomina *criterio de la bola máxima*.

La suma de las componentes normales de las fuerzas aplicadas en la frontera del objeto es indicativo de las fuerzas internas que soporta el objeto cuando se aplica una perturbación externa. Luego, se define otra medida de calidad como la suma de los módulos de las componentes normales de las fuerzas aplicadas que se necesitan para lograr un *wrench* esperado (que puede ser sólo el peso del objeto). Esta medida se llama *Fuerza de Agarre Normal Máxima* [LXWL04] o *Esfuerzo de Agarre* ([Pol04]) y debe ser minimizada para obtener un agarre óptimo. Como diferencia con la aplicación del criterio de la bola máxima, esta medida fija de antemano el *wrench* externo que se puede resistir y considera entonces las fuerzas necesarias para compensarlo.

Otro enfoque tiene en cuenta que si las fuerzas del dedo aplicadas en cada punto de contacto en ausencia de perturbaciones están próximas a las normales a la frontera del objeto, entonces la fuerza aplicada puede variar en un amplio rango de direcciones para contrarrestar las perturbaciones externas, mientras que si las fuerzas del dedo están inicialmente cerca del límite de los conos de fricción, entonces los dedos podría fácilmente deslizar cuando tratan de mantener el agarre *force-closure*. Este efecto se considera otra medida de calidad de agarres llamada *Centro Analítico Mínimo* [LXWL04].

Cuando se dispone de una descripción detallada de la tarea a realizar, la medida de calidad puede cuantificar la habilidad del agarre para contrarrestar las perturbaciones esperadas durante la ejecución de tarea. La tarea se puede caracterizar por un conjunto de *wrenches* que se deben aplicar sobre el objeto para lograr un objetivo determinado y por un conjunto de *wrenches* debidos a perturbaciones esperadas que el objeto debe soportar mientras se manipula. Todos estos *wrenches* forman un politopo de tarea (denominada *Espacio Wrench de la Tarea* por [BFH04] y [Pol96]), que se suele aproximar con un elipsoide [LS87a]. La medida de calidad es el factor de escala que se necesita para obtener el mayor elipsoide centrado en el origen y plenamente contenido en el *convex hull*.

2.6.3. Síntesis de agarres

La síntesis de agarres se refiere al desarrollo de algoritmos para determinar un agarre apropiado de acuerdo con los criterios de calidad seleccionados. Existen multitud de algoritmos de síntesis de agarres, tanto centrados en el cálculo de los puntos óptimos de contacto como en el cálculo de las fuerzas óptimas a aplicar sobre tales puntos. Para esta tesis serán más relevantes los primeros. Además, puesto que la síntesis de agarres es necesaria en varias ocasiones durante una tarea de manipulación, es conveniente que el coste computacional de

estos algoritmos sea tan bajo como sea posible, a fin de permitir el cálculo del agarre durante la ejecución de la tarea.

En planificación de agarres se ha desarrollado una amplia gama de algoritmos: desde los que proporcionan pruebas cualitativas (es decir, determinar si un conjunto de contactos permite obtener un agarre FC o no), a los que utilizan métodos heurísticos para obtener un buen agarre o técnicas de optimización para obtener el agarre óptimo, en ambos casos basándose en una o varias medidas de calidad.

A partir de la condición general necesaria y suficiente establecida por [MSS87], para garantizar la existencia de un agarre FC, la prueba cualitativa más general para afirmar si una serie de puntos de contacto permite un agarre FC o no, es comprobar si el *convex hull* P_w del conjunto de posibles *wrenches* producidos por las fuerzas de contacto contiene el origen del espacio del *wrench* (en [BDH96] se propone un algoritmo eficiente para calcular el *convex hull*). Para simplificar este método, algunos autores han desarrollado otros test cualitativos. En esta línea, todas las condiciones desarrolladas para agarres FC específicos (véase la tabla 2.2) se pueden también utilizar como test cualitativos. Test cualitativos más generales se desarrollaron en [CB93a], que son válidos para n dedos y objetos en 2D, y en [LW98], que introdujo un algoritmo de disparo de rayos (*ray-shooting*) válido también para n dedos, pero considerando objetos 3D. En [ZQ05] se realizaron algunas mejoras al algoritmo de disparo de rayos.

Los test cualitativos constituyen el núcleo de los algoritmos desarrollados para la búsqueda de agarres FC. Como ejemplos, [SVR03] desarrolló un algoritmo para buscar agarres FC de objetos 2D basado en la condición necesaria y suficiente propuesta por [BM90], y en [LLD04] se desarrollaron algoritmos para la búsqueda de agarres FC de objetos 3D basados en el algoritmo de disparo de rayos.

Otra forma de saber si un conjunto de puntos de contacto produce un agarre FC es mediante el cálculo de todo el espacio FC (es decir, el subespacio del espacio de contacto formado por todos los agarres FC), y, a continuación, comprobar si el agarre pertenece a este subespacio o no. En esta línea, [Liu98] y [LYT02] propusieron métodos para calcular todo el espacio FC para objetos 2D poligonales considerando cualquier número de dedos. En [vdSWO00] y [CvdS05] proponen algoritmos para calcular todos los agarres *form-closure* de objetos poligonales y no poligonales, respectivamente, con cuatro puntos de contacto con fricción.

El **cálculo de los puntos de contacto** mediante test cuantitativos es el principal objetivo de la síntesis de agarres. La forma más básica de agarrar un objeto es con dos puntos de contacto con fricción; en este caso, varios autores ([MiERSdP01] [Jia02] [Jia04])

consideraron que los agarres más estables son los formados por dos puntos anti-podales, es decir, puntos de la frontera del objeto donde las direcciones normales son colineales . En [CB93b] se propone un algoritmo para encontrar los puntos anti-podales más distantes para aumentar la estabilidad del agarre frente a pares externos, y en [SPF92] y [FP91] se introdujo el uso de regiones de agarre independientes en el cálculo de agarres con dos dedos para proporcionar robustez a los agarres frente a los errores de posicionamiento de los dedos.

Entre todos los índices de calidad descritos en la sección previa, los dos más populares son: la bola máxima (y sus variantes) por su robustez frente a fuerzas de perturbación externas; y la longitud de las regiones de agarre independientes por su robustez frente a errores de posicionamiento de los dedos.

El problema de calcular el agarre óptimo basándose en el criterio de la bola máxima fue abordado por [Tri92] quien propuso un algoritmo basado en programación lineal; [Jia95] identificó diferentes casos de agarres óptimos con cuatro puntos de contacto sin fricción; [ZW03] y [LXWL04] propusieron algunas variaciones de este criterio para aplicar algoritmos de búsqueda de gradiente. El principal inconveniente de estos métodos es el coste computacional, lo que implica que estos enfoques han de simplificarse cuando se aplican en sistemas con restricciones temporales [LXWL04]. Debido a la complejidad de los algoritmos de optimización, el criterio de la bola máxima ha sido utilizado frecuentemente para evaluar agarres FC generados con diferentes estrategias heurísticas ([HeKT99], [BFH99], [PGS03], [BFH03], [BFH04], [MKCA03], [MA04], [HSSR05]).

El problema de calcular las máximas regiones de agarre independientes fue abordado por [Ngu88] teniendo en cuenta el número mínimo necesario de dedos para agarrar un objeto. En [PF91], [PF95], [PSBM93] y [PSS+96] se realizó un trabajo muy destacado en el cálculo de las máximas regiones de agarre independientes para objetos poligonales y poliédricos.

[Pol96] desarrolló un algoritmo que combina el criterio de bola máxima y el de las regiones independientes para objetos 2D. Basado en un agarre prototipo (que se calcula fuera de línea), el algoritmo determina las regiones de agarre independientes que permiten agarres FC con un determinado porcentaje de calidad respecto al agarre prototipo. Este enfoque también se ha ampliado a objetos 3D [Pol04]. Sin embargo, la selección de un prototipo adecuado sigue siendo un paso crítico en este algoritmo.

La minimización de la distancia entre el centro de masas del objeto y el centro geométrico de los puntos de agarre también se ha utilizado en algunos algoritmos, como, por ejemplo, los propuestos por [MP89], [DLW00] y [DLW01].

Otros algoritmos a fin de obtener una medida global de calidad combinan varios criterios (por ejemplo, la inclusión de criterios relacionados con la destreza o la estabilidad). En esta

línea, [BSD04], [Kim04], [CFMdP03] y [CMFdP05] proponen algoritmos para calcular el agarre basándose en una medida de calidad global.

La mayoría de los algoritmos de síntesis de agarres parten de un conocimiento preciso de la geometría del objeto a agarrar, ya sea 2D o 3D. Sin embargo, otros trabajos plantean el cálculo de agarres desde una perspectiva más cercana a una aplicación práctica, de modo que la única información disponible sobre el objeto es la adquirida mediante los sensores presentes en el sistema. Así, en [HRSF99] se calculan los agarres a partir de la información del contorno del objeto obtenida mediante un par estéreo. Se establecen criterios distintos para el agarre de un cierto objeto en función de la relación entre la máxima dimensión vertical y horizontal del mismo (se distingue el agarre de objetos tumbados del agarre de objetos levantados). En ambos casos se buscan los puntos de contacto que optimizan ciertas características geométricas prefijadas. Se utiliza una pinza de dos dedos y los puntos de contacto son obtenidos como puntos tridimensionales, gracias a la información estéreo disponible. En [MiESP04] se utiliza también información visual estéreo para obtener el contorno de una pieza. En este caso se utiliza una pinza de tres dedos pero los agarres son bidimensionales (planos). Se consideran únicamente como agarres válidos aquellos en los que las normales a la superficie de la pieza sean confluyentes, por razones de estabilidad. Entre todos los agarres alcanzables por la pinza que cumplen la condición anterior se selecciona el más robusto; tal robustez se predice a partir de experimentos de medida de la capacidad de sujeción del agarre. En [FP05] se propone un sistema para la síntesis de agarres basado en aprendizaje automático. Las reglas que permiten calcular tanto los puntos de agarre como la configuración de la pinza se infieren a partir de los ejemplos proporcionados por el usuario, es decir, el usuario debe agarrar diversos objetos para proporcionar ejemplos de entrenamiento. Para el aprendizaje automático se utilizan los árboles de decisión.

2.6.4. Taxonomías de agarres

Cuando se busca que el robot trabaje en un entorno no estructurado, el diseño de los efectores finales debe de ser muy versátil, como en las manos mecánicas. Sin embargo, esta característica hace que la síntesis de agarres sea muy compleja, por el gran número de grados de libertad que poseen las manos robóticas. Esta versatilidad da lugar a un enorme conjunto de posibles configuraciones de la mano.

En un intento de reducir la complejidad de un agarre se ha estudiado la forma en que los humanos cogen los objetos. Las personas seleccionan inconscientemente una postura apropiada para agarrar un objeto y efectuar con él una determinada tarea. El estudio de cómo estos factores influyen en la elección del agarre conduce a una taxonomía con la cual

proporcionar una forma sistemática de elegir un agarre apropiado para un conjunto concreto de requisitos de la tarea y características del objeto.

La literatura médica ha tratado de clasificar las posturas de las manos humanas en taxonomías de agarres, como la propuesta en 1956 [Nap56]. Napier argumenta que cuando se agarra un objeto para cualquier propósito, la mayor preocupación es la estabilidad del agarre. Así, basándose en cómo se alcanza la estabilidad, establece las diferencias fundamentales entre los agarres enérgicos, que envuelven el objeto y normalmente proporcionan máxima estabilidad, y los agarres de precisión, donde los contactos tienen lugar principalmente en la yema de los dedos y normalmente ofrecen máxima manipulabilidad.

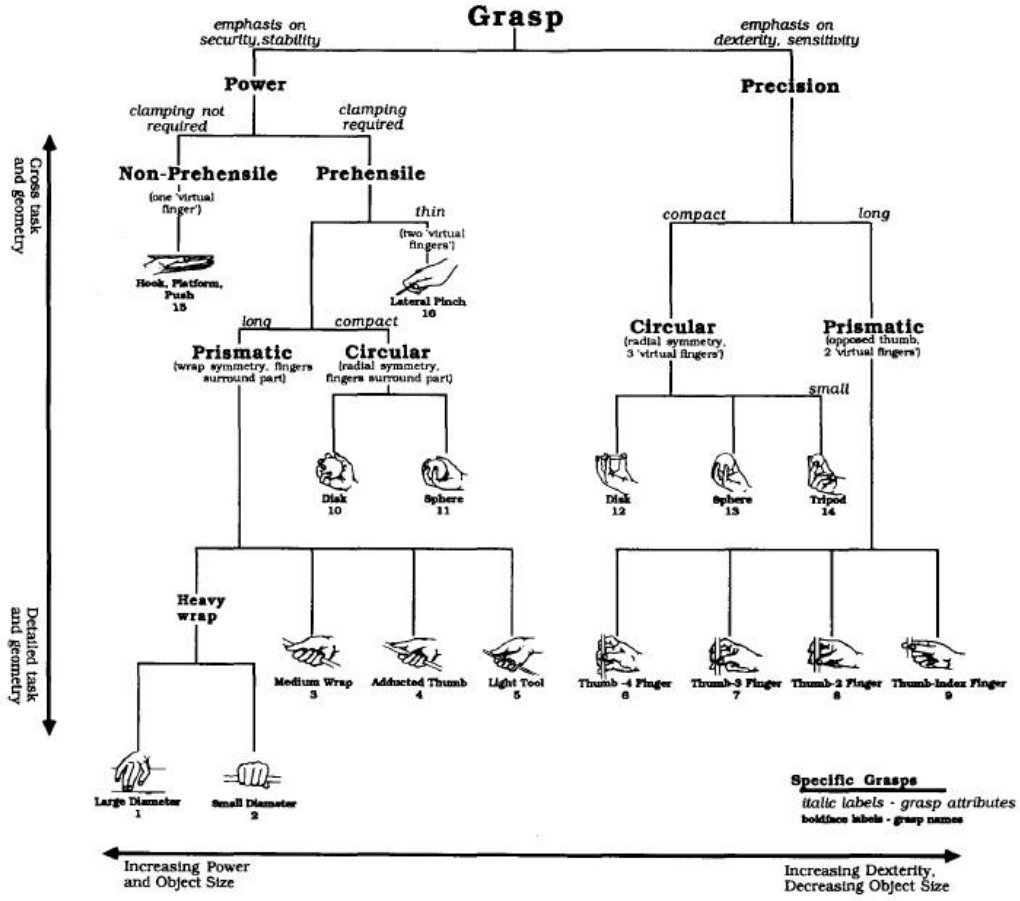


Figura 2.9: Taxonomía de los agarres humanos [CW86]

En [CW86] se extiende esta clasificación a los agarres propios de un entorno de fabricación y se examina la forma en que la tarea y la geometría del objeto afectan a la elección del agarre más adecuado. Esta clasificación en árbol se puede ver en la figura 2.9. De izquierda a derecha en el árbol, los agarres son menos potentes y más diestros, y los objetos agarrados más pequeños. Los agarres envolventes son los más potentes, aunque menos hábiles y, por

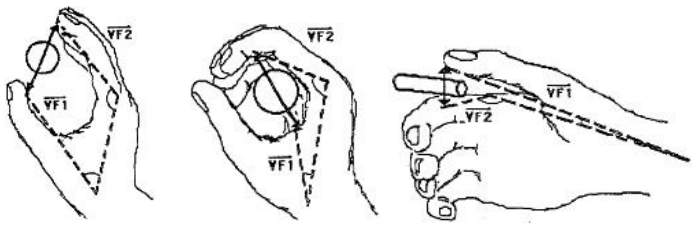
lo tanto, necesitan que las manipulaciones deban realizarse con la muñeca. Por el contrario, los agarres de precisión son mucho más delicados. De arriba a abajo, las opciones van desde agarres orientados a tareas a agarres orientados a objetos. Co mo con cualquier clasificación jerárquica, hay algunas excepciones a estas reglas.

En [Ibe87] y [Ibe97] se propone una taxonomía basada en el concepto de *dedos virtuales*, que tiene en cuenta las restricciones impuestas por la tarea. Un *dedo virtual VF* es un dedo abstracto que representa el mismo efecto mecánico que algunos de los dedos reales. Para seleccionar un agarre se identifican las características de la tarea específica, y, posteriormente, se particulariza el agarre teniendo en cuenta el objeto a manipular. Por lo tanto, el agarre se define en términos de los dedos virtuales, y más tarde se hace corresponder con los dedos físicos. Según Iberall, existen tres formas básicas de colocar los dedos para aplicar fuerzas opuestas sobre un objeto con la finalidad de ejecutar una determinada tarea (Figura 2.10):

1. Oposición de yemas, entre la yema del pulgar (VF1) y de los dedos (VF2). Este agarre proporciona una gran flexibilidad para un ajuste fino de los movimientos, pero poca estabilidad.

2. Oposición de palma, entre la palma (VF1) y los dedos (VF2). Se sacrifica la flexibilidad en favor de la estabilidad.

3. Oposición lateral se produce tanto entre la yema del pulgar (VF1) y el lateral del dedo índice (VF2) como entre los lados de los dedos. Este agarre representa un compromiso entre los dos previos.



(a) (b) (c)

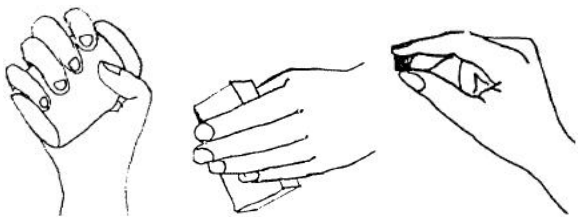
Figura 2.10: Tipos de oposiciones básicas en los agarres humanos de Iberall: (a) Oposición de puntas. (b) Oposición de palma. (c) Oposición lateral.

En [Lyo85] se clasifican los agarres teniendo en cuenta que la mano humana, en el proceso de acercamiento al objeto, *preforma* la posición de los dedos para agarrarlo adecuadamente. Mientras que el acercamiento se refiere a la posición del objeto, el agarre se refiere a las características de los objetos y a la tarea a realizar posteriormente con ellos. Por lo tanto,

se asocia con cada agarre una *configuración de preforma* de los dedos para ese agarre y se restringen los grados de libertad de la mano una vez que el objeto ha sido agarrado. Lyons clasifica los agarres en tres categorías principales, según la precisión y la estabilidad (Figura 2.11).

1. Agarre envolvente. La mano envuelve al objeto. Se trata de un agarre firme sin capacidad de manipulación.
2. Agarre lateral. Las superficies planas de los dedos se utilizan para agarrar el objeto. Proporciona cierta capacidad de manipulación y cierta estabilidad.
3. Agarre de precisión. El objeto se coloca entre los dedos. Se permite el movimiento arbitrario, pero proporciona poca estabilidad.

Cada uno de estos agarres se particulariza para adaptarse al tamaño y la forma del objeto.



(a) (b) (c)

Figura 2.11: Agarres de Lyons: (a) Agarre envolvente. (b) Agarre lateral. (c) Agarre de precisión.

El paradigma de las *preformas* ha sido reconocido como una forma práctica de reducir la complejidad del problema de la síntesis de agarres en efectores finales con muchos grados de libertad. Por ejemplo, la mano de Salisbury y la de Utah/MIT tienen 15 y 22 grados de libertad, respectivamente. Una preforma de la mano es la postura que los dedos adoptan cuando la muñeca se acerca al objeto. El agarre se ejecuta colocando la muñeca en una posición que envuelva el objeto, y luego flexione los dedos (por ejemplo cerrándolos) hasta que contacten con el objeto.

A partir de una clasificación sencilla de tipos de agarres, Stansfield [Sta91] construye un sistema basado en reglas que, tomando como entrada una descripción simplificada de un objeto obtenida desde un subsistema de visión, proporciona un conjunto de posibles preformas de la mano de Salisbury y direcciones alcanzables para la etapa de pre-contacto. Sin embargo, tiene limitaciones en cuanto a que: sólo se examinan cinco posibles direcciones de aproximación, no obtiene el mejor agarre del conjunto de posibles agarres, y para cada agarre elegido los dedos se cierran sin tratar de optimizar la estabilidad del agarre.

Wren and Fisher [WF95] son los primeros en proponer el uso de prototipos predefinidos de agarres (preformas de la mano) para disminuir los grados de libertad internos de la mano. Además de decidir qué preforma utilizar, también consideran las trayectorias de los dedos cuando se flexionan desde la preforma hasta el agarre. Así, definen la *estrategia de agarre* que asocia las trayectorias de los dedos con la preforma.

Pollard [Pol96] desarrolló un algoritmo que, basándose en una preforma que se calcula fuera de línea, determina las regiones independientes que permiten agarres con un determinado porcentaje de calidad respecto a la preforma. Sin embargo, la selección de la preforma sigue siendo un paso crítico en este algoritmo, haciendo que no resulte adecuado para objetos que difieren sustancialmente de los objetos contenidos en una base de datos.

En [MKCA03] se planifican agarres estables de un objeto integrados en su aplicación de simulación de agarres GraspIp [MA00]. Este planificador consta de dos etapas. En la primera, se genera un conjunto inicial de los lugares de agarre, basándose en un modelo simplificado del objeto y una serie de preformas que dependen de la mano y de la tarea a realizar. La segunda es un test para evaluar la viabilidad y la calidad de estos agarres.

En [LFP07] presenta un algoritmo de síntesis automática de agarres basándose en una base de datos de posturas de agarres de las manos humanas. Cada vez que se añade un agarre a la base de datos, se seleccionan los puntos de contacto significativos que representan las zonas en la superficie de las manos que se utilizan normalmente en un agarre. Dada una descripción geométrica tridimensional del objeto a agarrar, se buscan en la base de datos las posturas de la mano que más se aproximan al objeto (presentan mayor coincidencia en su forma - algoritmos del vecino más cercano). Para seleccionar el mejor agarre del conjunto preseleccionado se le aplica un test de calidad específico para la mano que compara la habilidad de la mano para aplicar fuerzas en el objeto con las fuerzas necesarias para realizar la tarea. Para discriminar mejor los agarres, permiten seleccionar zonas de agarres preferidas aplicando una función de peso a estas zonas.

En [BS08] utilizan el mismo algoritmo de [LFP07] para determinar las preformas en un sistema donde además se tiene en cuenta el entorno en el que se sitúa el objeto y si el agarre está libre de colisiones. Sin embargo, para objetos más complejos, la determinación de las preformas para agarrar ese objeto se realiza manualmente.

El desarrollo de métodos para determinar las preformas es un tema de interés para los investigadores no sólo en el campo de la robótica sino también en animación digital o en neurociencia.

3

Síntesis de agarres

3.1. Introducción

Una de las formas primarias que un robot tiene de interaccionar con los objetos de su entorno es agarrándolos, tanto si el robot es teleoperado como si dispone de un cierto grado de autonomía. Este trabajo de tesis tiene como finalidad automatizar los procesos de agarre para entornos no estructurados.

Tal y como se ha analizado en el capítulo de introducción existen muchos trabajos de investigación centrados en el agarre de objetos por medio de pinzas y manos robóticas. Así, se ha avanzado significativamente en el análisis de los agarres para comprobar si cumplen ciertas condiciones respecto a resistir perturbaciones externas, como *force closure* y *form closure* [Bic95]; en la propuesta de [SCR06] se definen diferentes índices de calidad para seleccionar el mejor agarre; y en el desarrollo de algoritmos de síntesis de agarres que permiten calcular configuraciones de agarre que cumplan ciertas medidas de calidad. El objetivo de estos algoritmos es calcular los puntos donde los dedos de la pinza contactan con la superficie del objeto para conseguir un agarre óptimo según los criterios de calidad establecidos. Tienen

en cuenta características geométricas de los puntos de contacto, coeficientes de rozamiento entre los dedos y el objeto, las fuerzas aplicadas por los dedos, etc.

Otros trabajos plantean el cálculo de los puntos de contacto desde una perspectiva más cercana a una aplicación real, de modo que no se busca calcular una configuración óptima de agarre, sino una configuración adecuada a la tarea a realizar. En [CW86] se presenta una taxomonía de agarres para las manos humanas según los agarres estén orientados a tareas o estén orientados a los objetos. Así, dado un objeto prismático a agarrar, como por ejemplo un bolígrafo, la configuración de la mano y las zonas por donde se agarre el objeto vendrán determinadas por lo que se desee hacer con él: cogerlo para escribir o para lanzarlo.

Este será el planteamiento subyacente en este trabajo de tesis: para una pinza con una determinada estructura cinemática, calcular los puntos de contacto del efector de acuerdo con la tarea a realizar y con el objeto a agarrar. Concretamente, vamos a proponer un método formal que nos permita calcular la configuración de agarre del efector.

En el método propuesto se definirán dos funciones de entrada:

- una función para representar al robot con el efector final, donde las zonas de agarre se primarán en función de la tarea a realizar. De esta forma, se distinguirán, si así se desea, determinadas partes de la pinza, como la palma o los dedos, que en cada caso se buscan que contacten con el objeto.

- una función que defina al entorno, donde se distingue entre los obstáculos y el objeto a agarrar. También se podrán primar aquellas zonas de la frontera por donde se desea agarrarlo, siguiendo determinados criterios de calidad o en función de la tarea a realizar. Los puntos de contacto se encuentran en la frontera del objeto, y no en su interior.

La salida será un agarre viable en el sentido de que la mano puede adoptar la configuración de agarre calculada. También, dicho método facilitará encontrar un camino libre de colisiones desde la configuración actual a la configuración de agarre. De esta forma, los agarres obtenidos dispondrán de una de las principales características que se suelen imponer a un agarre para determinar su calidad como es la alcanzabilidad.

El método permite adaptar las funciones de entrada a la tarea específica a desarrollar. Se trata de un método genérico en el sentido de que no está restringido a un determinado efector final. Para un pinza concreta, habrá que particularizar la función que representa al robot según su estructura cinemática. También se pueden considerar manos, si se plantea, por ejemplo, el concepto de *dedos virtuales*. Además dicho método es independiente de la geometría del robot y del objeto.

En el siguiente apartado exponemos el formalismo que permite calcular el conjunto de configuraciones de *grasp* G_M para un determinado objeto M_j, que es independiente de la geometría del robot y del objeto. En esas configuraciones existirá un contacto entre alguna de las partes de agarre del robot A y alguna de las zonas del objeto M_j. En los siguientes apartados aplicaremos el formalismo propuesto a diversos casos de estudio.

3.2. Formalismo propuesto

Se considera un robot A que realiza tareas en un espacio de trabajo $W = R^N$, con $N = 2$ ó 3, donde existen objetos que el robot puede agarrar. El robot A consta de una base y de un efector final diseñado para realizar el agarre de objetos. En A se fija un sistema de referencia cartesiano F_A con respecto al sistema de referencia fijo F_W en W. El espacio de las configuraciones de A es C.

Sea $\mathbf{A}(\mathbf{q})$ el subconjunto de W que representa al robot en la configuración q; y $\mathbf{FA}(\mathbf{q})$ el subconjunto de W ocupado por la frontera de las partes del efector final diseñadas para contactar con un objeto movible. Estas partes del efector final se seleccionarán en función de la tarea que el robot vaya a llevar a cabo. Se define la función que representa al robot $A : C \times W \rightarrow R$ como

$$A(q,x) = \begin{cases} 1 & \text{si } x \in \mathbf{A}(q) \\ V_A & \text{si } x \in \mathbf{FA}(q) \\ 0 & \text{si } x \notin \mathbf{A}(q) \end{cases} \tag{3.1}$$

donde V_A (con $V_A > 1$) es el valor asignado para primar las zonas de la pinza en función del tipo de agarre (prensil, no-prensil, de precisión, etc.), según determinadas taxonomías de agarre.

En el espacio de trabajo W existen obstáculos fijos y otros que el robot puede coger y mover (manipular). A los subconjuntos de W que describen este entorno se les denomina, respectivamente, $\mathbf{B}$ para los obstáculos y $\mathbf{M}$ para los objetos movibles. La **frontera exterior** de estos conjuntos se denota por $\partial^e\mathbf{B}$ y $\partial^e\mathbf{M}$, que corresponde con:

$$\begin{aligned} \partial^e\mathbf{B} &= \partial\mathbf{B}^C \\ \partial^e\mathbf{M} &= \partial\mathbf{M}^C \end{aligned} \tag{3.2}$$

donde $\mathbf{B}^C$ y $\mathbf{M}^C$ denotan, respectivamente, a los conjuntos complementarios de $\mathbf{B}$ y $\mathbf{M}$ (i.e. las zonas del espacio libre que rodean a los obstáculos y a los objetos). En el caso de los objetos movibles los puntos de contacto donde se realizará el agarre se encontrarán sobre $\partial^e\mathbf{M}$.

Se define la función que representa al entorno $E : W \rightarrow R$ como

$$E(x) = \begin{cases} 1 & \text{si } x \in (\mathbf{B} \cup \mathbf{M}) \\ 0 & \text{si } x \notin (\mathbf{B} \cup \mathbf{M}) \end{cases} \tag{3.3}$$

Así mismo, para tener en cuenta la frontera de los objetos, se define el *entorno aumentado* $AE : W \rightarrow R$ como:

$$AE(x) = \begin{cases} 1 & \text{si } x \in (\mathbf{B} \cup \partial^e\mathbf{B}) \\ V_M & \text{si } x \in (\mathbf{M} \cup \partial^e\mathbf{M}) \\ V_{FM} & \text{si } x \in \mathbf{FM} \\ 0 & \text{si } x \notin (\mathbf{B} \cup \partial^e\mathbf{B} \cup \mathbf{M} \cup \partial^e\mathbf{M}) \end{cases} \tag{3.4}$$

donde $\mathbf{FM}$ es el subconjunto de W ocupado por las zonas de la frontera exterior $\partial^e\mathbf{M}$ por las que se desea agarrar el objeto. A los objetos movibles se les asigna un valor V_M para diferenciarlos de los obstáculos.

A continuación se proyectan estos conjuntos en el espacio de las configuraciones C. Así, primero, se define la función $CE : C \rightarrow R$ que representa el interior de los objetos (fijos y movibles) en el C-espacio como:

$$CE(q) = \int A(q,x)E(x)dx \quad \forall q \in C \tag{3.5}$$

De igual forma, se define la función $ACE : C \rightarrow R$ que representa los objetos (fijos y movibles) con su frontera exterior (zona donde se generará el agarre):

$$ACE(q) = \int A(q,x)AE(x)dx \quad \forall q \in C \tag{3.6}$$

Se puede definir la *frontera de agarre* $\partial_g CE$ como el conjunto:

$$\partial_g CE = \{q \in C, , (CE(q) = 0) \ \wedge \ (ACE(q) > 0)\} \tag{3.7}$$

Este conjunto contendrá, como demostraremos, el conjunto de configuraciones objetivo de la síntesis de agarres.

Teorema 3.1

Una configuración de un robot será de agarre si pertenece al conjunto $\partial_g CE$ (frontera de agarre).

Demostración:

La demostración se realizará por reducción al absurdo al suponer que existe una configuración $q' \in CE, , q' \notin \partial_g CE$ que si se puede considerar como una configuración de agarre, pero perteneciente al espacio libre. Como se considera que en una configuración de agarre existe

contacto y se entiende como tal que la intersección entre la frontera externa del robot y el obstáculo es no nula, esto es:

$$\partial^e \mathbf{A}(q') \cap (\mathbf{B} \cup \mathbf{M}) \neq \emptyset \tag{3.8}$$

Es decir, existe al menos un punto del contorno exterior del robot que intersecta (*toca*) con los obstáculos (fijos o movibles). Sea $x' \in W$ dicho punto en el espacio de trabajo que cumple:

$$\begin{aligned} x' &\in \partial^e \mathbf{A}(q') \\ x' &\in (\mathbf{B} \cup \mathbf{M}) \end{aligned} \tag{3.9}$$

Para ello se considera el conjunto $\partial^e(\mathbf{B} \cup \mathbf{M})_{x'}$ siendo $(\mathbf{B} \cup \mathbf{M})_{x'}$ el conjunto constituido por una bola puntual alrededor de x'. Además como

$$x' \in \partial^e \mathbf{A}(q') \equiv x' \in \partial \mathbf{A}^C \tag{3.10}$$

Debe existir un punto $y \in \mathbf{A}(q')$ que es vecino[1] de x'. Por esta razón

$$A(q', y) \neq 0 \tag{3.11}$$

Ahora bien como $x' \in \mathbf{B} \cup \mathbf{M}$ su vecino y cumplirá

$$AE(y) \neq 0 \tag{3.12}$$

Por tanto, ese verifica que

$$(A(q', y) \neq 0) \ \wedge \ (AE(y) \neq 0) \Rightarrow ACE(q') \neq 0 \tag{3.13}$$

Pero si se considera que q' es una configuración libre:

$$CE(q') = 0 \tag{3.14}$$

y por la definición $q' \in \partial_g CE$

$$q' \in \partial_g CE \tag{3.15}$$

que está en contradicción con lo establecido, por lo que queda demostrado el teorema.

En todo caso la resolución del problema de síntesis se concreta en la selección de una configuración del conjunto $\partial_g CE$ que determine el mejor agarre. Para ello se debe considerar el siguiente corolario.

Corolario 3.1

La configuración de agarre será la que verifique que

$$G_{M_j} = \left\{ q_{grasp} \in \partial_g CE, , ACE(q_{grasp}) = \max_{q \in \partial_g CE} (ACE(q)) \right\} \tag{3.16}$$

[1] Se dice que un punto es vecino de otro si pertenece a una bola centrada en x' de radio mínimo. En entornos discretos esta relación estará relacionada con el nivel de discretización con el que se trabaje.

Es evidente que la definición de la función *entorno aumentado* AE, así como la definición de la función característica del robot $A(q, x)$ tendrán un papel fundamental en la determinación del máximo. Así, V_M es el valor asignado a las posiciones ocupadas por los objetos movibles, que servirá para distinguirlos del resto de los objetos (obstáculos); V_{FM} es el valor asignado a las zonas preferidas de agarre del objeto movible en función de criterios normalmente asociados a la tarea; y V_A es el valor asignado a las zonas de la pinza con las que se desea coger el objeto, que estarán determinadas tanto por el diseño del efector final como por la tarea.

3.3. Robot rígido con zonas de agarre

En esta sección vamos a aplicar el formalismo propuesto para calcular el conjunto de configuraciones de *grasp* a un robot A que se desplaza y cambia su orientación en un espacio de trabajo $W \subset R^2$, donde existen objetos movibles M y obstáculos B.

Sea A un objeto rígido que dispone de cavidades de cualquier geometría para agarrar objetos. La cavidad modela la forma más simple de una pinza, sin considerar el grado de libertad asociado a su apertura. El robot A puede agarrar un objeto movible M_j cuando una cavidad encaje con alguna zona de M_j. En la figura 3.1 puede observarse un ejemplo de este escenario que coincide con el planteado en [Lat91b]. Los objetos movibles M_j aparecen en verde mientras los obstáculos B_i están coloreados en azul. En el robot A se ha coloreado en rojo la frontera de la zona interior de la cavidad, con la que se desea contactar con los objetos movibles. Aunque en la figura aparece una determinada representación geométrica de todos los objetos (A, B_i,M_j) la geometría del robot y la de los objetos puede ser cualquiera.

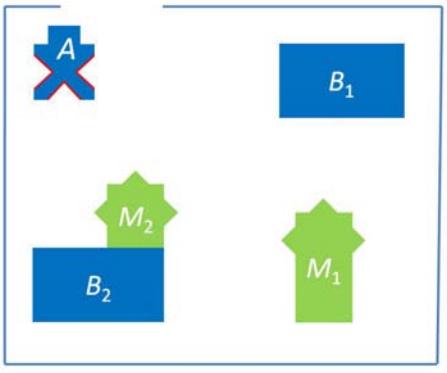


Figura 3.1: Robot rígido A y entorno con objetos movibles M_j y obstáculos B_k.

En W se define un sistema de referencia cartesiano F_W fijo, y además se define un sistema de referencia F_A que se mueve con el robot (figura 3.2). Una configuración q de A vendrá

especificada por la posición (x_r, y_r) y la orientación (θ_r) de F_A con respecto de F_W. En este caso una configuración viene parametrizada por (x_r, y_r, θ_r), con $C \subset R^2 \times [-\pi, \pi)$. Como parametrización de W se utilizan las coordenadas cartesianas (x, y).

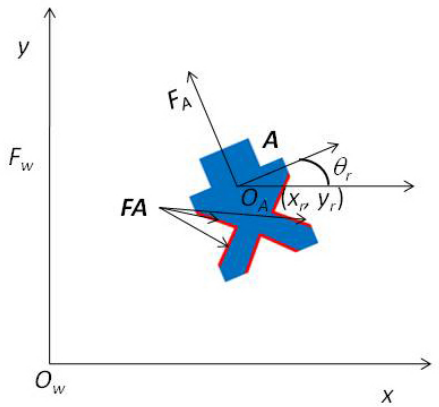


Figura 3.2: Elección de los sistemas de referencia.

La función que representa al entorno $E : W \to R$ estaría dada por

$$E(x,y) = \begin{cases} 1 & \text{si } (x,y) \in (\mathbf{B} \cup \mathbf{M}) \\ 0 & \text{si } (x,y) \notin (\mathbf{B} \cup \mathbf{M}) \end{cases} \tag{3.17}$$

y la del entorno aumentado $AE : W \to R$ por

$$AE(x,y) = \begin{cases} 1 & \text{si } (x,y) \in (\mathbf{B} \cup \partial^e\mathbf{B}) \\ V_M & \text{si } (x,y) \in (\mathbf{M} \cup \partial^e\mathbf{M}) \\ V_{FM} & \text{si } (x,y) \in \mathbf{FM} \\ 0 & \text{si } (x,y) \notin (\mathbf{B} \cup \partial^e\mathbf{B} \cup \mathbf{M} \cup \partial^e\mathbf{M}) \end{cases} \tag{3.18}$$

donde $\mathbf{FM}$ es el subconjunto de los puntos de W ocupado por los puntos de la frontera exterior del objeto movible por donde se ha elegido agarrar el objeto en función de la tarea; V_M es el valor asignado a los objetos movibles y a su frontera exterior $\partial^e\mathbf{M}$, excepto a los puntos de la frontera exterior por donde se ha elegido agarrar el objeto que se les asigna un valor V_{FM}.

En cuanto a la función que representa al robot $A : C \times W \to R$ estaría dada por

$$A(x_r, y_r, \theta_r, x, y) = \begin{cases} 1 & \text{si } (x,y) \in \mathbf{A}_{(x_r,y_r,\theta_r)} \\ V_A & \text{si } (x,y) \in \mathbf{FA}_{(x_r,y_r,\theta_r)} \\ 0 & \text{si } (x,y) \notin \mathbf{A}_{(x_r,y_r,\theta_r)} \end{cases} \tag{3.19}$$

donde $\mathbf{A}_{(\mathbf{x_r},\mathbf{y_r},\theta_\mathbf{r})}$ es el subconjunto de los puntos de W que representa al robot en la posición (x_r, y_r) con orientación θ_r, y $\mathbf{FA}_{(\mathbf{x_r},\mathbf{y_r},\theta_\mathbf{r})}$ es el subconjunto de los puntos pertenecientes a

la frontera de la pinza del robot que se desean contacten con el objeto movible, y V_A es el valor asignado para primar dichos puntos (en la Figura 3.1 aparece resaltada dicha zona en rojo). Normalmente se elige dicho valor teniendo en cuenta que V_A es $k \bullet longitud(\partial A)$ siendo $k > 1$.

Así, la función $CE : C \rightarrow R$ estaría definida como

$$CE(x_r, y_r, \theta_r) = \int A(x_r, y_r, \theta_r, x, y)E(x, y)dxdy \quad \forall (x_r, y_r, \theta_r) \in C \quad (x, y) \in W \tag{3.20}$$

Para simplificar el cálculo de esta integral, definimos unos nuevos sistemas de referencia, de tal forma que trasladamos el origen O_A a O_W. En este nuevo C-espacio existe un conjunto de configuraciones del robot dado por $q = (0, 0, \theta_r)$. Sobre este nuevo sistema de referencia se cumple que

$$A(x_r, y_r, \theta_r, x, y) = A(0, 0, \theta_r, x - x_r, y - y_r) \tag{3.21}$$

Por claridad, utilizamos la siguiente notación

$$A_{(0,0,\theta_r)}(x - x_r, y - y_r) = A(0, 0, \theta_r, x - x_r, y - y_r) \tag{3.22}$$

Con ello obtenemos

$$CE(x_r, y_r, \theta_r) = \int A_{(0,0,\theta_r)}(x - x_r, y - y_r)E(x, y)dxdy \quad \forall (x_r, y_r) \in C \quad (x, y) \in W \tag{3.23}$$

Si consideramos la siguiente notación

$$\overline{A}_{(0,0,\theta_r)}(x_r - x, y_r - y) = A_{(0,0,\theta_r)}(x - x_r, y - y_r) \tag{3.24}$$

entonces se puede calcular la función CE como el producto de convolución de dos funciones definidas en R^2 sobre las dos variables

$$CE(x_r, y_r, \theta_r) = \overline{A}_{(0,0,\theta_r)} * E(x_r, y_r) \tag{3.25}$$

Con el teorema de convolución se llega a

$$\mathcal{F}[CE(x_r, y_r, \theta_r)] = \mathcal{F}[\overline{A}_{(0,0,\theta_r)}]\mathcal{F}[E(x_r, y_r)] \tag{3.26}$$

En la ecuación 3.26 se observa que la transformada de Fourier de CE es el producto de las transformadas, en dos dimensiones, de dos funciones definidas en R^2:

- $\overline{A}_{(0,0,\theta_r)}$ que representa al robot en la posición $(0, 0)$ para cada una de sus orientaciones posibles θ_r

- la función E que define el entorno con los objetos fijos y los movibles.

La representación de CE vendría dada por las proyecciones del entorno E en C para cada una de las orientaciones del robot.

Se procede de igual forma para la función $ACE : C \rightarrow R$ que estaría definida como

$$ACE(x_r, y_r, \theta_r) = \int A(x_r, y_r, \theta_r, x, y) AE(x, y) dxdy$$
$$\forall (x_r, y_r, \theta_r) \in C \quad (x, y) \in W \tag{3.27}$$

Para simplificar el cálculo de esta integral, definimos unos nuevos sistemas de referencia, de tal forma que trasladamos el origen O_A a O_W. En este nuevo C-espacio existe un conjunto de configuración del robot dado por $q = (0, 0, \theta_r)$. Sobre este nuevo sistema de referencia se cumple que

$$A(x_r, y_r, \theta_r, x, y) = A(0, 0, \theta_r, x - x_r, y - y_r) \tag{3.28}$$

y utilizando la misma notación previa se obtiene

$$ACE(x_r, y_r, \theta_r,) = \int A_{(0,0,\theta_r)}(x - x_r, y - y_r) AE(x, y) dxdy$$
$$\forall (x_r, y_r) \in C \quad (x, y) \in W \tag{3.29}$$

Con las mismas consideraciones previas se puede calcular la función ACE como el producto de convolución de dos funciones definidas en R^2 sobre las dos variables

$$ACE(x_r, y_r, \theta_r) = \overline{A}_{(0,0,\theta_r)} * AE(x_r, y_r) \tag{3.30}$$

Con el teorema de convolución se llega a

$$\mathcal{F}[ACE(x_r, y_r, \theta_r)] = \mathcal{F}[\overline{A}_{(0,0,\theta_r)}]\mathcal{F}[AE(x_r, y_r)] \tag{3.31}$$

Para cada una de las posibles orientaciones θ_r, tendríamos un determinado ACE que se calcula como la transformada inversa de la ecuación 3.31.

La frontera de agarre $\partial_g CE$ estaría formada por el conjunto de configuraciones:

$$\partial_g CE = \{q \in C, , (CE(q) = 0) \ \wedge \ (ACE(q) > 0)\} \tag{3.32}$$

Si se establece como criterio de selección de la configuración óptima de agarre que el objeto contacte con la máxima zona de la cavidad del robot, entonces el conjunto de configuraciones de agarre vendría determinado por aquellas configuraciones en la frontera de agarre que alcancen un máximo, considerando cada una de las orientaciones posibles del robot.

$$q_{grasp} \in \partial_g CE, , ACE(q_{grasp}) = \max_{\theta_r \in [0, 2\pi]} (\max_{(x_r, y_r) \in \partial_g CE(\theta_r)} (ACE(q))) \tag{3.33}$$

Con el fin de garantizar que en las configuraciones q_{grasp} se alcanza un máximo, se ha elegido:

- $V_A = k_1 \bullet longitud(\partial A)$ siendo $k_1 > 1$, lo que supone en este caso que producirá la mayor longitud de contacto entre la zona de agarre de A y M_j.
- $V_M(x, y) = 2$, lo que supone que el objeto movible se proyectará en un ACE con mayores valores que otro que represente a un objeto fijo con la misma geometría.
- $V_{FM}(x, y) = k_2 \bullet longitud(\partial^e \mathbf{M})$ siendo $k_2 > 1$, para conseguir que se tenga mayor valor en las zonas de agarre del objeto, o $V_{FM}(x, y) = 2$ cuando no se especifica una determinada zona de agarre, que es el caso del ejemplo que aparece en la figura 3.1.

Con las expresiones previas se puede obtener el conjunto G_{M_j} independientemente de:

- la geometría concreta del robot (forma y dimensiones) y zonas de agarre (número, posición y geometría)
- la geometría exacta del objeto y de su posición en el espacio de trabajo.

3.4. Robot rígido con pinza de dos dedos paralelos

Se considera un robot A que se traslada sin cambiar de orientación sobre un plano ($W \subset R^2$) con un pinza H con dos dedos $[D_1, D_2]$ paralelos que pueden trasladarse simultáneamente en sentidos opuestos en la dirección que une sus centros.

En W se sitúa un sistema de referencia F_W fijo. En la pinza H se sitúa un sistema de referencia F_H, que se mueve con el robot, cuyo origen O_H se encuentra en un punto equidistante entre $[D_1, D_2]$ y el eje x paralelo a los dedos.

Una configuración q vendrá parametrizada por $(x_r, y_r, d_h) \in R^2 \times [0, d_{max}]$, donde (x_r, y_r) son las coordenadas de O_H respecto de F_W y d_h es el grado de libertad asociado al desplazamiento de los dedos, siendo d_{max} la apertura máxima de la pinza.

Se considera que el robot puede coger objetos con determinadas zonas de la parte interna de los dedos (Figura 3.3), que denominamos *finger-tips*. Definimos el subconjunto de W que representa los *finger-tips* en la configuración (x_r, y_r, d_h) como

$$\mathbf{FH}_{(x_r,y_r,d_h)} = \{f_1(x_r, y_r, d_h) \in D_1, f_2(x_r, y_r, d_h) \in D_2\} \tag{3.34}$$

Por tanto, la función que representa al robot con la pinza $AH : C \times W \to R$ estaría dada por

$$AH(x_r, y_r, d_h, x, y) = \begin{cases} 1 & \text{si } (x, y) \in \mathbf{A}_{(x_r,y_r,d_h)} \\ V_H & \text{si } (x, y) \in \mathbf{FH}_{(x_r,y_r,d_h)} \\ 0 & \text{si } (x, y) \notin \mathbf{A}_{(x_r,y_r,d_h)} \end{cases} \tag{3.35}$$

donde $\mathbf{A}_{(\mathbf{x_r},\mathbf{y_r},\mathbf{d_h})}$ el subconjunto de puntos de W que representa al robot en la configuración (x_r, y_r, d_h). V_H es el valor asignado a los *finger-tips* teniendo en cuenta que V_H es $k \bullet longitud(\partial(\mathbf{FH}))$ siendo $k > 1$.

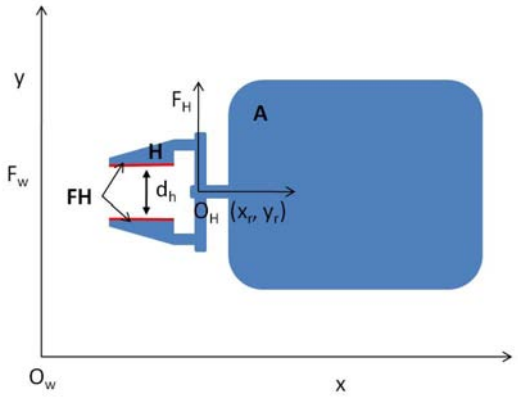


Figura 3.3: Robot rígido con pinzas paralelas.

Considerando la función 3.17 para definir el entorno, la función $CE : C \rightarrow R$ estaría definida como

$$CE(x_r, y_r, d_h) = \int AH(x_r, y_r, d_h, x, y)E(x, y)dxdy$$
$$\forall (x_r, y_r, d_h) \in C \quad (x, y) \in W \tag{3.36}$$

Al trasladar el origen O_H a O_W y definir nuevo sistema de referencia se cumple que

$$AH(x_r, y_r, d_h, x, y) = AH(0, 0, d_h, x - x_r, y - y_r) \tag{3.37}$$

y utilizando la siguiente notación

$$AH_{(0,0,d_h)}(x - x_r, y - y_r) = AH(0, 0, d_h, x - x_r, y - y_r) \tag{3.38}$$

se obtiene

$$CE(x_r, y_r, d_h) = \int AH_{(0,0,d_h)}(x - x_r, y - y_r)E(x, y)dxdy$$
$$\forall (x_r, y_r, d_h) \in C \quad (x, y) \in W \tag{3.39}$$

Si consideramos la siguiente expresión

$$\overline{AH}_{(0,0,d_h)}(x_r - x, y_r - y) = AH_{(0,0,d_h)}(x - x_r, y - y_r) \tag{3.40}$$

entonces, para cada valor de d_h, se puede calcular la función CE como el producto de convolución de dos funciones definidas en R^2 sobre las dos variables

$$CE(x_r, y_r, d_h) = (\overline{AH}_{(0,0,d_h)} * E)(x_r, y_r) \tag{3.41}$$

Con el teorema de convolución se llega a

$$\mathcal{F}[CE(x_r, y_r, d_h)] = \mathcal{F}[\overline{AH}_{(0,0,d_h)}]\mathcal{F}[E(x_r, y_r)] \tag{3.42}$$

Para cada apertura de la pinza d_h, se obtiene la función CE como la transformada inversa en dos dimensiones de la ecuación previa.

De igual forma, considerando la definición 3.18 para la función AE, se llega a que la función $ACE : C \rightarrow R$ estaría definida como

$$ACE(x_r, y_r, d_h) = \int AH(x_r, y_r, d_h, x, y)AE(x, y)dxdy$$
$$\forall (x_r, y_r, d_h) \in C \quad (x, y) \in W \tag{3.43}$$

Realizando los mismos pasos que para calcular CE podemos calcular ACE como

$$ACE(x_r, y_r, d_h) = \int \overline{AH}_{(0,0,d_h)}(x_r - x, y_r - y)AE(x, y)dxdy$$
$$\forall (x_r, y_r, d_h) \in C \quad (x, y) \in W \tag{3.44}$$

Con ello obtenemos

$$ACE(x_r, y_r, d_h) = (\overline{AH}_{(0,0,d_h)} * AE)(x_r, y_r) \tag{3.45}$$

Con el teorema de convolución se llega a

$$\mathcal{F}[ACE(x_r, y_r, d_h)] = \mathcal{F}[\overline{AH}_{(0,0,d_h)}]\mathcal{F}[AE(x_r, y_r)] \tag{3.46}$$

La representación de ACE vendría dada por las proyecciones del entorno AE en C para cada posible apertura de la pinza.

La frontera de agarre $\partial_g CE$ estaría formada por el conjunto:

$$\partial_g CE = \{q \in C, , (CE(q) = 0) \ \wedge \ (ACE(q) > 0)\} \tag{3.47}$$

y el conjunto de configuraciones de agarre vendría determinado por aquellas configuraciones en la frontera de agarres que alcanzan un máximo (considerando como criterio de selección la máxima longitud de contacto entre ambos), para cada una de las posibles aperturas de la pinza.

$$q_{grasp} \in \partial_g CE, , ACE(q_{grasp}) = \max_{d_r \in [0, d_{max}]} (\max_{(x_r, y_r) \in \partial_g CE(d_r)} (ACE(q))) \tag{3.48}$$

Con el fin de garantizar que en las configuraciones q_{grasp} se alcanza un máximo, se ha elegido:

- $V_A = k_1 \bullet longitud(\partial A)$ siendo $k_1 > 1$, lo que supone en este caso que producirá la mayor longitud de contacto entre la zona de agarre de A y M_j

- $V_M(x,y) = 2$, lo que supone que el objeto movible se proyectará en un ACE con mayores valores que otro que represente a un objeto fijo con la misma geometría.
- $V_{FM}(x,y) = k_2 \bullet longitud(\partial^e \mathbf{M})$ siendo $k_2 > 1$, para conseguir que se tenga mayor valor en las zonas de agarre del objeto, o $V_{FM}(x,y) = 2$ cuando no se especifica una determinada zona de agarre.

3.5. Robot con pinza de dos dedos paralelos que se desplaza y gira

Se considera un robot A que se traslada y gira sobre un plano ($W \subset R^2$) con un pinza H con dos dedos $[D_1, D_2]$ paralelos que pueden trasladarse simultáneamente en sentidos opuestos en la dirección que une sus centros.

En W se sitúa un sistema de referencia F_W fijo. En la pinza H se sitúa un sistema de referencia F_H, que se mueve con el robot, cuyo origen O_H en un punto equidistante entre $[D_1, D_2]$ y el eje x paralelo a los dedos.

Una configuración q vendrá parametrizada por $(x_r, y_r, \theta_r, d_h) \in C \subset R^2 \times [-\pi, \pi) \times [0, d_{max}]$, donde (x_r, y_r) representa la posición y θ_r la orientacion de F_H respecto de F_W y d_h es el grado de libertad asociado al desplazamiento de los dedos, siendo d_{max} la apertura máxima de la pinza.

Primamos la parte interna de los dedos para que el robot coja el objeto, marcadas en color rojo en la Figura 3.4 (*finger-tips*). Definimos el subconjunto de W que representa los *finger-tips* en la configuración $(x_r, y_r, \theta_r, d_h)$ como

$$\mathbf{FH}_{(x_r,y_r,\theta_r,d_h)} = \{f_1(x_r, y_r, \theta_r, d_h) \in D_1, f_2(x_r, y_r, \theta_r, d_h) \in D_2\} \tag{3.49}$$

La función característica del robot con la pinza $AH : C \times W \to R$ estaría dada por

$$AH(x_r, y_r, \theta_r, d_h, x, y) = \begin{cases} 1 & \text{si } (x,y) \in \mathbf{A}_{(x_r,y_r,\theta_r,d_h)} \\ V_H & \text{si } (x,y) \in \mathbf{FH}_{(x_r,y_r,\theta_r,d_h)} \\ 0 & \text{si } (x,y) \notin \mathbf{A}_{(x_r,y_r,\theta_r,d_h)} \end{cases} \tag{3.50}$$

donde $\mathbf{A}_{(\mathbf{x_r},\mathbf{y_r},\theta_\mathbf{r},\mathbf{d_h})}$ el subconjunto de puntos de W que representa al robot en la configuración $(x_r, y_r, \theta_r, d_h)$. V_H es el valor asignado a los *finger-tips* teniendo en cuenta que V_H es $k \bullet longitud(\partial(\mathbf{FH}))$ siendo $k > 1$.

La función $CE : C \to R$ estaría definida como

$$CE(x_r, y_r, \theta_r, d_h) = \int AH(x_r, y_r, \theta_r, d_h, x, y) E(x,y) dxdy$$
$$\forall (x_r, y_r, \theta_r, d_h) \in C \quad (x,y) \in W \tag{3.51}$$

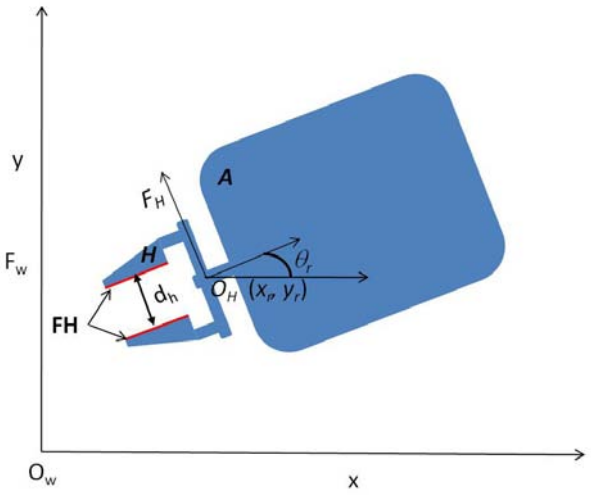


Figura 3.4: Robot rígido con pinzas paralelas que se desplaza y gira libremente.

y procediendo de forma similar que en el apartado 3.4 se llega a

$$\mathcal{F}[CE(x_r, y_r, \theta_r, d_h)] = \mathcal{F}[\overline{AH}_{(0,0,\theta_r,d_h)}]\mathcal{F}[E(x_r, y_r)] \tag{3.52}$$

Para cada una de las orientaciones del robot y para cada una de las aperturas de la pinza, se obtendría un CE como la transformada inversa de la ecuación previa.

De igual forma la función $ACE : C \to R$ estaría definida como

$$ACE(x_r, y_r, \theta_r, d_h) = \int \overline{AH}_{(0,0,\theta_r,d_h)}(x_r - x, y_r - y)AE(x, y)dxdy$$
$$\forall (x_r, y_r, \theta_r, d_h) \in C \quad (x, y) \in W \tag{3.53}$$

Entonces se puede calcular la función ACE como el producto de convolución de dos funciones definidas en R^2 sobre las dos variables para cada valor de θ_r y de d_h

$$ACE(x_r, y_r, \theta_r, d_h) = (\overline{AH}_{(0,0,\theta_r,d_h)} * AE)(x_r, y_r) \tag{3.54}$$

Con el teorema de convolución se llega a

$$\mathcal{F}[ACE(x_r, y_r, \theta_r, d_h)] = \mathcal{F}[\overline{AH}_{(0,0,\theta_r,d_h)}]\mathcal{F}[AE(x_r, y_r)] \tag{3.55}$$

La representación de ACE vendría dada por las proyecciones del entorno AE en C para cada una de las orientaciones y para cada una de las aperturas de la pinza del robot.

La frontera de agarre $\partial_g CE$ estaría formada por el conjunto:

$$\partial_g CE = \{q \in C, , (CE(q) = 0) \ \wedge \ (ACE(q) > 0)\} \tag{3.56}$$

y el conjunto de configuraciones de agarre vendría determinado por aquellas configuraciones en la frontera de agarres que alcanzan un máximo, considerando todas las orientaciones y

cada una de las posibles aperturas de la pinza.

$$q_{grasp} \in \partial_g CE, , ACE(q_{grasp}) = \max_{d_r \in [0, d_{max}]} (\max_{\theta_r \in [0, 2\pi]} (\max_{(x_r, y_r) \in \partial_g CE(d_r, \theta_r)} (ACE(q)))) \quad (3.57)$$

Al igual que en los casos anteriores y con el fin de garantizar que en las configuraciones q_{grasp} se alcanza un máximo, se ha elegido:

- $V_A = k_1 \bullet longitud(\partial A)$ siendo $k_1 > 1$, lo que supone en este caso que producirá la mayor longitud de contacto entre la zona de agarre de A y M_j
- $V_M(x, y) = 2$, lo que supone que el objeto movible se proyectará en un ACE con mayores valores que otro que represente a un objeto fijo con la misma geometría.
- $V_{FM}(x, y) = k_2 \bullet longitud(\partial^e \mathbf{M})$ siendo $k_2 > 1$, para conseguir que se tenga mayor valor en las zonas de agarre del objeto, o $V_{FM}(x, y) = 2$ cuando no se especifica una determinada zona de agarre.

3.6. Aportaciones

En este capítulo hemos propuesto un método formal para la síntesis de agarres. Con respecto a un planificador de manipulaciones podemos afirmar que la potencia de nuestro método reside principalmente en que permite calcular diversas entradas al planificador:

- El conjunto de configuraciones de agarre para los objetos movibles, donde el planificador elegirá la más conveniente. En la determinación de los agarres óptimos se puede incorporar información procedente de otros métodos, bien de análisis de agarres o de métricas de agarres, relacionada con la tarea a realizar, marcando las zonas preferidas de agarre en pinza y/o en objeto.
- La proyección del entorno en el espacio de las configuraciones del robot, que serviría para calcular los caminos de tránsito a la configuración de agarre seleccionada, disponiendo de las configuraciones que producen colisión con los obstáculos. Esta información estaría disponible con el cálculo de la función $CE(q)$ con la expresión 3.5.
- La proyección del entorno, excluido el objeto agarrado, en el espacio de las configuraciones del objeto compuesto por el robot y el objeto agarrado, que serviría para calcular los caminos de transferencia a la configuración donde se desea dejar el objeto. Esta información estaría disponible con el cálculo de la función $CE(q)$ con la expresión 3.5. Para ello se utilizará la definición de la función A en la expresión 3.1, considerando

en $\mathbf{A}(q)$ el objeto compuesto por el robot y el objeto agarrado y $V_A = 1$; en la definición de las funciones *entorno* E en la expresión 3.3 y *entorno aumentado* AE en la expresión 3.4, se tendrán en cuenta únicamente los obstáculos y no el objeto movible con $V_M = 0$ y $V_{FM} = 0$.

4

Propuesta de algoritmos de agarre, criterios de calidad y accesibilidad

En este capítulo se describirán los algoritmos desarrollados que permiten validar el formalismo propuesto para obtener el conjunto de configuraciones de agarre. Para ello se partirá de escenarios sencillos que permitan aplicar el método propuesto, para una pinza con dos dedos que agarra una pieza. Se mostrará cómo los algoritmos son independientes de la forma de la pieza y de la geometría de la pinza. Además, se propondrá un criterio de calidad para conseguir inmovilizar la pieza y otro criterio orientado a la tarea. Por último, situamos nuestra propuesta en un escenario con la presencia de obstáculos y objetos que pueden ser agarrados por el robot. De esta forma pondremos de manifiesto que en los algoritmos se dispone de una estructura de datos para examinar la accesibilidad a las configuraciones de agarre calculadas.

4.1. Discretización de los espacios W y C

Antes de exponer los algoritmos que permitirán evaluar las expresiones propuestas en el formalismo para obtener el conjunto de configuraciones de agarre se hace necesario discretizar el espacio de trabajo W y de las configuraciones C. Esto implica que las funciones A, E y AE estarán definidas sobre estos espacios discretos, y consecuentemente las funciones CE y ACE.

Sea x un punto de W y q una configuración de C, parametrizados, respectivamente, por $x = (x_1, x_2, \cdots, x_n)$ y $q = (q_1, q_2, \cdots, q_m)$. En general, n es 2 ó 3, mientras que m depende del número de grados de libertad del robot considerado. Sobre el intervalo de definición de cada una de estas coordenadas se distribuyen de manera uniforme un número finito de puntos. Por ejemplo, si una de las coordenadas de x o de q es una variable lineal $d \in [a, b]$, se distribuyen uniformemente N puntos, dando lugar a un vector cuya componente i-ésima sería

$$a + i\frac{b-a}{N}$$

y si la coordenada es angular, distribuiríamos M puntos sobre el intervalo $\theta \in [-\pi, \pi]$, por lo que la componente i-ésima del vector θ sería

$$\theta_i = -\pi + i\frac{2\pi}{M}$$

Para simplificar la presentación del método, se supone que todas las coordenadas se discretizan con el mismo número de puntos. Sin embargo, esta suposición no es cierta, pues tanto N como M se deben fijar de forma que la representación del entorno del robot (formada por los obstáculos y los objetos movibles) en ambos espacios, W y C, sea correcta. Así, para representar un espacio de trabajo de 2 dimensiones se utilizará una matriz de dimensión $\underbrace{N \times N}_{n=2}$ o para 3D $\underbrace{N \times N \times N}_{n=3}$, mientras que el espacio de las configuraciones se representrá en otra matriz de dimensión $\underbrace{N \times \cdots \times N}_{m}$, siendo m el número de grados de libertad del robot.

Se puede utilizar el índice $i \in D$, siendo $D = \{1, \cdots, N\}$, como variable independiente para representar cada uno de los N puntos que se han distribuido sobre el intervalo de definición de cada coordenada de $x \in W$ o $q \in C$.

4.1.1. Discretización de las funciones A, E y AE

Al discretizar el dominio de la función que representa al robot A, se obtendrá la función discreta $A^* : \underbrace{D \times \cdots \times D}_{m} \times \underbrace{D \times \cdots \times D}_{n} \rightarrow R$ Mediante una matriz se representará al robot

en una determinada configuración q en el espacio de trabajo. Un elemento de esta matriz tomará el valor '1' si la celda está ocupada por el robot; un valor $V_A > 1$ si se corresponde con la zona de agarre de la pinza; y '0' en caso contrario.

De igual forma, la función del *entorno* $E^* : \underbrace{D \times \cdots \times D}_{n} \rightarrow R$ se definirá en una matriz que representa a los obstáculos y a los objetos movibles en el espacio de trabajo. Un elemento de esta matriz E^* tomará un valor '1' si existe un obstáculo fijo o un objeto movible en la celda que representa dicho elemento; y un valor '0' en caso contrario.

La función discretizada del *entorno aumentado* $AE^* : \underbrace{D \times \cdots \times D}_{n} \rightarrow R$ se definirá en una matriz que representa a los obstáculos fijos y movibles, junto con su frontera exterior, en el espacio de trabajo. Un elemento de esta matriz AE^* tomará un valor '1' si existe un obstáculo fijo en la celda que representa dicho elemento y en las celdas de su frontera exterior (entendiendo por frontera exterior la frontera de la matriz complementaria de AE^*); un valor V_M en las posiciones ocupadas por el objeto movible y en las de su frontera exterior; un valor V_{FM} en los puntos de la frontera exterior del objeto movible si está ocupada por una zona por la que se desea agarrar un objeto; y en cualquier otro caso el valor será '0'.

4.1.2. Discretización de las funciones CE y ACE

De la misma forma que para las funciones previas, se denomina CE^* a la función CE que toma valores sobre el C-espacio discreto, y estaría definida por

$$CE^*(q_j) = \sum_{i=0}^{N-1} A^*(q_j, x_i)E^*(x_i) \quad \forall q_j \in C, \quad \forall x_i \in W \tag{4.1}$$

A partir de ella, la región $\mathbf{CE_f}$, que es el subconjunto de los puntos del C-espacio discreto donde se proyecta el entorno $\mathbf{B} \cup \mathbf{M}$, se define como

$$\mathbf{CE_f} = \{q_j \in C, , CE^*(q_j) > 0\} \tag{4.2}$$

Por tanto, se cumple que un punto del C-espacio discreto es libre si y sólo si $CE^*(q_j) = 0$.

Si se trabaja con los índices, el C-espacio se puede representar por una matriz m-dimensional (o por m submatrices $(m-1)$-dimensionales). Así, la función $CE^* : \underbrace{D \times \cdots \times D}_{m} \rightarrow R$ proporciona una matriz que representa a todos los objetos (fijos y movibles) en el espacio de las configuraciones. Un determinado elemento de esta matriz CE^* tomará un valor no nulo si el robot en la configuración que esa celda representa, intersecta con algún objeto del entorno; o un valor '0' si está libre.

Se procede de igual manera con la función ACE, que representa a todos los objetos (fijos y movibles) junto con su frontera exterior en el C-espacio. Se denomina ACE^* a la función

ACE que toma valores sobre el C-espacio discreto, y estaría definida por

$$ACE^*(q_j) = \sum_{i=0}^{N-1} A^*(q_j, x_i) AE^*(x_i) \quad \forall q_j \in C, \quad \forall x_i \in W \tag{4.3}$$

De la misma forma, la función ACE^* se representa por una matriz m-dimensional, donde un elemento tendrá un valor nulo si no existe intersección entre el robot y todos los objetos del entorno incluidas sus fronteras.

Utilizando las funciones discretizadas previamente, la *frontera de agarre* $\partial_g CE$ estaría definida por el conjunto:

$$\partial_g CE^* = \{q_j \in C, , (CE^*(q_j) = 0) \ \wedge \ (ACE^*(q_j) > 0)\} \tag{4.4}$$

4.2. Algoritmo básico: pinza con apertura fija

Para mostrar los algoritmos de cálculo de las configuraciones de agarres partiremos de ejemplos sencillos e iremos incrementando la dificultad en las secciones sucesivas.

El primer escenario sobre el que vamos a aplicar el formalismo propuesto está formado por una pinza de dedos paralelos con apertura fija y una pieza que debe coger.

4.2.1. Funciones discretizadas y algoritmo

Consideramos que el robot se desplaza (no gira) en un espacio de trabajo $W = [a, b] \times [c, d] \subset R^2$. En W se define un sistema de referencia cartesiano F_W fijo, y elegimos que un punto $x \in W$ esté parametrizado por las coordenadas (x, y). Además, se define un sistema de referencia F_A que se mueve con el robot (figura 4.1). Una configuración q de A vendrá especificada por la posición (x_r, y_r) del origen de F_A con respecto de F_W, puesto que la pinza no puede girar.

Sobre las coordenadas lineales, tanto de W como C, se distribuyen N puntos. Cada celda espacial estaría dada por la expresión

$$\left[a + i\frac{b-a}{N}, a + (i+1)\frac{b-a}{N}\right] \times \left[c + j\frac{d-c}{N}, c + (j+1)\frac{d-c}{N}\right] \tag{4.5}$$

Las funciones discretizadas E^* y AE^* que representan al entorno y al *entorno extendido* estarían representadas por dos matrices de tamaño $N \times N$. El elemento (i, j) de cualquier matriz correspondería con la celda espacial previa.

Para simplificar el cálculo de las funciones CE y ACE hacemos coincidir el origen de ambos sistemas de referencia con lo que

$$A(x_r, y_r, x, y) = A_{(0,0)}(x - x_r, y - y_r) \tag{4.6}$$

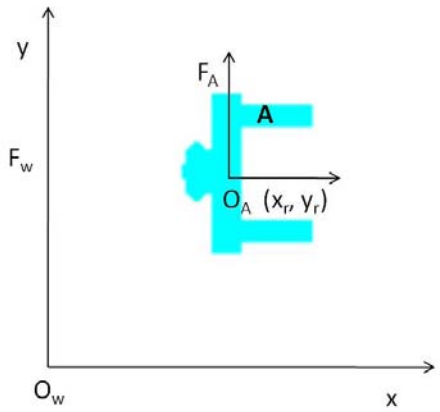


Figura 4.1: Sistemas de referencia para una pinza paralela.

Así la función que define al robot en una configuración determinada (x_r, y_r) y en un punto (x, y) de W coincide con el robot en la configuración $(0, 0)$, y donde se ha realizado una traslación de coordenadas.

De esta forma la función discretizada que representa a los objetos en el C-espacio discreto vendría dada por

$$CE^*(x_{r_k}, y_{r_l}) = \sum_{i,j=0}^{N-1} A^*_{(0,0)}(x_i - x_{r_k}, y_j - y_{r_l}) E^*(x_i, y_j) \tag{4.7}$$

donde la celda espacial (x_{r_k}, y_{r_l}) de C corresponde con el elemento (k, l) de la matriz CE^*. La simplificación proviene de considerar que el cálculo se puede realizar como

$$CE^*(x_{r_k}, y_{r_l}) = (\bar{A}^*_{(0,0)} * E^*)(x_{r_k}, y_{r_l}) \tag{4.8}$$

pues aparece el producto de convolución de la función que describe al robot en la configuración $(0, 0)$ y la que representa a los objetos en W.

Utilizando los índices, la función $CE^* : D \times D \to R$ se expresaría como

$$CE^*(k, l) = (\bar{A}^*_{(0,0)} * E^*)(k, l) \tag{4.9}$$

Luego, si el elemento (k, l) de la matriz donde se representa CE^* tienen un valor 0 entonces la celda espacial correspondiente es una celda libre.

Con las mismas consideraciones, el cálculo de la matriz ACE^* se realizaría mediante

$$ACE^*(k, l) = (\bar{A}^*_{(0,0)} * AE^*)(k, l) \tag{4.10}$$

Tanto CE^* como ACE^* se encuentran representadas cada una en una matriz de tamaño $N \times N$.

Y la *frontera de agarre* $\partial_g CE$ estaría definida por el conjunto:

$$\partial_g CE^* = \{(x_{r_k}, y_{r_l}) \in C, , (CE^*(x_{r_k}, y_{r_l}) = 0) \ \wedge \ (ACE^*(x_{r_k}, y_{r_l}) > 0)\} \tag{4.11}$$

Teniendo en cuenta las expresiones previas el algoritmo que proponemos para calcular las configuraciones de agarre es el algoritmo 4.1.

Algoritmo 4.1: Algoritmo básico de obtención de agarres

Entrada: Matriz del robot (A^*), matriz del entorno (E^*)

Salida: Configuraciones de agarre (Q_{grasp})

1 **Inicio**

2 $\partial {E^*}^C \leftarrow$ Obtener la frontera de la matriz complementaria del entorno;

3 $AE^* \leftarrow E^* + \partial {E^*}^C$; `/*Obtener el entorno aumentado*/`

4 $CE^* \leftarrow \overline{A^*} * E^*$;

5 $ACE^* \leftarrow \overline{A^*} * AE^*$;

6 $\partial_g CE^* \leftarrow$ Obtener la frontera de agarre $((CE^* = 0) \ \wedge \ (ACE^* > 0))$;

7 $Q_{grasp} \leftarrow$ Configuraciones con valor máximo en la frontera de agarre;

Para validar el algoritmo vamos a considerar distintas formas para la pinza, diferentes modelos de piezas a agarrar y diferentes objetos presentes en su entorno de trabajo. Con ello se pretende además demostrar que el algoritmo es independiente de la forma de la pinza y de las características geométricas de los objetos en el entorno de trabajo.

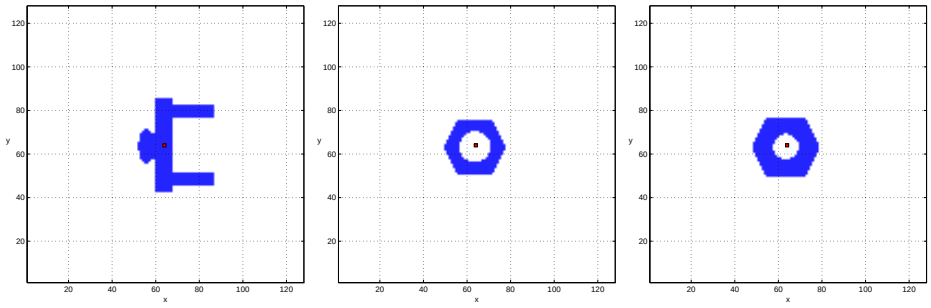
4.2.2. Resultados del algoritmo básico

Inicialmente, suponemos un espacio de trabajo formado únicamente por un objeto (una tuerca) que el robot debe de coger, y no se considera que otros objetos estén presentes en W (Figura 4.2). Hemos elegido una discretización de $N = 128$ para cada una de las coordenadas de W y C. Inicialmente, en la definición de las funciones del robot (A^*) y del entorno (E^* y AE^*) hemos elegido $V_A = 1$, $V_M = 1$ y $V_{FM} = 1$, con el fin de analizar posteriormente la influencia de estos parámetros.

Figura 4.2: Escenario básico con una pinza de dedos paralelos y un objeto (una tuerca) a manipular.

Así, obtenemos las representaciones de dichas funciones (A^*, E^* y AE^*) en las figuras 4.3(a), 4.3(b) y 4.3(c). Las celdas ocupadas por la pinza, el objeto y el objeto aumentado se representan, respectivamente, en estas matrices con el valor 1 y las restantes con 0. El

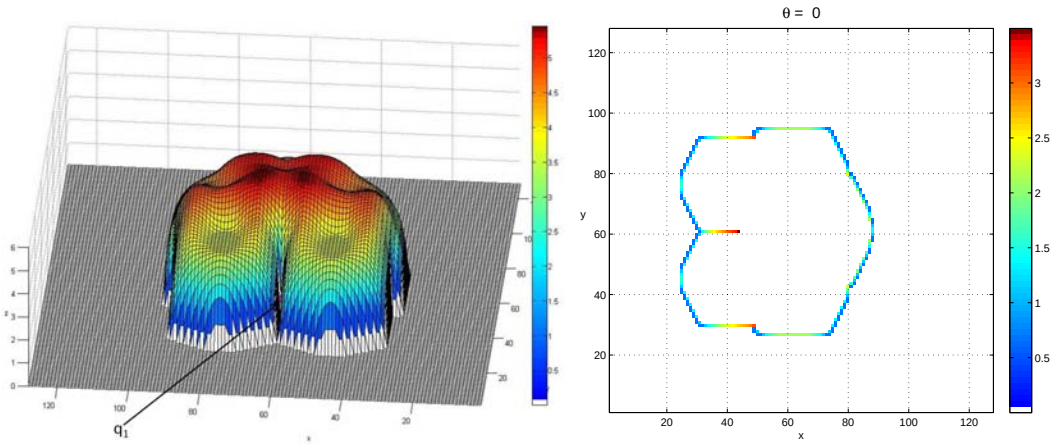
entorno aumentado estaría formado por las celdas que ocupa el objeto en el espacio de trabajo y las celdas que forman su frontera exterior. El punto en color rojo de la figura 4.3(a) muestra el punto donde se ha situado O_A.



(a) La pinza A^* (b) El entorno E^* (c) El entorno aumentado AE^*

Figura 4.3: Discretización de las funciones para el escenario básico con resolución $N = 128$.

En las líneas 4 y 5 del algoritmo 4.1 se calcula la proyección del entorno y del entorno aumentado en el C-espacio, mediante el producto de convolución de dos matrices, aunque se podría haber utilizado el teorema de convolución. El resultado de ambas operaciones son dos matrices de 128×128: una donde se representa CE^* y otra para ACE^*. En la figura 4.4(a) se observa la proyección del objeto en C, donde en los ejes x e y se representan los índices de la matriz, y en z el valor de la celda correspondiente al producto de convolución. La representación de ACE^* sería similar.



(a) C-Obstáculo (b) Frontera de agarre

Figura 4.4: Tuerca proyectada en el C-espacio. En z se muestra el logaritmo de la convolución, para mayor claridad. q_1 es la configuración de agarre.

Las configuraciones de agarre pertenecen a la *frontera de agarre*, que se calcula en la

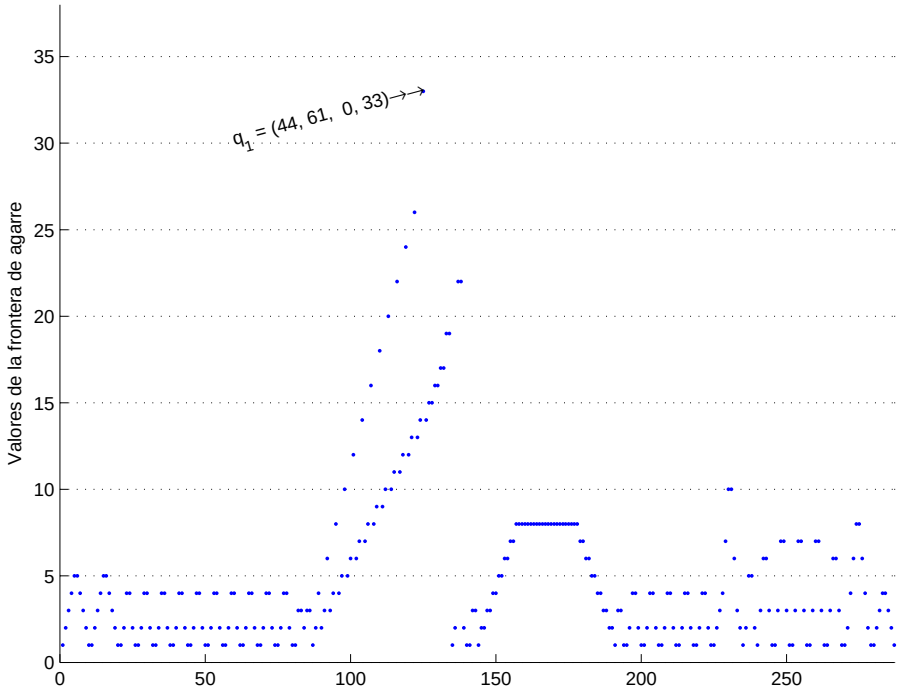


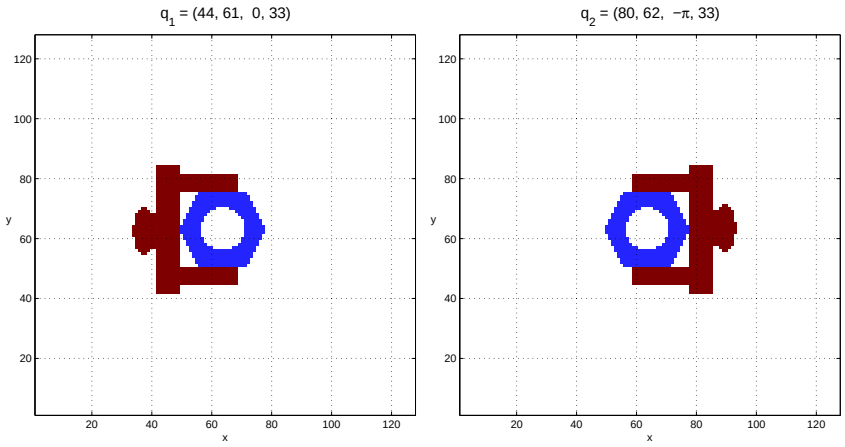
Figura 4.5: Perfil de la frontera de agarre.

línea 6 del algoritmo 4.1, según la expresión 4.11. Los valores máximos de esta frontera representan las zonas de máximo contacto entre la pinza y el objeto a agarrar y, por tanto, atendiendo a nuestro criterio de calidad de agarre, es en esta zona donde se produce el mejor agarre. En la figura 4.4(b) se muestra la frontera de agarre que, para mayor claridad, se ha representado en 2 dimensiones. Existe una configuración, la correspondiente a los índices $(44, 61)$, que tiene el valor máximo (de 33, en color marrón). Dicha configuración será la *configuración de agarre*.

Para observar mejor los valores de la frontera se ha obtenido *el perfil de la frontera de agarre* (figura 4.5). Se etiquetan los puntos de la figura utilizando la notación $q_n = (i, j, \theta, \partial_g)$ asociada al valor de los índices y al valor del producto de convolución. Así, el valor máximo se ha etiquetado con $q_1 = (44, 61, 0, 33)$.

En la figura 4.6(a) se muestra la pinza en dicha configuración de agarre con la tuerca, donde se puede observar que la parte interior de los dedos de la pinza contactan con dos lados de la tuerca.

Para la orientación que se ha fijado de la pinza, esa configuración es la única donde los dos dedos de la pinza contactan con la máxima zona del objeto. Si se hubiera considerado que la pinza tiene una orientación $-\pi$ se hubiera obtenido la configuración que se muestra en la figura 4.6(b).



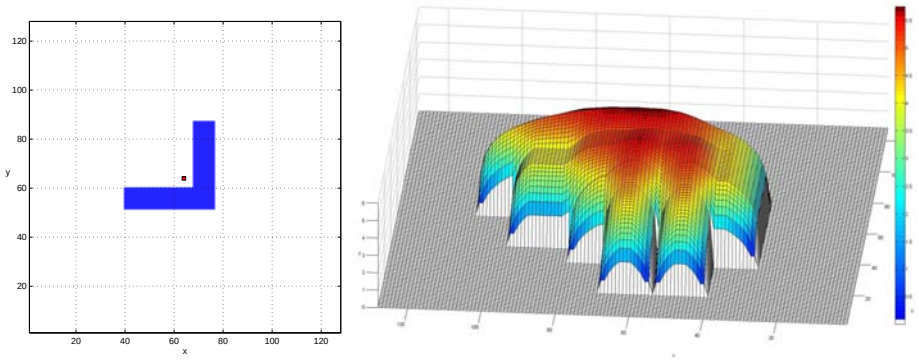
(a) Pinza en la configuración de agarre $(44, 61, 0)$ en W

(b) Pinza en la configuración de agarre $(80, 62, -\pi)$ en W

Figura 4.6: Pinza en la configuración de agarre con dos orientaciones.

4.2.3. Independencia de la forma de la pieza

El algoritmo propuesto es independiente del objeto a agarrar. Así, para calcular las configuraciones de agarre de un objeto en forma de "L", que aparece en la figura 4.7(a), con la misma pinza con orientación 0, únicamente habría que construir el bitmap E^* que representa a dicho objeto y que sería la entrada al algoritmo 4.1. Al proyectar dicho objeto en C, resultado de la línea 4, se obtiene la representación de la figura 4.7(b).



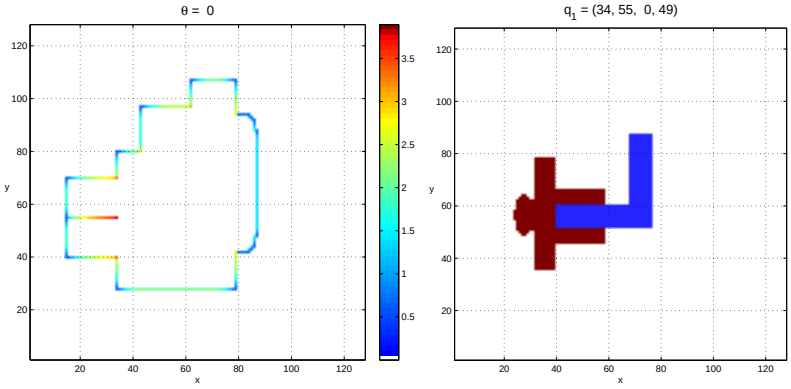
(a) Pieza en W

(b) Pieza en C

Figura 4.7: Pieza a agarrar con forma de "L"

En la línea 5 de dicho algoritmo se calcula la *frontera de agarre*, cuyo resultado se muestra en la figura 4.8(a). Según la escala de colores, el valor máximo de la frontera (color marrón) se obtendría en la configuración cuyos índices son $(34, 55)$. Si se muestra a la pinza adoptando

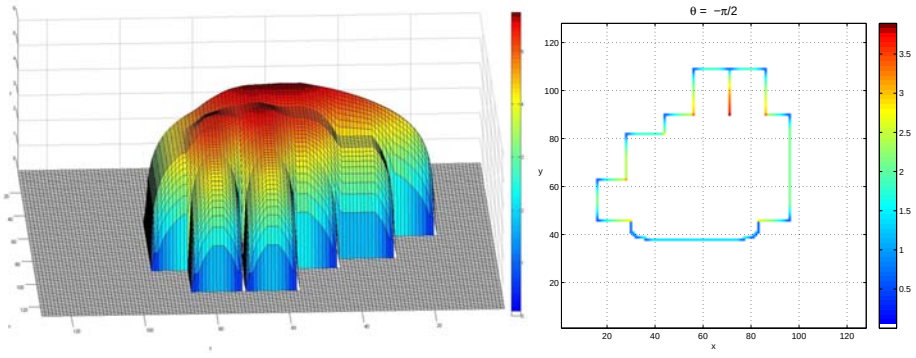
dicha configuración, se observa que el contorno interior de la pinza, correspondiente a los dedos y la palma, están en contacto con el objeto. Existen otras configuraciones donde la zona de los dedos también contactan con el objeto, aunque no así la zona de la palma. Por eso, la configuración que se muestra es la configuración que produce la máxima inmovilización del objeto.



(a) Frontera de agarre (b) Pinza en configuración de agarre

Figura 4.8: Configuración de agarre para pieza con forma de "L"

Para que la pinza agarre al objeto por el otro extremo final de la "L", es necesario que ésta adopte otra orientación. Así, si representamos a la pinza con orientación $-\frac{\pi}{2}$ en el bitmap de entrada al algoritmo A^*, obtenemos la matriz CE^* que aparece en la figura 4.9(a), cuya frontera de agarre aparece en la figura 4.9(b). Sobre dicha frontera, la configuración con un valor máximo (49) corresponde a los índices $(71, 90)$. Si situamos el centro O_A de la pinza en dichas coordenadas tendremos el máximo contorno de contacto con el otro extremo de la pieza en "L"(figura 4.10).



(a) Pieza en "L" en C (b) Frontera de agarre

Figura 4.9: Pieza con forma de "L" en C y su frontera de agarre para pinza con orientación $\theta = -\frac{\pi}{2}$

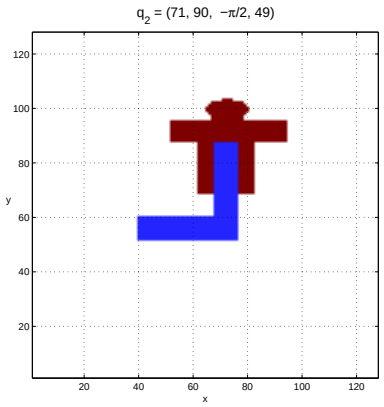
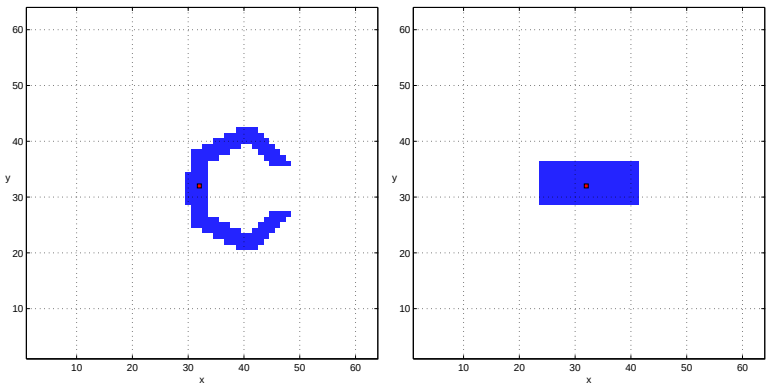


Figura 4.10: Configuración de agarre para pieza en "L" para pinza con orientación $\theta = -\frac{\pi}{2}$.

4.2.4. Independencia con la geometría de la pinza

Supongamos que disponemos de una pinza con una geometría diferente. El algoritmo es válido sin más que representar la pinza en el bitmap A^* de entrada. Éste sería el caso de la pinza de la figura 4.11(a), donde además aparece la pieza a agarrar, con forma de barra (figura 4.11(b)), que corresponde con E^*. Al calcular el producto de convolución de ambos bitmaps obtenemos CE^* (figura 4.12)



(a) La pinza A^* (b) El objeto E^*

Figura 4.11: Discretización del escenario como matrices de $64x64$.

La *frontera de agarre* tiene un valor máximo de 18 en $(21, 32)$, de tal forma que $\partial_g CE^*(21, 32) = 18$. Por tanto, la configuración de agarre corresponde al índice 21 y 32, de los vectores discretizados x_r e y_r, respectivamente. Esta configuración aparece resaltada en la figura 4.12 con un asterisco de color rojo y con una leyenda q_1.

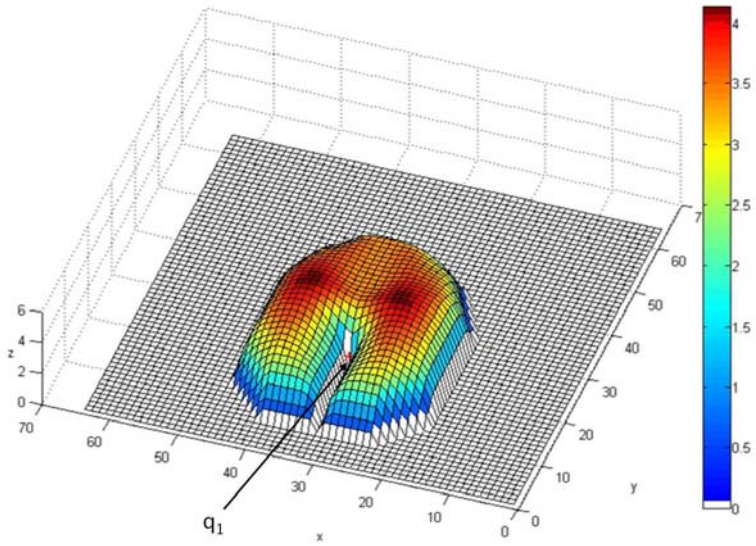


Figura 4.12: Proyección en la pieza en C para el escenario de la figura 4.11. El eje z está en escala logaritmica. q_1 es la configuración de agarre.

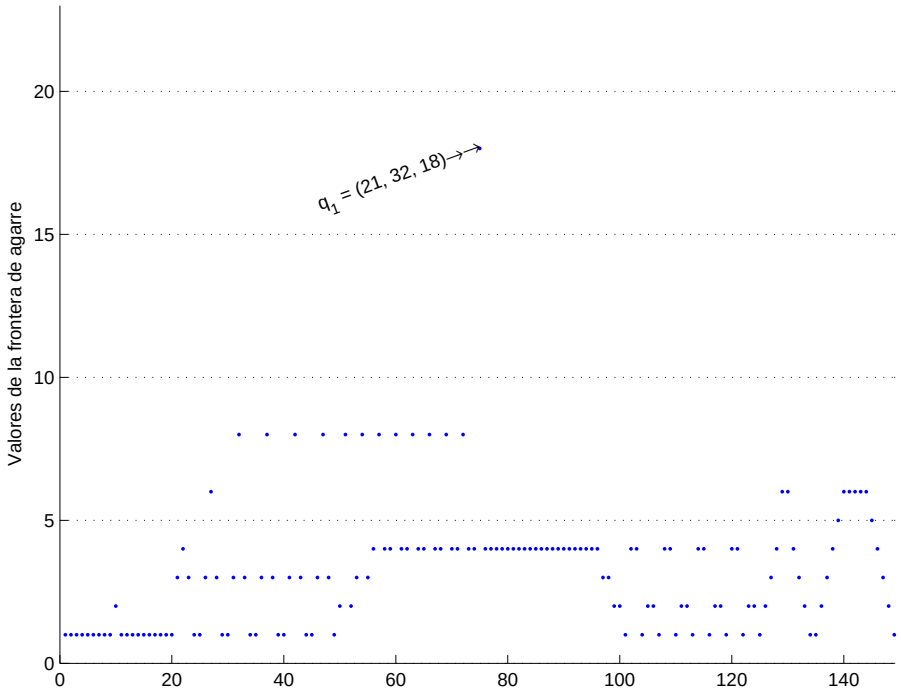


Figura 4.13: Perfil de la frontera de agarre. La configuración de agarre se alcanza en $q_1(x, y) = (21, 32)$ con el valor máximo de 18.

4.2.5. Criterio de calidad

Si situamos la pinza en dicha configuración de agarre (figura 4.14) observamos cómo la pinza agarra la pieza en W, de forma que se produce la mayor longitud de contacto entre ambos: entre el objeto, los dedos y la palma. De forma intuitiva podemos pensar que cuánto mayor sea la longitud de contacto entre pinza y objeto a agarrar, el agarre será *force o form closure*. Por tanto el criterio de calidad, donde la configuración de agarre es la configuración donde se alcanza el máximo valor de la frontera de agarre, se denotará como el criterio de *mayor longitud de contacto*.

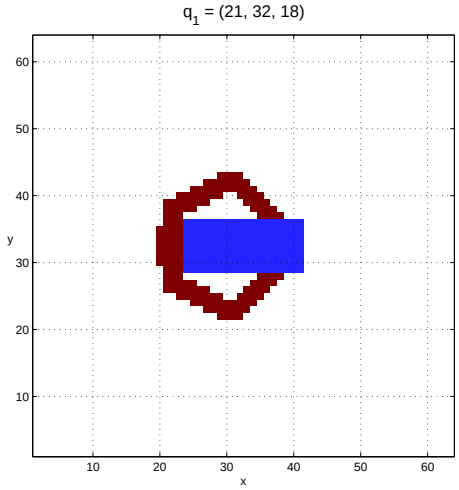
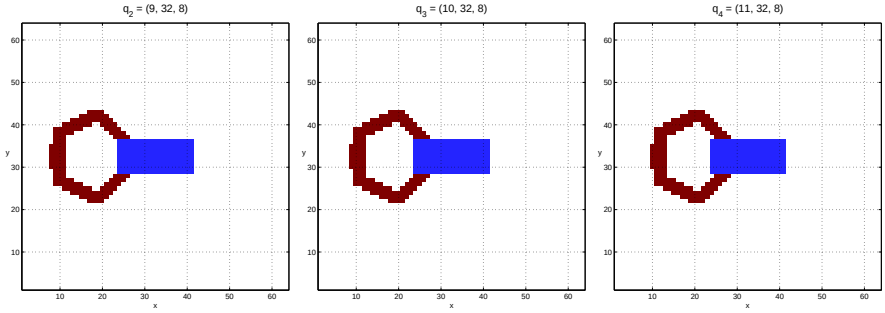


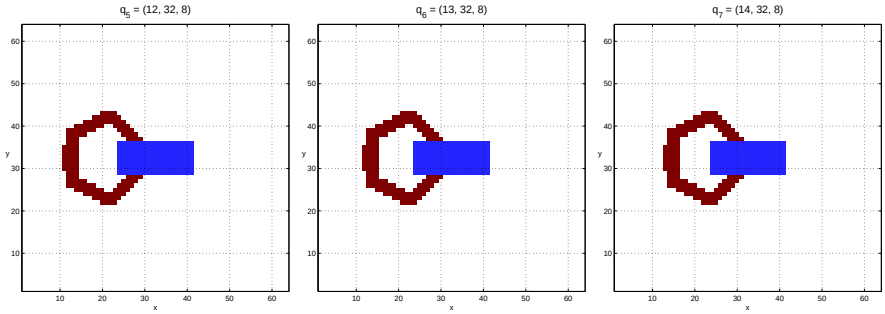
Figura 4.14: Pinza en la configuración del mejor agarre $q_1(x, y, \partial_g) = (21, 32, 18)$.

Sin embargo, en la figura 4.13 también se observan otros máximos locales. Concretamente, son 12 configuraciones cuyo valor de la frontera de agarre es 8. En la tabla 4.1 aparecen los índices en el eje x e y de dichas configuraciones junto con el valor de ACE^*. Como se puede observar en la tercera columna, estas configuraciones de agarre se corresponden con un desplazamiento en el eje x a lo largo de la pieza mientras la cara interior de los dedos de la pinza contacta completamente con la pieza (Figura 4.15). En ellas no se produce contacto con la palma de la pinza. Sin embargo son configuraciones de agarre válidas. Tomando como base métodos de análisis o heurísticas de agarres habría que acotar este conjunto, estableciendo, por ejemplo, un umbral mínimo por debajo del cual se descartarían estas configuraciones.

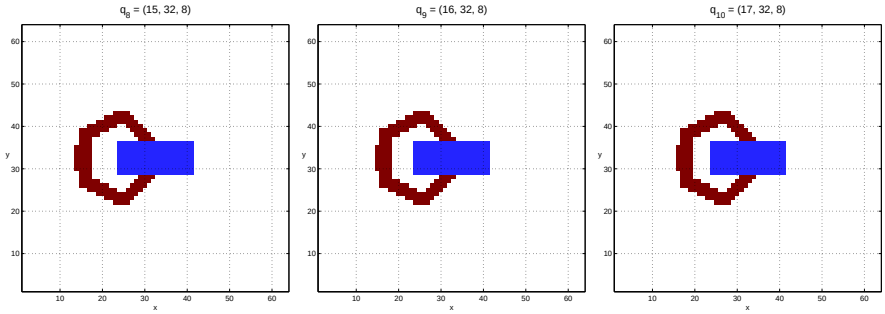
El resto de las configuraciones de la frontera (figura 4.13) con un valor no nulo se corresponden con puntos de la pinza que contactan con el objeto, bien con una parte de la cara interna de los dedos o bien con otras zonas de la pinza que no incluyan los dedos y la palma. Las primeras, donde el contacto se produce no con toda la superficie de los dedos, también son configuraciones posibles aunque el agarre sería menos óptimo.



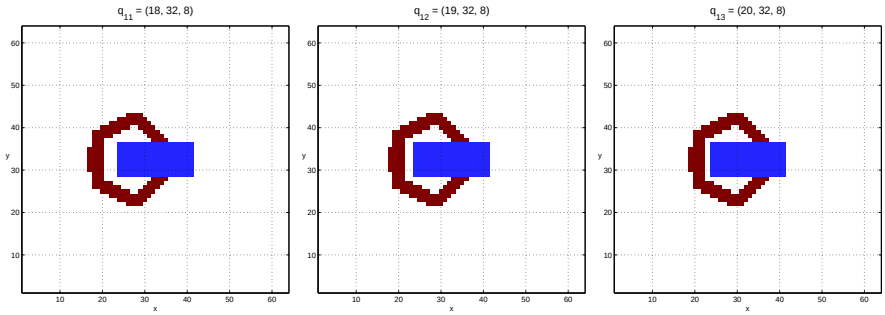
(a) $q_2 = (9, 32, 8)$ (b) $q_3 = (10, 32, 8)$ (c) $q_4 = (11, 32, 8)$



(d) $q_5 = (12, 32, 8)$ (e) $q_6 = (13, 32, 8)$ (f) $q_7 = (14, 32, 8)$



(g) $q_8 = (15, 32, 8)$ (h) $q_9 = (16, 32, 8)$ (i) $q_{10} = (17, 32, 8)$



(j) $q_{11} = (18, 32, 8)$ (k) $q_{12} = (19, 32, 8)$ (l) $q_{13} = (20, 32, 8)$

Figura 4.15: Pinza en otras configuraciones de agarre.

q_j	∂_g	**x**	**y**
q_1	18	21	32
q_2	8	9	32
q_3	8	10	32
q_4	8	11	32
q_5	8	12	32
q_6	8	13	32
q_7	8	14	32
q_8	8	15	32
q_9	8	16	32
q_{10}	8	17	32
q_{11}	8	18	32
q_{12}	8	19	32
q_{13}	8	20	32

Tabla 4.1: Las 12 configuraciones de agarre con máximo local de la frontera de agarre.

Respecto a las segundas, son configuraciones que no son físicamente realizables y que habría que descartar. En el formalismo que proponemos se contempla esta solución en la definición de la función $A(q, x)$, asignando un valor $V_A > 1$ en los puntos **FA** correspondientes a la cara interna de la pinza. Con ello, en el perfil de la frontera de agarre (Figura 4.16) se conseguirán máximos locales más diferenciados respecto al resto de configuraciones con valor no nulo. Dichas configuraciones se podrán descartar definiendo un umbral para este criterio de calidad de *mayor longitud de contacto*.

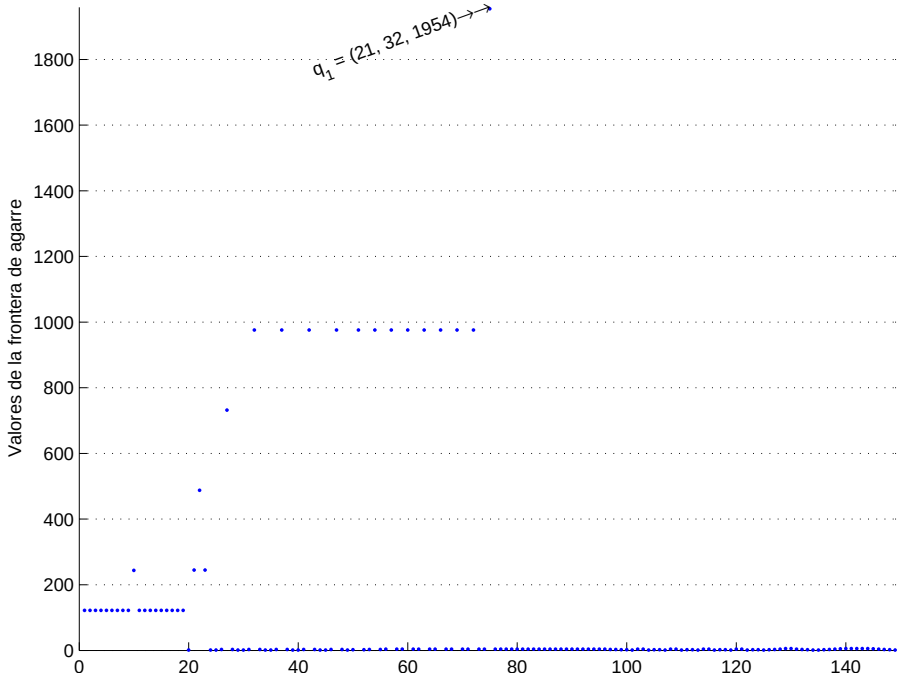


Figura 4.16: Perfil de la frontera de agarre especificando zona de agarre en dedos y palma.

En la siguiente sección veremos cómo introducir en nuestro algoritmo otros criterios de calidad de agarres que complementen al nuestro y que pueden ser obtenidos mediante métodos de análisis o heurísticas de agarres.

4.3. Algoritmo con criterios de calidad orientados a la tarea

Hasta ahora hemos considerado como *mejor agarre* aquel en el que existe la máxima longitud de contacto entre el robot y el objeto a agarrar. Pero no siempre se desean realizar agarres de este tipo, sino que el agarre deseado puede ser completamente independiente de la geometría del objeto y la pinza, y depender exclusivamente de la tarea que se desea realizar posteriormente con el objeto. Por ejemplo, no es lo mismo agarrar una taza para beber que para meterla al lavavajillas. Estos agarres pueden ser definidos mediante el aprendizaje o utilizando ciertas taxonomías de agarres.

En nuestra propuesta, los criterios de calidad orientados a la tarea se pueden tener en cuenta al establecer los valores de V_A, V_M y V_{FM} en la definición de las funciones del robot (A^*) y del entorno (E^* y AE^*).

En las matrices E^* y AE^*, los puntos de W ocupados por el objeto y el objeto aumentado se representan con el valor 1. Además, los puntos de las zonas preferidas para agarrar el objeto, denominados **FM** en nuestro formalismo, tomarán un valor $V_{FM} > 1$, para distinguirlas del resto. El objetivo es que en estas zonas exista contacto y formen parte de la frontera del entorno aumentado. Generalmente, el hecho de seleccionar una zona preferida de agarre en el objeto vendrá determinado por la tarea que se desee, posteriormente, realizar con él.

Como ya hemos comentado, en nuestro formalismo también se pueden establecer las zonas de la pinza por donde mecánicamente es posible efectuar el agarre: por la cara interior de los dedos y la palma. Así, los puntos de W ocupados por la pinza en una determinada configuración se representan en la matriz A^* con el valor 1, las zonas de agarre (internas a la pinza) **FA** con $V_A > 1$ y los puntos restantes por 0. Sin embargo, dependiendo de la tarea puede establecerse una zona de agarre en la pinza, como por ejemplo en las yemas de los dedos. Por tanto, **FA** definiría esa determinada zona de la pinza.

El algoritmo 4.2 amplía el algoritmo básico 4.1 considerando heurísticas que definan la calidad y/o el propósito del agarre. Para considerar estas heurísticas nuestro algoritmo recibe dos parámetros nuevos de entrada denominados **FA** y **FM** que representan, respectivamente, las zonas preferidas de agarre en el robot y en la pieza. Así, las matrices que representan

Algoritmo 4.2: Obtención de agarres con criterios de calidad

Entrada: Matriz del robot (A^*) y el entorno (E^*), zona de agarre en el robot (**FA**), zona preferida de agarre en la pieza (**FM**), factor de agarre (k), umbral (u)

Salida: Configuraciones de agarre (Q_{grasp})

1 **Inicio**
2 $\partial A^*\_inicial \leftarrow$ Obtener el contorno de A^*;
3 $V_A \leftarrow longitud(\partial A^*) \bullet k$;
4 $\overline{A^*} \leftarrow$ Marcar en el robot A^* las zonas preferidas de agarre **FA** con el valor V_A ;
5 $\partial E^{*C} \leftarrow$ Obtener la frontera de la matriz complementaria de E^*;
6 $AE^* \leftarrow E^* + \partial E^{*C}$; `/*Obtener el objeto aumentado*/`
7 $\partial AE^* \leftarrow$ Obtener el contorno de AE^*;
8 $V_{FM} \leftarrow longitud(\partial AE^*) \bullet k$;
9 $AE^* \leftarrow$ Marcar en el objeto aumentado AE^* las zonas preferidas de agarre **FA** con el valor V_{FM} ;
10 $CE^* \leftarrow \overline{A^*} * E^*$;
11 $ACE^* \leftarrow \overline{A^*} * AE^*$;
12 $\partial_g CE^* \leftarrow$ Obtener la frontera de agarre $((CE^* = 0) \ \wedge \ (ACE^* > 0))$;
13 $Q_{grasp} \leftarrow$ Obtener el máximo absoluto y los máximos locales, dentro umbral u, de la frontera de agarre $\partial_g CE^*$;

al robot y al entorno aumentado se modificarán dentro del algoritmo, asignando un valor $V_A = longitud(\partial A^*) \bullet k$ en las zonas de agarre del robot y de $V_{FM} = longitud(\partial AE^*) \bullet k$ en las zonas preferidas de agarre de la pieza, con $k > 1$ en ambos casos. De esta forma se distingue la zona de contacto deseada de otras zonas del objeto donde pueda también existir contacto, y/o también se distingue en la pinza.

4.3.1. Objeto con zonas de agarre

En función de nuestra tarea queremos agarrar una pieza por una determina zona como en el escenario que se muestra en la figura 4.17donde se desea agarrar la barra por su parte superior, en la zona marcada en color rojo.

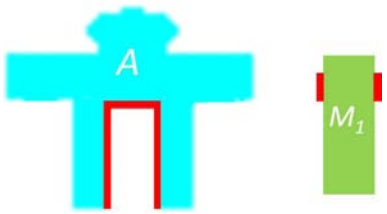
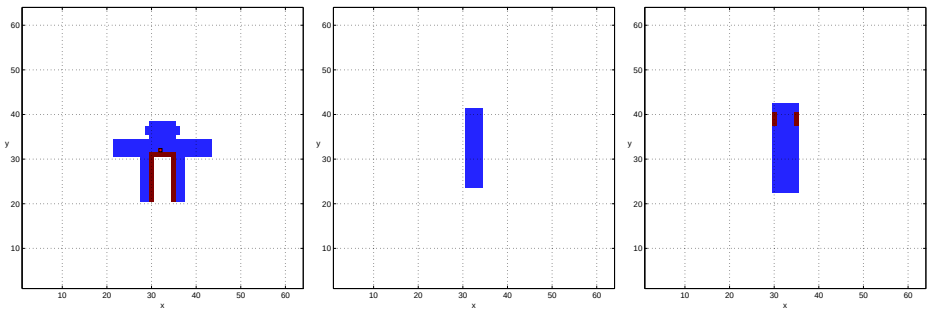


Figura 4.17: Escenario con zona de agarre marcada en color rojo en pinza y objeto movible.



(a) Pinza A^* (b) Objeto movible E^* (c) AE^* con zona de agarre

Figura 4.18: Escenario discretizado.

Para AE^*, se seleccionan las zonas preferidas de agarre de la pieza **FM**, que son las celdas marcadas en color rojo en la figura 4.18(c), y, concretamente, forman parte de su frontera. A estas celdas, dentro del algoritmo, se les asigna un valor $V_{FM} = longitud(\partial AE^*) \bullet k$. En A^* (figura 4.18(a)) , se especifica la cara interna **FA** y, dentro del algoritmo, se le asigna un valor $V_A = longitud(\partial A^*) \bullet k$.

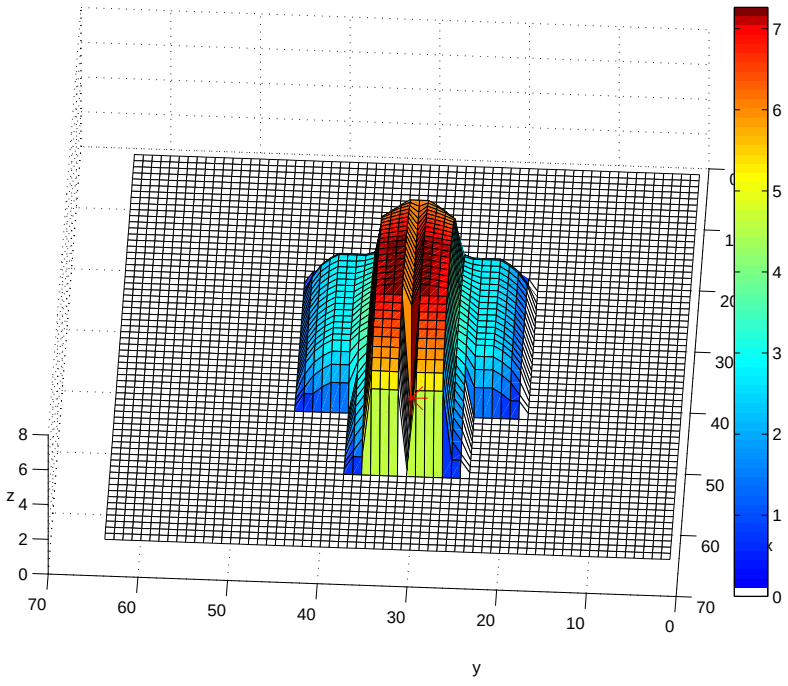


Figura 4.19: Pieza barra vertical proyectada en C.

La pieza proyectada en el C-espacio y el perfil de la frontera se observan en las figuras 4.19 y 4.20. Llevando la pinza a la configuración $q_1 = (31, 42)$ donde se alcanza el valor máximo (30800) de la frontera se consigue el mejor agarre (figura 4.21(a)).

Si analizamos el perfil de la frontera de agarre (figura 4.20) observamos que existen otras configuraciones donde se alcanzan máximos locales. Estas configuraciones podrían ser relevantes en otras situaciones, como cuando q_1 no sea accesible, y podrían ser útiles al planificador de manipulaciones. Por tanto, podríamos obtener un conjunto de configuraciones posibles fijando un umbral, que sería un parámetro de entrada al algoritmo 4.2. El umbral fijará un límite en el valor de la frontera para obtener el conjunto de configuraciones de agarre posibles.

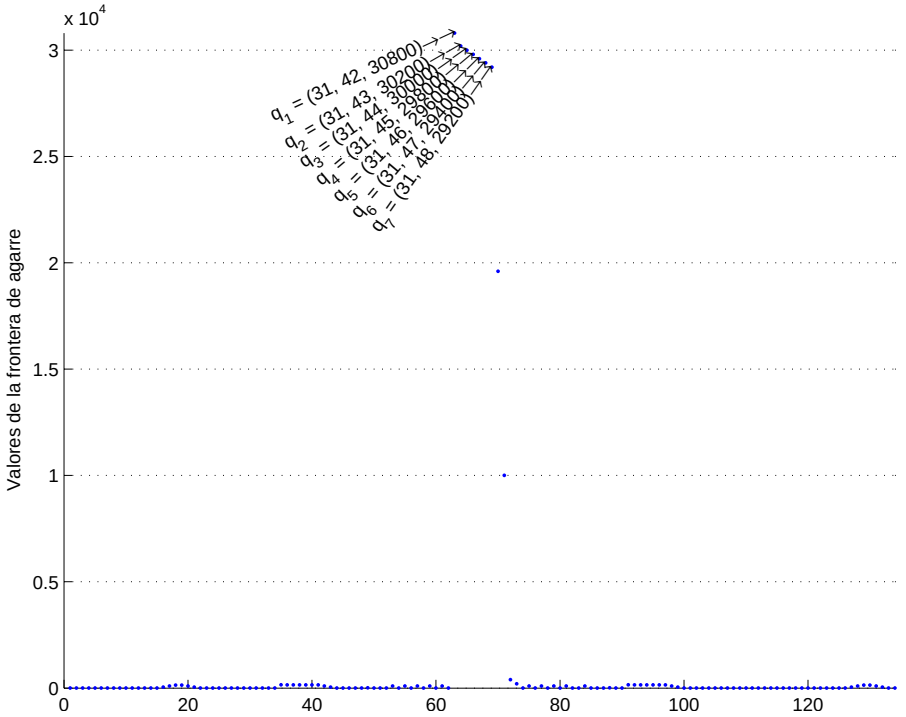


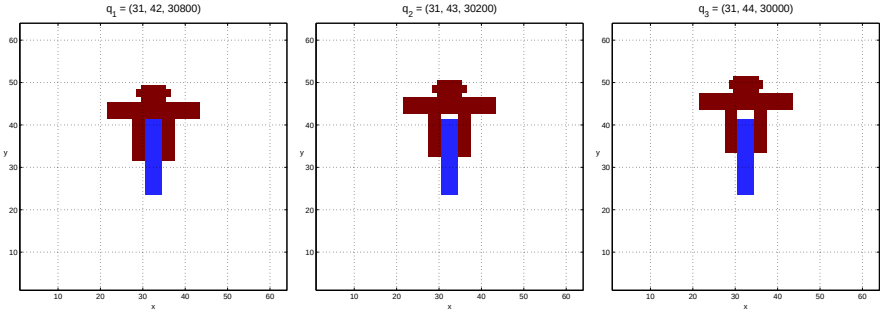
Figura 4.20: Perfil de la frontera de agarre con máximos locales seleccionado con umbral u.

En la figura 4.21 se muestra cómo la pinza trata de agarrar al objeto preferentemente en la zona elegida, con las configuraciones que cumplen el criterio de calidad exigido por el valor umbral. Los agarres están ordenados de mayor a menor valor de la frontera que se muestran en la parte inferior de cada una de las gráficas de la figura 4.21.

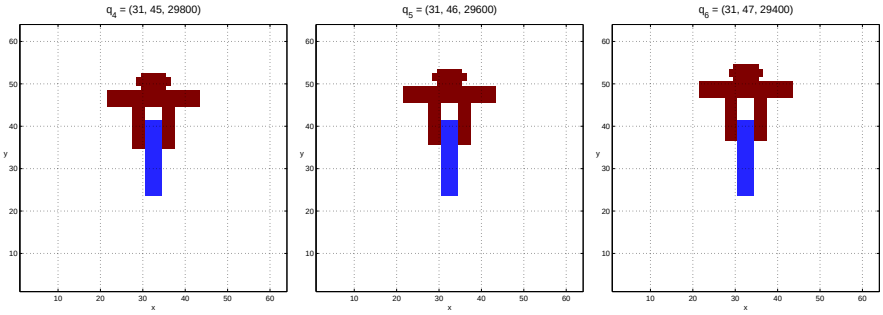
4.3.2. Pinza con zonas de agarre

En función de nuestra tarea deseamos agarrar una pieza con una determina zona de la pinza. Consideraremos el escenario de la figura 4.22, donde se han definido las yemas de los dedos como zona de agarre en la pinza. También se ha especificado una zona para agarrar el objeto.

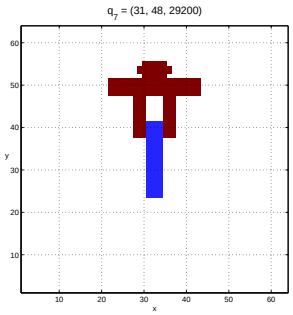
Como entrada al algoritmo 4.2 se utiliza **FA**, zonas de agarre de la pinza, marcadas en



(a) $q_1 = (31, 42, 30800)$ (b) $q_2 = (31, 43, 30200)$ (c) $q_3 = (31, 44, 30200)$

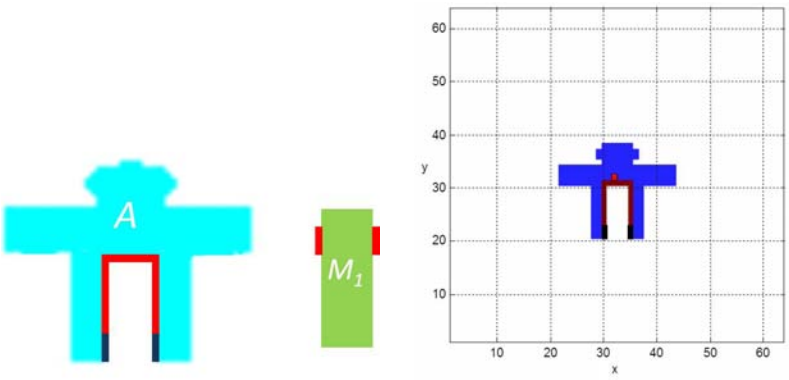


(d) $q_4 = (31, 45, 29800)$ (e) $q_5 = (31, 46, 29600)$ (f) $q_6 = (31, 47, 29400)$



(g) $q_7 = (31, 48, 29200)$

Figura 4.21: Mejores agarres con zona de agarre en pinza y objeto.



(a) (b)

Figura 4.22: Escenario con zonas de agarre en la yema de los dedos de la pinza y en el objeto.

color rojo (zona interior de la pinza) y en negro (yema de los dedos) en la figura 4.22(a). A partir de esta entrada se construye la matriz A^*, donde a las celdas correspondientes a la zona interior de la pinza se les asigna un valor $V_A = longitud(\partial A^*) \bullet k_i$ y para la yema de los dedos se considera otra constante k_p, siendo $k_p > k_i$. Es importante observar que las zonas de agarre en el robot son puntos de su frontera, pero en el entorno son puntos de la frontera exterior de la pieza y pertenecen, por tanto, al entorno aumentado.

Aplicando el algoritmo 4.2 se obtiene la proyección de la pieza en C y el perfil de la frontera de agarre (figura 4.23). Los valores máximos de la frontera de agarre (tabla 4.2) se corresponde con los puntos en que la pinza agarra con los dedos la barra en la zona deseada y en puntos próximos a esta. Como se puede observar en la figura 4.24, los valores de la frontera que reflejan las zonas donde no se produce contacto entre la yema de los dedos y la zona de agarre de la barra no han sido seleccionados, pues no superan el umbral elegido. Así el valor de la frontera q_9 es 25 veces mayor que el de q_{10}.

q_j	∂_g	x	y
q_1	58000	31	48
q_2	48800	31	47
q_3	39600	31	46
q_4	39000	31	49
q_5	31400	31	42
q_6	30800	31	43
q_7	30600	31	44
q_8	30400	31	45
q_9	20000	31	50
q_{10}	800	31	51
q_{11}	400	31	52

Tabla 4.2: Configuraciones de agarre ordenadas de mayor a menor calidad con zonas de agarre en la yema de los dedos de la pinza y en el objeto.

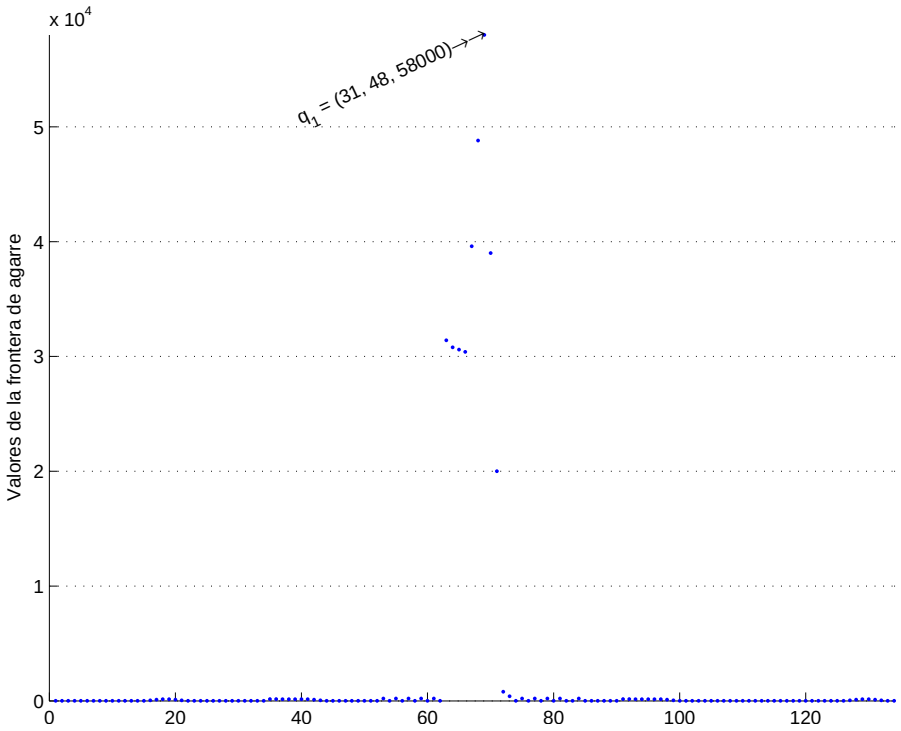


Figura 4.23: Perfil de la frontera de agarre.

Como vemos el hecho de seleccionar en el efector final las zonas con las que se desea agarrar el objeto tiene en nuestro método, al igual que en otros, una gran influencia en la selección de los mejores agarres, pero a diferencia de estos la forma de especificar estas preferencias es más sencilla y no supone añadir mayor complejidad a los cálculos implicados en el proceso.

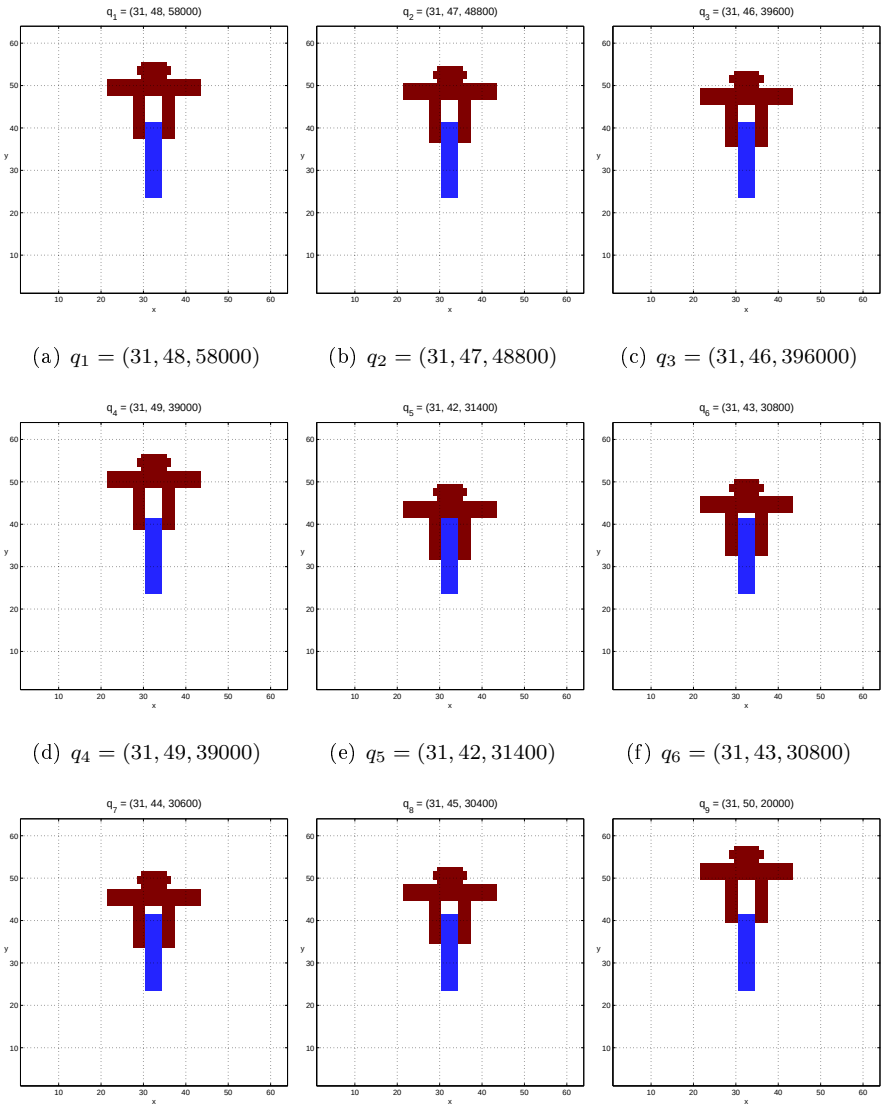
4.4. Algoritmo para una pinza que gira

En este apartado se propondrá un algoritmo para el cálculo de las configuraciones de agarre para un pinza con dos dedos que puede desplazarse y girar libremente en un espacio de trabajo $W = [a, b] \times [c, d] \subset R^2$. En la pinza, con apertura fija, se selecciona una zona de agarre **FA** en la parte interna de sus dedos.

4.4.1. Funciones discretizadas y algoritmo

En W se define un sistema de referencia cartesiano F_W fijo, y además se define un sistema de referencia F_A que se mueve con la pinza (figura 4.25(b)). Una configuración q de A viene parametrizada por (x_r, y_r, θ_r), con $C \subset R^2 \times [-\pi, \pi)$.

Sobre cada coordenada lineal, tanto de W como C, se distribuyen N puntos. Cada celda

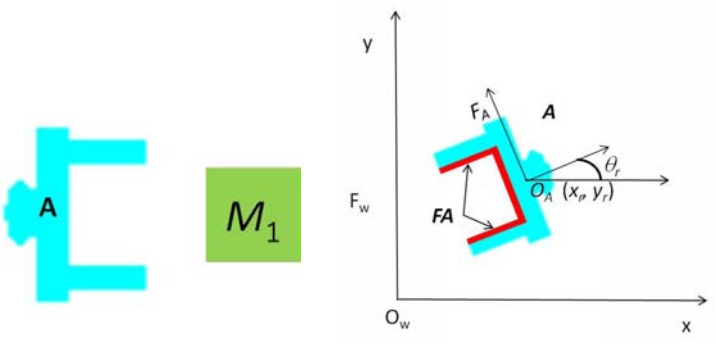


(a) $q_1 = (31, 48, 58000)$ (b) $q_2 = (31, 47, 48800)$ (c) $q_3 = (31, 46, 396000)$

(d) $q_4 = (31, 49, 39000)$ (e) $q_5 = (31, 42, 31400)$ (f) $q_6 = (31, 43, 30800)$

(g) $q_7 = (31, 44, 30600)$ (h) $q_8 = (31, 47, 30400)$ (i) $q_9 = (31, 50, 20000)$

Figura 4.24: Mejores agarres con zona de agarre en el objeto y en la yema de los dedos de la pinza.



(a) Escenario (b) Sistemas de referencia F_W y F_A

Figura 4.25: Pinza paralela A con zonas de agarre y un objeto movible M_1.

espacial estaría dada por la expresión

$$\left[a+i\frac{b-a}{N}, a+(i+1)\frac{b-a}{N}\right] \times \left[c+j\frac{d-c}{N}, c+(j+1)\frac{d-c}{N}\right] \tag{4.12}$$

Las funciones discretizadas E^* que representan al entorno y al *entorno extendido* estarían representadas por dos matrices de tamaño $N \times N$. El elemento de la matriz (i, j) de E^* y AE^* correspondería con la celda espacial previa.

Para la coordenada angular $\theta_r \in [-\pi, \pi)$ se distribuyen M puntos, obteniéndose que la componente m-ésima del vector discretizado resultante sería

$$-\pi + m\frac{2\pi}{M}$$

De esta forma el C-espacio resultante estaría representado por una matriz de tamaño $N \times N \times M$.

Para simplificar el cálculo de las funciones CE y ACE hacemos coincidir el origen de ambos sistemas de referencia con lo que

$$A(x_r, y_r, \theta_r, x, y) = A_{(0,0,\theta_r)}(x - x_r, y - y_r) \tag{4.13}$$

Así la función que define al robot en una configuración determinada (x_r, y_r, θ_r) y en un punto (x, y) de W coincide con el robot en la configuración $(0, 0, \theta_r)$, y donde se ha realizado una traslación de coordenadas. De esta forma la función discretizada que representa a los objetos en el C-espacio discreto vendría dada por

$$CE^*(x_{r_k}, y_{r_l}, \theta_{r_m}) = \sum_{i,j=0}^{N-1} A^*_{(0,0,\theta_{r_m})}(x_i - x_{r_k}, y_j - y_{r_l})E^*(x_i, y_j) \tag{4.14}$$

donde la celda $(x_{r_k}, y_{r_l}, \theta_{r_m})$ de C corresponde con el elemento (k, l, m) de la matriz CE^*. La simplificación proviene de considerar que el cálculo se puede realizar como

$$CE^*(x_{r_k}, y_{r_l}, \theta_{r_m}) = (\bar{A}^*_{(0,0,\theta_{r_m})} * E^*)(x_{r_k}, y_{r_l}) \tag{4.15}$$

donde para cada orientación θ_{r_m} del robot aparece el producto de convolución de la función que describe al robot en la configuración $(0, 0, \theta_{r_m})$ y la que representa a los objetos en W.

Luego, la matriz CE^* es una matriz $N \times N \times M$. Utilizando los índices, la función CE^* se expresaría como

$$CE^*(k, l, m) = (\bar{A}^*_{(0,0,m)} * E^*)(k, l) \tag{4.16}$$

Si el elemento (k, l, m) de esta matriz tienen un valor 0 entonces la celda correspondiente es una celda libre.

Realizando las mismas consideraciones, el cálculo de la matriz ACE^* de tamaño $N \times N \times M$ también se simplificaría mediante

$$ACE^*(k, l, m) = (\bar{A}^*_{(0,0,\theta_m)} * AE^*)(k, l) \tag{4.17}$$

Para cada orientación θ_r se tendría una *frontera de agarre* $\partial_g CE$ que estaría definida por el conjunto:

$$\begin{aligned} \partial_g CE^* = \{(x_{r_k}, y_{r_l}, \theta_{r_m}) \in C, , (CE^*(x_{r_k}, y_{r_l}, \theta_{r_m}) = 0) \ \wedge \\ (ACE^*(x_{r_k}, y_{r_l}, \theta_{r_m}) > 0)\} \end{aligned} \tag{4.18}$$

Por tanto, el conjunto de configuraciones de agarre vendría determinado por

$$q_{grasp} \in \partial_g CE, , ACE(q_{grasp}) = \max_{\theta_{r_m} \in [-\pi, \pi)} (\max_{(x_{r_k}, y_{r_l}) \in \partial_g CE(\theta_{r_m})} (ACE(q))) \tag{4.19}$$

Teniendo en cuenta las expresiones previas el algoritmo que proponemos para calcular las configuraciones de agarre de una pinza que gira es el algoritmo 4.3. En la línea 10 se discretiza la variable angular. Para cada orientación, se calcula la matriz A^*_θ de tamaño $N \times N$, donde se representa a la pinza en cada orientación θ. De igual forma que en el algoritmo 4.2 se obtienen, para cada orientación, las configuraciones cuyo valor de la frontera de agarre supera un determinado umbral u. El conjunto Q_{grasp} contendrá todas las configuraciones de agarre.

Algoritmo 4.3: Obtención de agarres considerando la orientación

Entrada: Matriz del robot (A^*) y el entorno (E^*), zona de agarre de la pinza (**FA**), zona preferida de agarre del objeto (**FM**), factor de agarre (k), umbral (u)

Salida: Configuraciones de agarre (Q_{grasp})

Inicio

$\partial A^* \leftarrow$ Obtener el contorno de A^*;

$V_A \leftarrow longitud(\partial A^*) \bullet k$;

$A^* \leftarrow$ Marcar en la pinza A^* las zonas preferidas de agarre **FA** con el valor V_A;

$\partial {E^*}^C \leftarrow$ Obtener la frontera de la matriz complementaria de E^*;

$AE^* \leftarrow E^* + \partial {E^*}^C$; `/*Obtener el entorno aumentado*/`

$\partial AE^* \leftarrow$ Obtener el contorno de AE^*;

$V_{FM} \leftarrow longitud(\partial AE^*) \bullet k$;

$AE^* \leftarrow$ Marcar en el objeto movible aumentado las zonas preferidas de agarre ;

$AE^* \leftarrow$ Marcar en el objeto aumentado AE^* las zonas preferidas de agarre **FA** con el valor V_{FM} ;

$inc \leftarrow \frac{2*\pi}{M}$; `/*Discretización de la orientación*/`

$Q_g \leftarrow []$; `/*Configuraciones de agarre para una orientación*/`

$Q_{grasp} \leftarrow []$; `/*Configuraciones de agarre*/`

para θ *desde* $-\pi$ *hasta* π *con incremento* $=$ inc **hacer**

  ${A^*}_\theta \leftarrow$ Girar A^* θ grados;

  ${CE^*}_\theta \leftarrow \overline{{A^*}_\theta} * E^*$;

  ${ACE^*}_\theta \leftarrow \overline{{A^*}_\theta} * AE^*$;

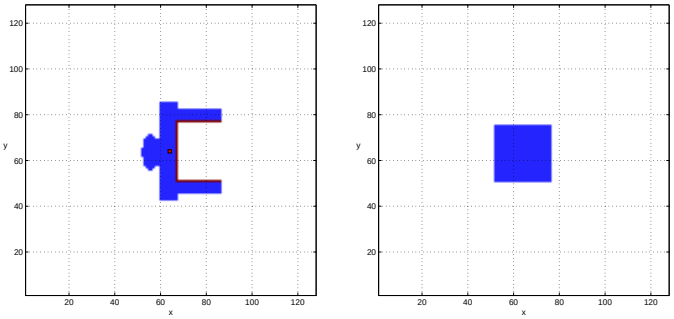
  $\partial_g {CE^*}_\theta \leftarrow$ Obtener la frontera de agarre $(({CE^*}_\theta = 0) \ \wedge \ ({ACE^*}_\theta > 0))$;

  $Q_g \leftarrow$ Obtener las configuraciones con el valor máximo y los máximos locales, por encima de un umbral u, de la frontera de agarre $\partial_g {CE^*}_\theta$ en cada orientación;

  $Q_{grap} \leftarrow Q_g \cup Q_{grasp}$;

4.4.2. Resultados de validación del algoritmo

Para validar el algoritmo planteamos el escenario de la figura 4.26, donde aparece una pinza, en la que se han seleccionado la palma y los dedos como zona de agarre (**FA**); y una pieza que presenta los mejores agarres en cuatro orientaciones diferentes. Se ha elegido una discretización $N = 128$ para las coordenadas espaciales y $M = 128$ para la coordenada angular. Tanto la pinza como la pieza se han representado en dos matrices de 128×128. Por tanto, el C-espacio completo se representa en $M = 128$ matrices de tamaño 128×128.



(a) Pinza A^* con zona de agarre V_A

(b) Objeto movible E^*

Figura 4.26: Pinza y pieza discretizadas del escenario elegido.

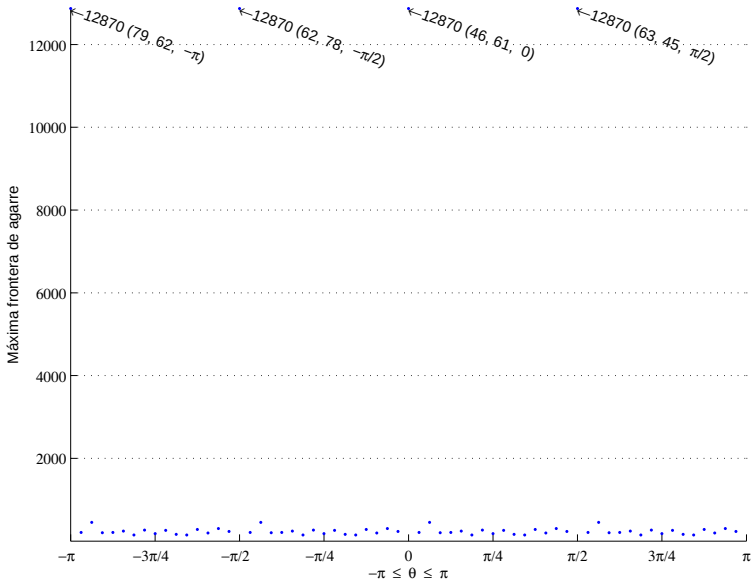
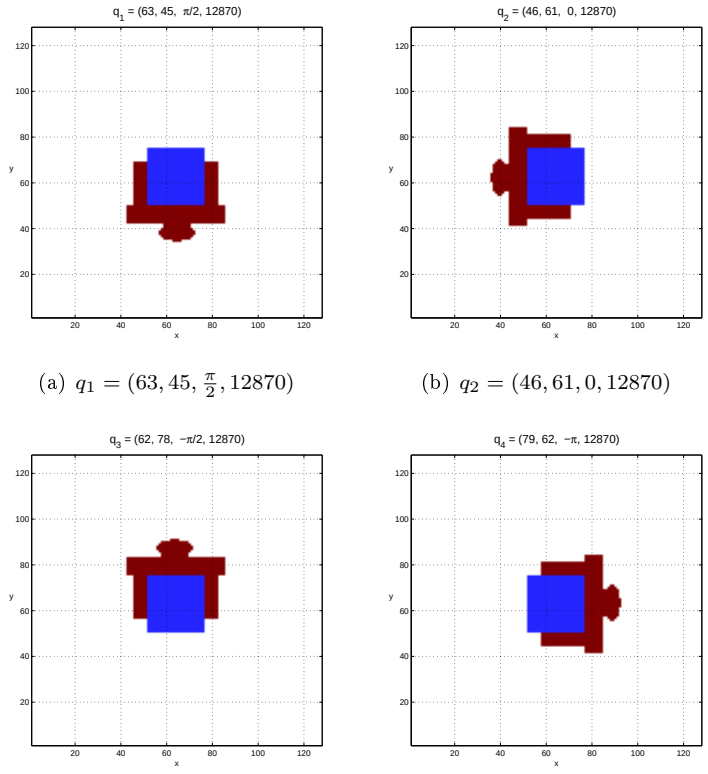


Figura 4.27: Valor máximo de la frontera de agarre para todas las orientaciones.

En la figura 4.27 se representa, para cada una de las orientaciones de la pinza, únicamente el valor máximo de la frontera de agarre. Como puede observarse, aquellas configuraciones

de agarre que producen una mayor longitud de contacto (criterio de *mayor longitud de contacto*) entre la pinza y la pieza corresponden a las orientaciones $[-\frac{\pi}{2}, 0, \frac{\pi}{2}, -\pi]$ de la pinza (figura 4.28). Sin embargo, para cada uno de estas cuatro orientaciones, también se producen máximos locales en su frontera de agarre, que se pueden obtener definiendo un determinado valor umbral u. En la figura 4.29 se muestra el perfil de la frontera de agarre para una orientación $\frac{\pi}{2}$ de la pinza. En este caso si ajustamos el valor umbral a un valor de 2000 seleccionaríamos otras 15 configuraciones de agarre. Estos máximos locales se corresponden a configuraciones donde la pinza agarra al objeto, pero contacta en menos puntos.



(a) $q_1 = (63, 45, \frac{\pi}{2}, 12870)$ (b) $q_2 = (46, 61, 0, 12870)$

(c) $q_3 = (62, 78, -\frac{\pi}{2}, 12870)$ (d) $q_4 = (79, 62, -\pi, 12870)$

Figura 4.28: Mejores agarres considerando la orientación.

Si con la misma pinza deseamos agarrar una pieza en forma de «L» que sólo se puede agarrar en dos orientaciones, al aplicar nuestro algoritmo obtenemos los resultados deseados. Los mejores agarres se obtienen para las orientaciones $-\pi$ y $-\frac{\pi}{2}$ tal y como puede observarse en la figura 4.30. De igual forma que para el caso anterior en ambas orientaciones también se podrían seleccionar otros buenos agarres (los de color rojo en la figura 4.31) ajustando el umbral para seleccionar otros máximos locales que se corresponden con las configuraciones

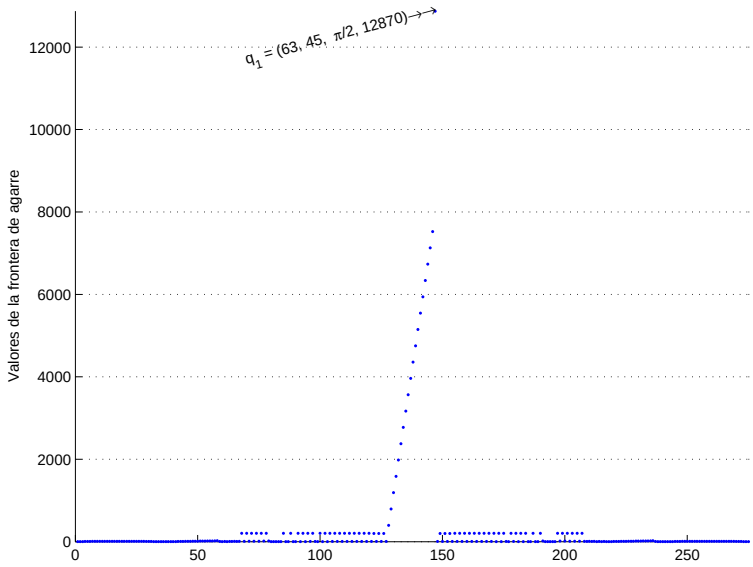


Figura 4.29: Valores de la frontera de agarre para la orientación $\theta = \frac{\pi}{2}$.

donde la pinza agarra la pieza con menos puntos de contacto con el objeto. Finalmente en la figura 4.32 se muestra como la pinza agarra el objeto en las orientaciones $-\pi$ y $-\frac{\pi}{2}$.

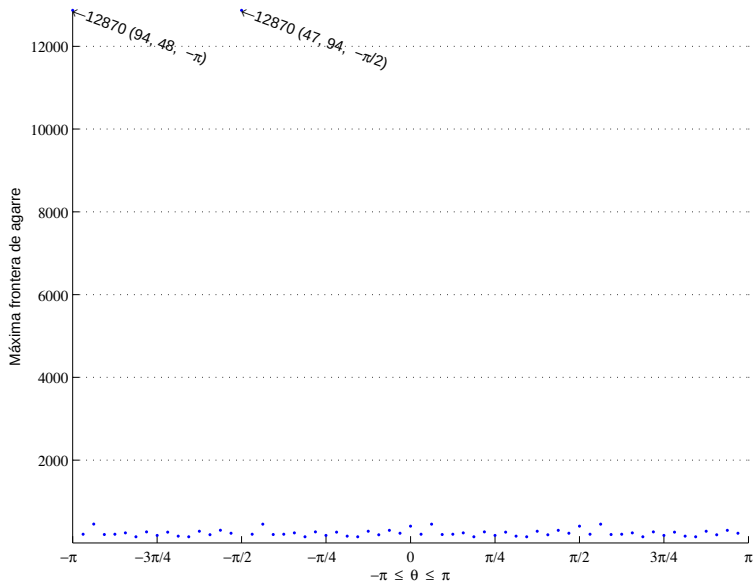
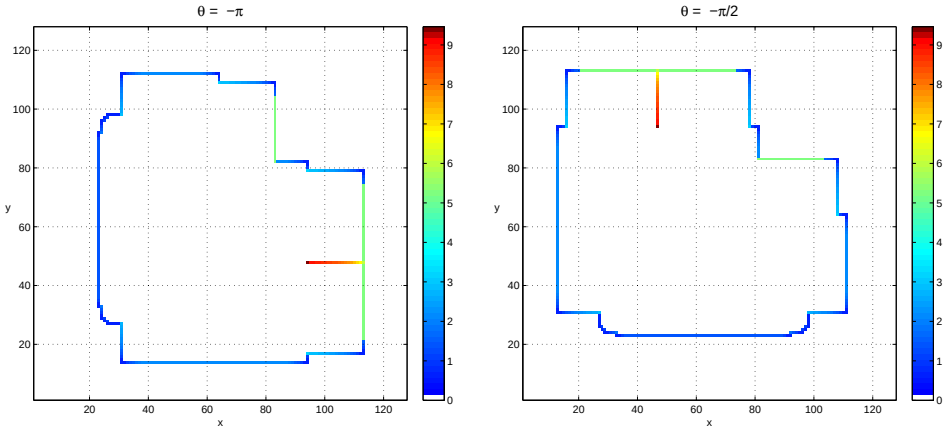
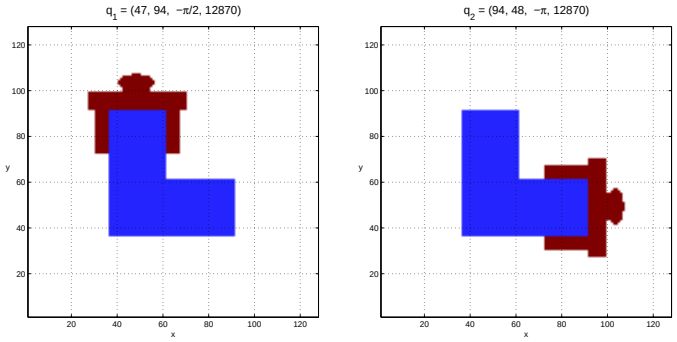


Figura 4.30: Valores máximos de la frontera de agarre en todas las orientaciones.



(a) $\theta = -\pi$ (b) $\theta = -\frac{\pi}{2}$

Figura 4.31: Frontera de agarre en 2D en las orientaciones donde se obtienen las configuraciones de los mejores agarres.

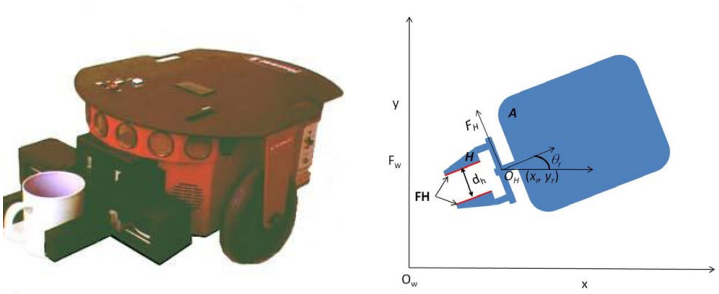


(a) $q_1 = (49, 94, -\frac{\pi}{2}, 12870)$ (b) $q_2 = (94, 48, -\pi, 12870)$

Figura 4.32: Pinza agarrando la pieza en L.

4.5. Algoritmo para un robot con pinza de apertura variable

En esta sección se considerará un robot móvil, como el de la figura 4.33(a), que dispone de una pinza con dos dedos paralelos para agarrar objetos de diferentes dimensiones, variando la apertura de la pinza.



(a) (b)

Figura 4.33: Robot móvil con pinzas paralelas.

4.5.1. Funciones discretizadas y algoritmo

El robot puede desplazarse y girar libremente en un espacio de trabajo $W = [a, b] \times [c, d] \subset R^2$. En la figura 4.33(b) se muestra los sistema de referencia elegidos junto con los parámetros $(x_r, y_r, \theta_r, d_h)$ que definen una configuración: 4 DOFs. Se supone que la figura corresponde a un plano $z = constante$ sobre el suelo.

Sobre el intervalo de definición $[0, d_{max}]$ de la nueva coordenada de configuración d_h se distribuyen uniformemente P puntos, con un incremento $inc_d = \frac{d_{max}}{P}$.

Procediendo de forma similar que para la pinza con apertura fija, la simplificación para el cálculo de CE^* y de ACE^* se realiza mediante las siguientes expresiones

$$CE^*(x_{r_k}, y_{r_l}, \theta_{r_m}, d_{h_p}) = (\bar{A}^*_{(0,0,\theta_{r_m},d_{h_p})} * E^*)(x_{r_k}, y_{r_l}) \tag{4.20}$$

$$ACE^*(x_{r_k}, y_{r_l}, \theta_{r_m}, d_{h_p}) = (\bar{A}^*_{(0,0,\theta_{r_m},d_{h_p})} * AE^*)(x_{r_k}, y_{r_l}) \tag{4.21}$$

donde para cada apertura d_{h_p} y cada orientación θ_{r_m} de la pinza aparece el producto de convolución de la función que describe al robot en la configuración $(0, 0, \theta_{r_m}, d_{h_p})$ y la que representa a la pieza y a la pieza aumentada en W.

Para cada apertura d_{h_p} y cada orientación θ_r se tendría una *frontera de agarre* $\partial_g CE$ que estaría definida por el conjunto:

$$\partial_g CE^* = \{(x_{r_k}, y_{r_l}, \theta_{r_m}, d_{h_p}) \in C, , (CE^*(x_{r_k}, y_{r_l}, \theta_{r_m}, d_{h_p}) = 0) \ \wedge$$

$$(ACE^*(x_{r_k}, y_{r_l}, \theta_{r_m}, d_{h_p}) > 0)\} \tag{4.22}$$

Por tanto, el conjunto de configuraciones de agarre vendría determinado por

$$q_{grasp} \in \partial_g CE, , ACE(q_{grasp}) =$$
$$\max_{d_{h_p} \in [0, d_{max}]} (\max_{\theta_{r_m} \in [-\pi, \pi]} (\max_{(x_{r_k}, y_{r_l}) \in \partial_g CE(d_{h_p}, \theta_{r_m})} (ACE(q)))) \tag{4.23}$$

El algoritmo para calcular las configuraciones de agarre q_{grasp} según las expresiones previas es el algoritmo 4.4. Recibe como entradas las matrices que representan a la pieza E^* y al robot A^* con la pinza, las zonas de agarre en pinza y en pieza, y la apertura máxima de la pinza.

De forma general, para cada una de las aperturas consideradas, se obtiene la matriz A^*_d que representa al robot en cada valor de apertura de la pinza. El robot para cada apertura de la pinza convolucionará con el entorno (línea 19) y con el entorno aumentado (línea 20), en cada una de las orientaciones, según las expresiones 4.20 y 4.21. Se obtendrán las configuraciones de agarre, cuyo valor de la frontera supera un umbral u en cada apertura d_h y cada orientación θ.

4.5.2. Resultados de validación del algoritmo

Para validar el algoritmo propuesto se va a utilizar el robot con la pinza de la figura 4.33(b), donde se ha seleccionado las yemas de los dedos como zona de agarre. Como pieza, se ha elegido una forma geométrica que permite ser agarrada por diferentes zonas, con la pinza en determinadas orientaciones y con diferentes aperturas.

El robot y el objeto se han representado con matrices de 128×128 (figura 4.34). Para cada apertura de la pinza se obtiene una matriz A^* que representa al robot en esa apertura. Para mostrar más claramente los resultados se tendrán en cuenta cuatro aperturas de la pinza: una con valor $d_h = 2$, donde el robot podrá agarrar el objeto por su parte más estrecha; la apertura ($d_h = 12$) en la que el robot agarra el objeto por su parte más ancha; y otras dos con valores ($d_h = 6$) y ($d = 14$) donde el robot no podrá agarrar la pieza (figuras 4.34 y 4.35).

Aplicando el algoritmo a nuestro escenario se obtienen las configuraciones de los mejores agarres correspondientes a los máximos en la frontera de agarre . En la tabla 4.3, la línea roja divide las configuraciones en las que el robot agarra el objeto de las que no. En las configuraciones por debajo de la línea roja, el valor de la frontera de agarre no supera el valor umbral u seleccionado en nuestro algoritmo.

Las configuraciones de agarre, correspondientes a valores máximos en la frontera de agarre, se muestran en la figura 4.36. En estas configuraciones, denotadas por $q_j = (x, y, \theta, d, \partial_g)$,

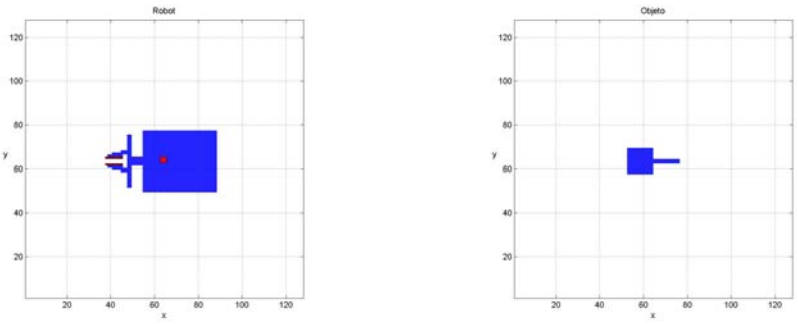
Algoritmo 4.4: Obtención de agarres variando la orientación y apertura de la pinza

Entrada: Matriz del robot (A^*) con la pinza cerrada y el objeto (E^*), zona de agarre de la pinza (**FA**), zona de agarre del objeto (**FM**), factor de agarre (k), umbral (u), máxima apertura (d_{max}) de la pinza

Salida: Configuraciones de agarre

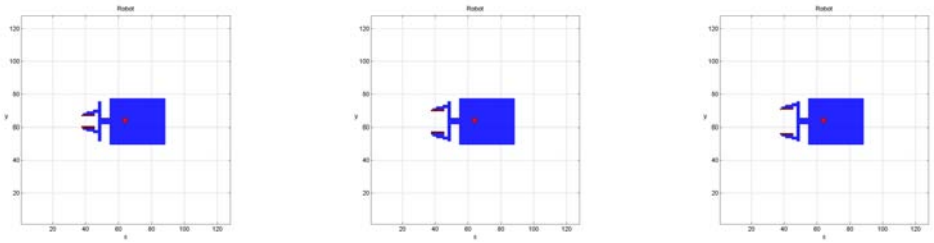
Inicio

- $\partial A^* \leftarrow$ Obtener el contorno de A^*;
- $V_A \leftarrow longitud(\partial A^*) \bullet k$;
- $A^*_0 \leftarrow$ Marcar en la pinza A^* las zonas preferidas de agarre **FA** con el valor V_A;
- $\partial {E^*}^C \leftarrow$ Obtener la frontera de la matriz complementaria de E^*;
- $AE^* \leftarrow E^* + \partial {E^*}^C$; `/*Obtener el objeto aumentado*/`
- $\partial AE^* \leftarrow$ Obtener el contorno de AE^*;
- $V_{FM} \leftarrow longitud(\partial AE^*) \bullet k$;
- $AE^* \leftarrow$ Marcar en el objeto aumentado AE^* las zonas preferidas de agarre **FA** con el valor V_{FM};
- $inc_\theta \leftarrow \frac{2*\pi}{M}$; `/*Discretización de la orientación*/`
- $inc_d \leftarrow \frac{d_{max}}{P}$; `/*Discretización de la apertura de la pinza*/`
- $Q_{g\theta} \leftarrow []$; `/*Configuraciones de agarre parciales en cada orientación*/`
- $Q_{gd} \leftarrow []$; `/*Configuraciones de agarre parciales en cada apertura*/`
- $Q_{grasp} \leftarrow []$; `/*Configuraciones de agarre*/`
- **para** d_h *desde* inc_d *hasta* d_{max} *con incremento* $= inc_d$ **hacer**
  - $A^*_d \leftarrow$ Abrir d_h la pinza de A^*_0 ;
  - $Q_{g\theta} \leftarrow []$; `/*Agarres parciales en todas las orientaciones*/`
  - **para** θ *desde* $-\pi$ *hasta* π *con incremento* $= inc_\theta$ **hacer**
    - ${A^*}_\theta \leftarrow$ Girar A^*_d θ grados;
    - ${CE^*}_\theta \leftarrow \overline{A^*_{d\theta}} * E^*$;
    - ${ACE^*}_\theta \leftarrow \overline{A^*_{d\theta}} * AE^*$;
    - $\partial_g {CE^*}_\theta \leftarrow$ Obtener la frontera de agarre $(({CE^*}_\theta = 0) \ \wedge \ ({ACE^*}_\theta > 0))$;
    - $Q_{\theta_g} \leftarrow$ Obtener las configuraciones con el valor máximo y los máximos locales, por encima de un umbral u, de la frontera de agarre $\partial_g {CE^*}_\theta$ en cada orientación ;
    - $Q_{g\theta} \leftarrow Q_{g\theta} \cup Q_{\theta_g}$ Configuraciones de agarre para cada orientación;
  - $Q_{gd} \leftarrow Q_{gd} \cup Q_{g\theta}$ Configuraciones de agarre para cada apertura;
- $Q_{grasp} \leftarrow Q_{gd}$ Todas las configuraciones de agarre ;



(a) $d = 2$ (b) Objeto $M_1{}^*$

Figura 4.34: Robot con pinza con zona de agarres en la yema de los dedos y en la pieza.



(a) $d = 6$ (b) $d = 12$ (c) $d = 14$

Figura 4.35: Robot con pinza de dedos paralelos en diferentes aperturas.

q	d_h	∂_g	θ	**X**	**Y**
q_1	12	3502	$-\pi$	35	62
q_2	2	3492	0	91	62
q_3	2	3488	0	92	62
q_4	2	3488	0	93	62
q_5	2	3488	0	94	62
q_6	12	3480	$-\pi$	32	62
q_7	12	3480	$-\pi$	33	62
q_8	12	3480	$-\pi$	34	62
q_9	12	3052	$-\pi$	31	62
q_{10}	2	3052	0	95	62
q_{11}	12	2616	$\frac{\pi}{2}$	57	89
q_{12}	12	2616	$-\frac{\pi}{2}$	57	35
q_{13}	12	2616	$-\pi$	30	62
q_{14}	2	2616	0	62	96
q_{15}	14	1758	$-\pi$	35	61
q_{16}	14	1758	$-\pi$	35	63
q_{17}	14	1748	0	91	56
q_{18}	14	1748	0	91	68
q_{19}	6	1748	0	91	60
q_{20}	6	1748	0	91	64

Tabla 4.3: Configuraciones de agarre ordenadas de mayor a menor calidad.

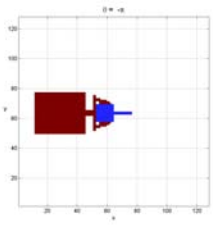
se observa cómo el robot agarra al objeto a lo largo de toda la yema de los dedos (configuraciones con máximos locales de la frontera, correspondientes a agarres en los que la pinza contacta con el objeto con menor longitud de los dedos) en las dos aperturas $d_h = 12$ y $d_h = 2$. Para la apertura $d_h = 12$ aparecen distintas orientaciones, mientras que con $d_h = 2$ solo es posible una orientación. También podemos observar como en las orientaciones $\frac{\pi}{2}$ y $-\frac{\pi}{2}$ para la apertura 12 se seleccionan menos configuraciones de agarre puesto que la pinza no puede agarrar completamente la pieza (no hay contacto con la palma) al golpear con la parte estrecha de la pieza (figuras 4.36(k) y (l)).

4.5.3. Criterio de calidad orientado a la tarea

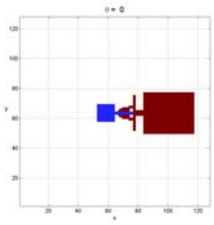
Las configuraciones con una apertura de $d_h = 12$ son configuraciones que presentan el mayor grado de inmovilización del objeto. Si se busca un agarre orientado a la tarea, como sería el caso de que la pieza fuera una herramienta como por ejemplo un mazo y se deseara agarrarlo por el mango, entonces se resaltaría esa zona asignado un V_{FM} superior que al resto de los puntos de la pieza. El robot agarrará el mazo por la zona deseada (figura 4.37(b)) y otros máximos locales en la misma zona (figura 4.37(a)) que se corresponden con el desplazamiento de los dedos a lo largo del mazo. Los agarres de apertura $d_h = 12$ no son seleccionados por no cumplir el criterio de calidad aplicado (los valores de su frontera de agarre no son máximos y quedan muy por debajo del umbral u especificado).

4.6. Accesibilidad: cálculo del C-espacio libre

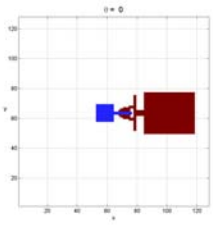
Cuando se plantea un problema de manipulación, además de calcular las configuraciones de agarre, se necesita especificar el camino de tránsito hacia la configuración elegida. Los algoritmos propuestos previamente para calcular las configuraciones de agarre para un robot con una pinza, además proporcionan estructuras de información que pueden utilizarse para conocer si una configuración de agarre es accesible (*reachable*). Mediante el cálculo de la función CB, se proyectan los objetos al espacio de las configuraciones. Por tanto, en la matriz CE^*, que se obtiene en cualquiera de los algoritmos previos, se dispone de información sobre las configuraciones libres de colisiones. Estas corresponderían a aquellos elementos de la matriz CE^* con un valor nulo. Por tanto, un planificador que calcule el camino desde una determinada configuración a la configuración de agarre, podría utilizar esta estructura de datos para validar una configuración (test de colisiones), sin más que consultar si el elemento correspondiente de la matriz tiene un valor nulo. Luego, con la matriz CB^* se puede comprobar la accesibilidad a la configuración de agarre calculada.



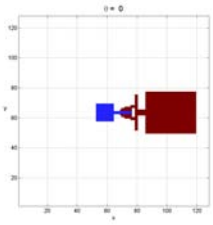
(a) $q_1 = (35, 62, -\pi, 12, 3502)$



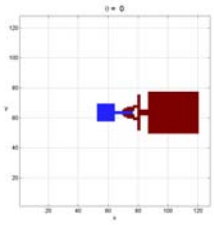
(b) $q_2 = (91, 62, 0, 2, 3492)$



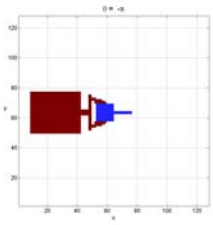
(c) $q_3 = (92, 62, 0, 2, 3488)$



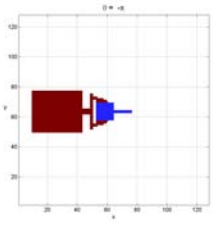
(d) $q_4 = (93, 62, 0, 2, 3488)$



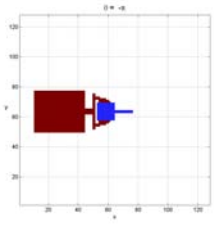
(e) $q_5 = (94, 62, 0, 2, 3488)$



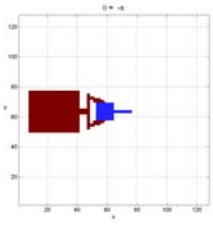
(f) $q_6 = (32, 62, -\pi, 12, 3480)$



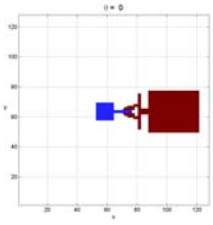
(g) $q_7 = (33, 62, -\pi, 12, 3480)$



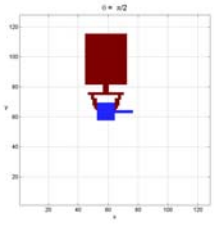
(h) $q_8 = (34, 62, -\pi, 12, 3480)$



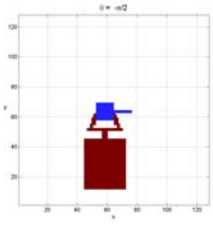
(i) $q_9 = (31, 62, -\pi, 12, 3052)$



(j) $q_{10} = (95, 62, 0, 2, 3052)$

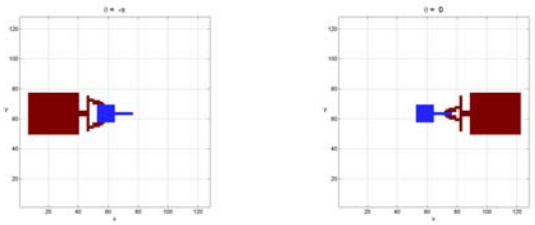


(k) $q_{11} = (57, 89, \frac{\pi}{2}, 12, 2616)$



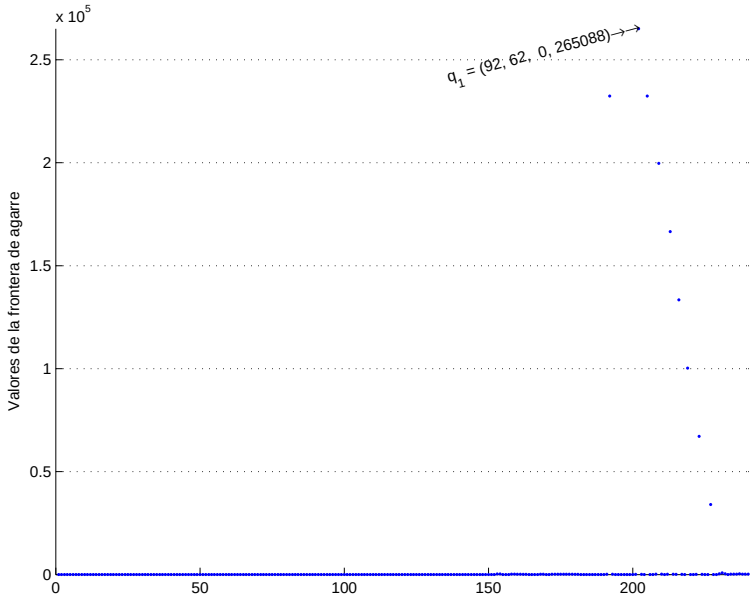
(l) $q_{12} = (30, 62, -\frac{\pi}{2}, 12, 2616)$

Figura 4.36: Robot en configuraciones de agarre (continua).

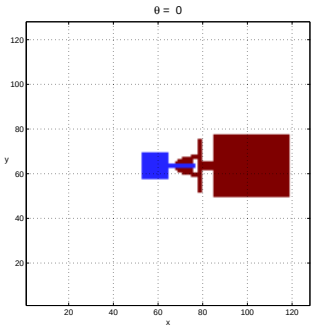


(m) $q_{13} = (30, 62, -\pi, 12, 2616)$ (n) $q_{14} = (62, 96, 0, 2, 2616)$

Figura 4.36: Robot en las configuraciones de agarre (continuación).



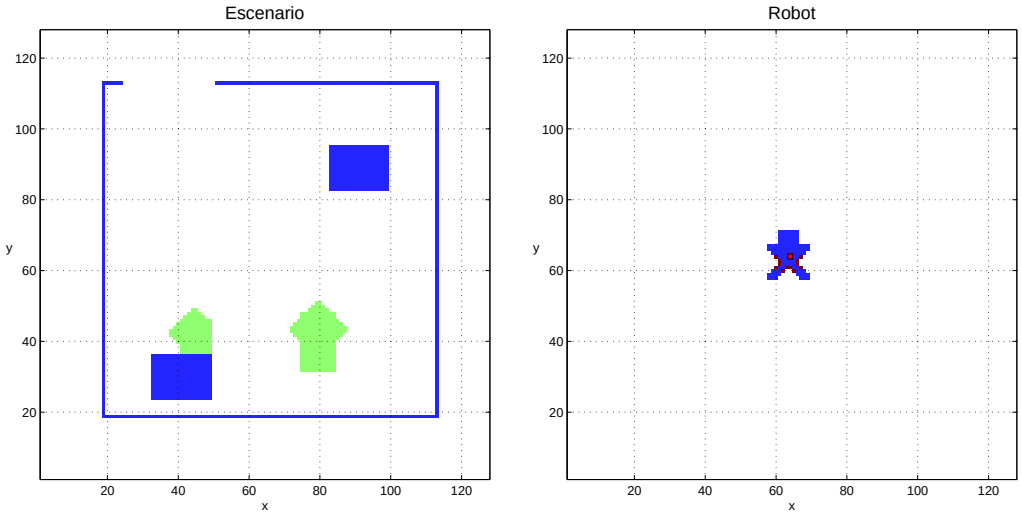
(a) Perfil de la frontera de agarre para $d_h = 2$ y $\theta = 0$



(b) Configuración de agarre

Figura 4.37: Agarrando por el mango.

A continuación, se va a plantear un entorno con objetos fijos (obstáculos) y objetos que el robot puede agarrar (objetos movibles). Como escenario de trabajo se plantea utilizar el que aparece en [Lat91b] (Figura 4.38). En azul aparecen los obstáculos y en verde los objetos movibles. El robot se puede desplazar y girar por un espacio de trabajo $W \subset R^2$. Se ha seleccionado la zona en la frontera de A^* que modela la zona de la pinza que se desea contacte con los objetos movibles. Como consecuencia se utilizará el algoritmo 4.3, pero en este caso la matriz E^* representará un entorno en el que puntos $(x, y) \in W$ ocupados por los obstáculos fijos se representan por un 1 y los ocupados por los objetos que puede manipular el robot (objetos movibles) por un valor superior a 1. De esta forma se puede diferenciar entre ambos tipos de objetos. El espacio de trabajo se ha discretizado en una matiz de 128×128, mientras que en la orientación θ_r se ha considerado una discretización $M = 128$.



(a) Escenario (b) Robot con zonas de agarre

Figura 4.38: Robot con zonas de agarre en un escenario con objetos movibles y obstáculos.

En la línea 15 del algoritmo 4.3, se proyectan los objetos del espacio de trabajo W al C-espacio, mediante la convolución de la matriz que representa el entorno E^* y la que representa al robot A^* en cada una de las orientaciones. El C-espacio resultante estaría representado en 128 matrices de tamaño 128×128. En las figuras 4.39, 4.40 , 4.41 y 4.42 se presenta el C-espacio para las orientaciones $\theta_r = -\pi, -\frac{\pi}{2}, 0, \frac{\pi}{2}$.

Procediendo de igual forma que en las secciones previas para el cálculo de las configuraciones de agarre, en la figura 4.43 se representan los valores máximos en la frontera de agarre en cada una de las orientaciones. Para nuestro escenario los máximos se alcanzan en $\theta_r = -\pi, -\frac{\pi}{2}, 0, \frac{\pi}{2}$.

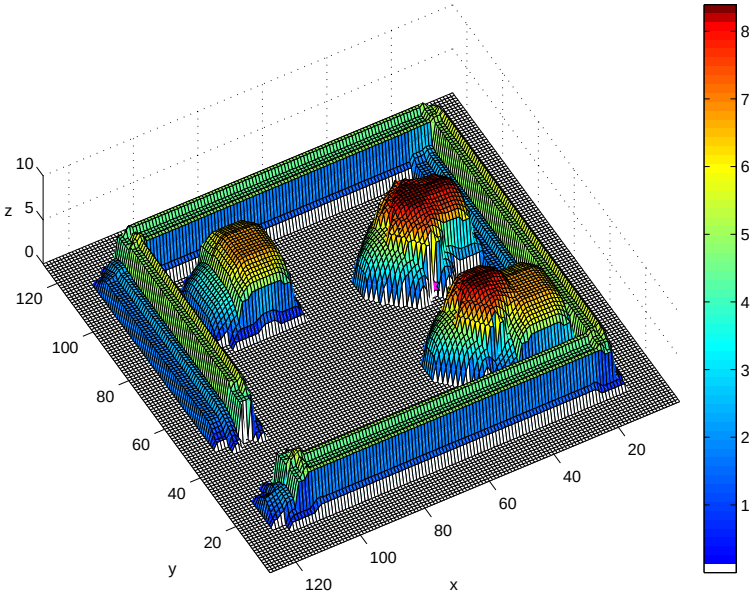


Figura 4.39: C-espacio en la orientación $\theta = -\pi$.

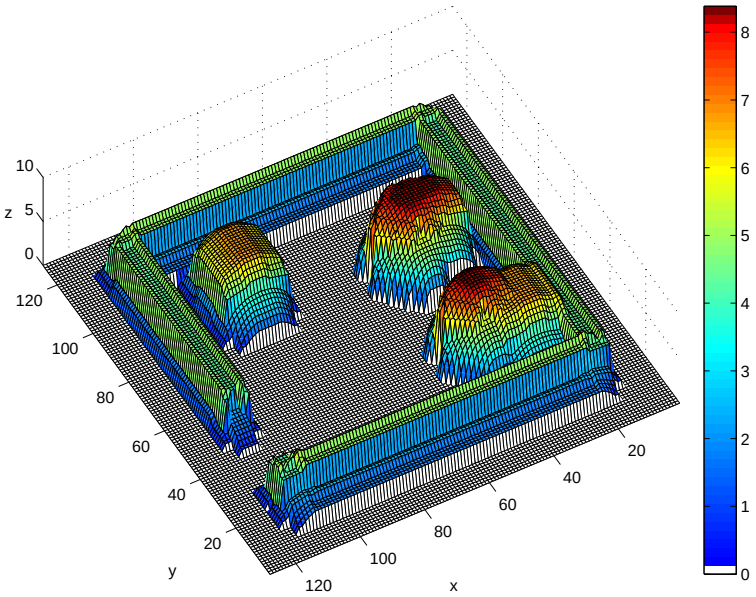


Figura 4.40: C-espacio en la orientación $\theta = -\frac{\pi}{2}$.

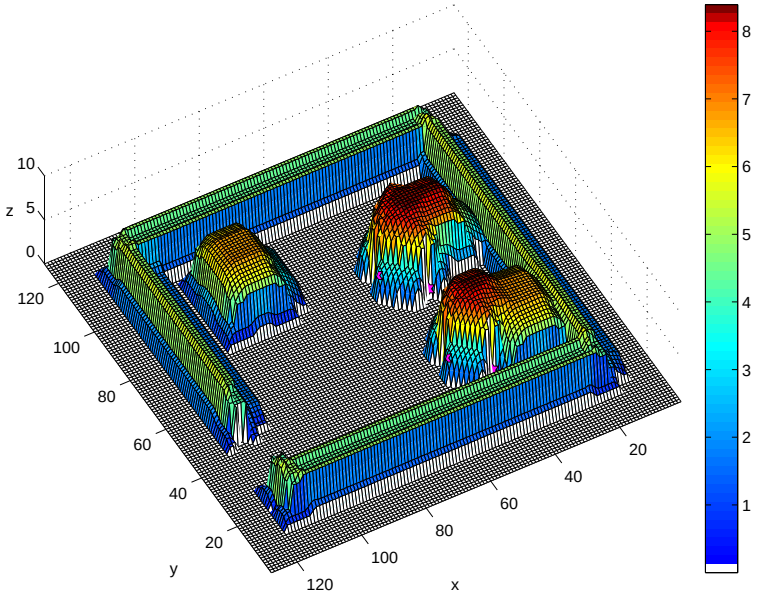


Figura 4.41: C-espacio en la orientación $\theta = 0$.

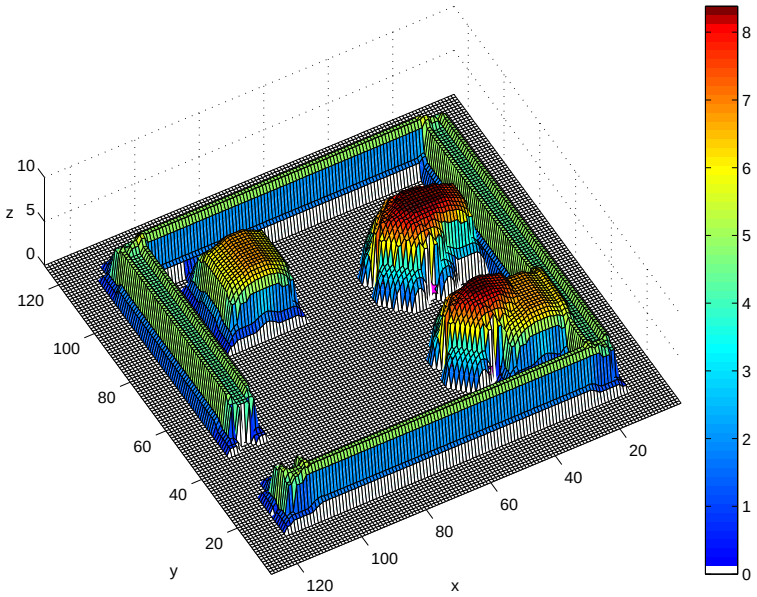


Figura 4.42: C-espacio en la orientación $\theta = \frac{\pi}{2}$.

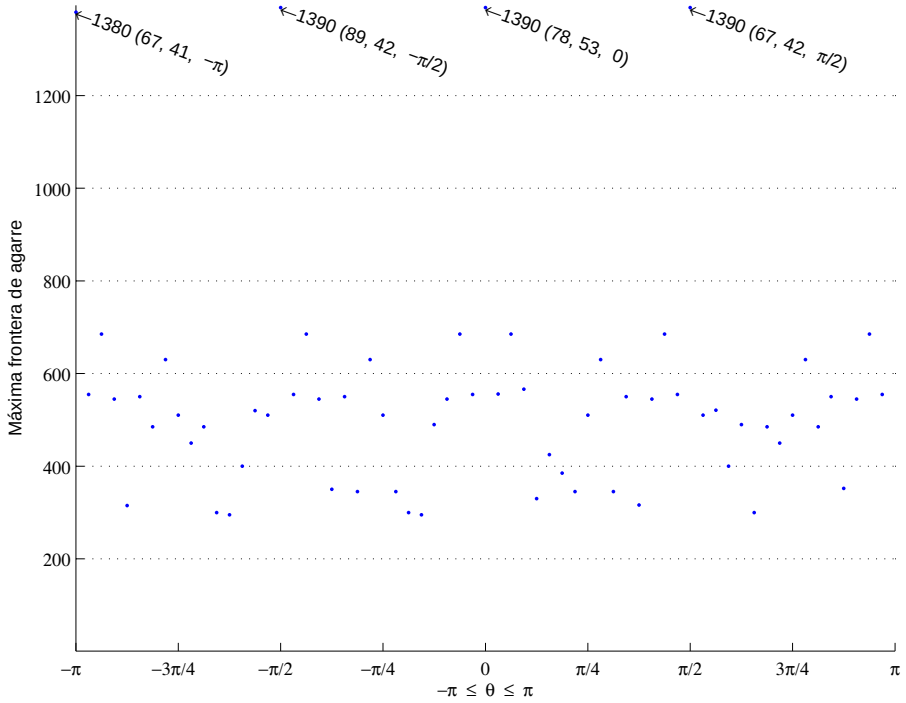
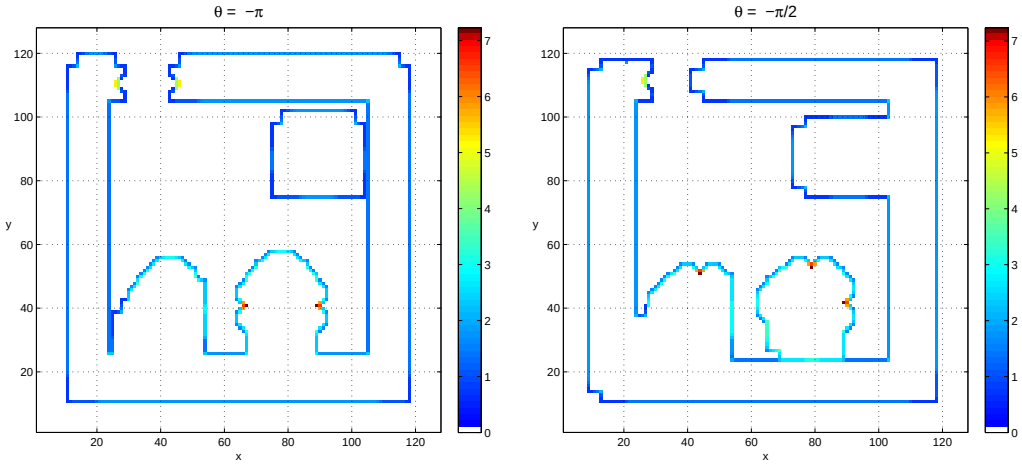


Figura 4.43: Valores máximos de la frontera de agarre en todas las orientaciones.

En la figura 4.44 se muestra en 2D la frontera de agarre para cada una de las cuatro orientaciones donde se han obtenido los máximos de la frontera. En cada una de las cuatro orientaciones aparecen máximos absolutos y máximos locales. Las respectivas configuraciones forman el conjunto G_M, que son las configuraciones con máxima longitud de contacto y, por tanto, los mejores agarres (figura 4.45).

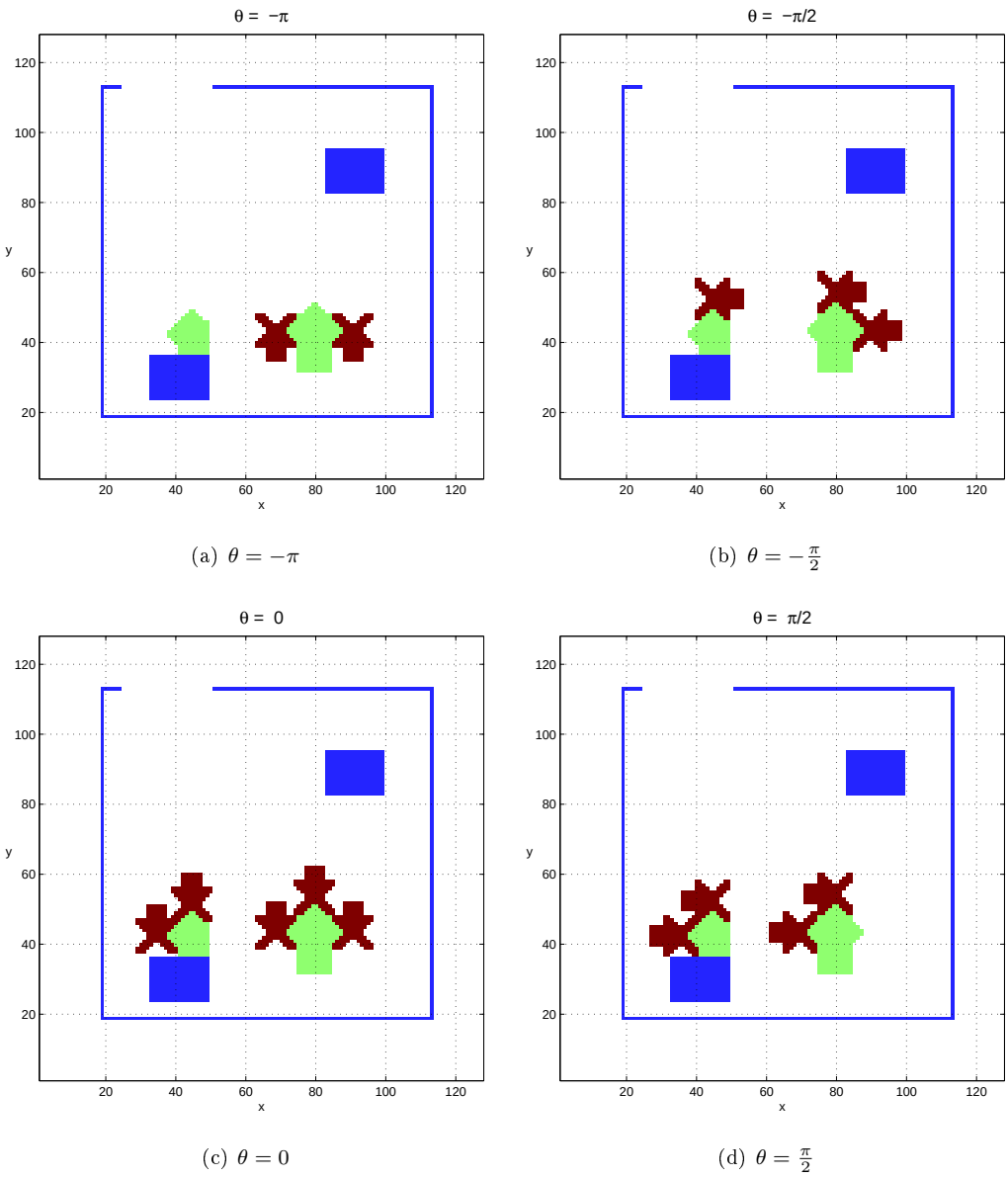
Hemos puesto de manifiesto que en nuestro método la forma de proyectar los objetos movibles en el espacio de las configuraciones se realiza de una manera sencilla, asignando a estos objetos en la matriz del entorno un valor mayor que 1 para así diferenciarlos de los obstáculos. Esto permitirá que los valores de la frontera de agarre en el espacio de las configuraciones sean mayores y, por tanto, sea en el objeto movible donde se seleccionen los agarres. Además, dentro del mismo algoritmo calculamos una estructura de datos que contiene información para determinar la accesibilidad a cada uno de los agarres de una forma simple.



(a) $\theta = -\pi$ (b) $\theta = -\frac{\pi}{2}$

(c) $\theta = 0$ (d) $\theta = \frac{\pi}{2}$

Figura 4.44: Frontera de agarre en 2D del escenario de la figura 4.38 en las orientaciones donde se obtienen las configuraciones de los mejores agarres.



(a) $\theta = -\pi$ (b) $\theta = -\frac{\pi}{2}$

(c) $\theta = 0$ (d) $\theta = \frac{\pi}{2}$

Figura 4.45: Escenario con el robot en las configuraciones de agarre.

4.7. Análisis de la complejidad de los algoritmos propuestos

En el capítulo 3 se han obtenido las expresiones del C-espacio $CE(q)$ y del C-espacio aumentado $CEA(q)$. En el cálculo de CE aparece el producto de convolución de A y E sobre unas ciertas variables. De igual forma para el cálculo de $ACE(q)$ donde el producto de convolución se realiza entre A y AE. En sus correspondientes expresiones discreta y particularizadas para cada robot analizado, en las expresiones de $CE^*(q_j)$ y de $ACE^*(q_j)$, surge la convolución discreta de A^* y E^* sobre dichas variables discretizadas para las expresiones de CE^* y de igual forma para A^* y AE^* en el cálculo de ACE^*. Puesto que la convolución se realiza dos veces, una con el entorno y otra con el entorno aumentado, el coste computacional de los algoritmos analizados habrá que multiplicarlo por dos.

Si se implementa directamente la convolución de las matrices A^* y E^*, la complejidad computacional del algoritmo es $O(N^{d^2})$, siendo N la resolución de la discretización utilizada y d la dimensión de la convolución.

Otra opción sería aplicar el teorema de convolución calculando la Transformada de Fourier Discreta (DFT) de $CE^*(q_j)$ como el producto punto a punto de la DFT de A^* y la de E^*, efectuando las transformadas sobre una o varias variables. La dimensión de la Transformada Discreta de Fourier y las variables concretas sobre las que se efectúa depende del tipo de robot considerado. Mediante el algoritmo de la Transformada Rápida de Fourier (FFT), cuya complejidad es $O(N^d \log N)$, se conseguiría reducir la carga asociada a la ejecución de los algoritmos. Para aplicar el algoritmo de FFT las funciones deben de ser periódicas. Sin embargo, las matrices binarias implicadas no son cíclicas. Una forma de solucionar este problema, [Kav95], es colocar un '1' en todos los elementos del contorno de la matriz binaria que representa al espacio de trabajo E^*. Así, además de conseguir funciones cíclicas, se asegura que el robot no pueda sobrepasar los límites de W.

En primer lugar analizaremos la complejidad computacional del algoritmo básico (algoritmo 4.1) de obtención de agarres donde se considera un robot y un objeto representado en el plano donde el robot se desplaza pero no puede girar. Si las variables lineales se discretizan con una resolución N, se dispone de una matriz E^* de tamaño $N \times N$, una matriz $\bar{A}^*$ de tamaño $N \times N$ y, como resultado final del algoritmo, una matriz de dimensión $N \times N$. Si se realiza como un producto de convolución la complejidad será $O(N^4)$. Utilizando la FFT, como la FFT es de dos dimensiones, la complejidad computacional del algoritmo es $O(N^2 \cdot \log(N))$.

Para el algoritmo (algoritmo 4.3) donde se añade un grado de libertad que es la orien-

tación, si las variables lineales se discretizan con una resolución N mientras que la variable angular θ_r con una resolución M, se dispone de una matriz E^* de tamaño $N \times N$, una matriz $\bar{A}^*$ de tamaño $N \times N$ para cada orientación del robot y, como resultado final del algoritmo, M matrices de dimensión $N \times N$. Se utiliza la técnica de *slicing*[1] sobre el grado de libertad, θ_r, donde no se produce convolución. Con el producto de convolución tendremos un coste computacional de $O(M \cdot N^4)$ mientras que como las FFT son en dos dimensiones la complejidad computacional del algoritmo es $O(M \cdot N^2 \cdot \log(N))$.

En el caso del robot que se mueve libremente en el espacio, puede girar y dispone de una pinza que puede variar su apertura (algoritmo 4.4) las distintas aperturas de la pinza se discretizan con una resolución P y la angular θ_r se discretiza con una resolución M. En este caso es necesario construir P matrices binarias $\bar{A}^*$ de tamaño $N \times N$ para las distintas aperturas de la pinza y, para cada orientación del robot, otras M matrices binarias $\bar{A}^*$ de tamaño $N \times N$. El resultado del algoritmo son $P \times M$ matrices binarias CE^* de tamaño $N \times N$. Se realizan por tanto $P \times M$ productos de convolución en dos dimensiones siendo su complejidad computacional $O(P \cdot M \cdot N^4)$. De igual forma se realizan $P \times M$ transformadas de Fourier en dos dimensiones de matrices de tamaño $N \times N$, entonces la complejidad computacional del algoritmo es $O(P \cdot M \cdot N^2 \cdot \log(N)$.

Es importante recordar que una diferencia fundamental que aparece con respecto de los métodos algebraicos para el cálculo del C-espacio, es que ahora el tiempo de cálculo únicamente depende de la resolución empleada para la representación, y no depende en absoluto de la forma de los obstáculos o del robot. La desventaja de los métodos discretos en general y del método expuesto en particular por el excesivo coste computacional se reduce drásticamente con la evaluación a través de la FFT.

Para la evaluación rápida de nuestros algoritmos habría que sustituir en todos ellos el producto de convolución de A^* y E^* por la aplicación del teorema de convolución como aparece en las líneas 4 a 10 del algoritmo 4.5. En el algoritmo 4.5 se muestran las modificaciones realizadas sobre el algoritmo básico para una evaluación más rápida de nuestros algoritmos. En el resto de los algoritmos se procedería de igual forma.

4.8. Aportaciones

En este capítulo se han descrito los algoritmos desarrollados que permiten validar el formalismo propuesto para obtener el conjunto de configuraciones de agarre considerando diferentes tipos de pinzas y grados de libertad. Se ha mostrado cómo los algoritmos son

[1] Puesto que sobre la orientación no se produce convolución, se puede discretizar con un número de puntos diferente M al utilizado en las variables sobre las que si se produce la convolución.

Algoritmo 4.5: Algoritmo básico de obtención de agarres con FFT

Entrada: Matriz del robot (A^*), matriz del entorno (E^*)

Salida: Configuraciones de agarre (Q_{grasp})

1 **Inicio**

2 $\partial {E^*}^C \leftarrow$ Obtener la frontera de la matriz complementaria del entorno;

3 $AE^* \leftarrow E^* + \partial {E^*}^C$; `/*Obtener el entorno aumentado*/`

4 $\mathcal{F}[E^*] \leftarrow$ Obtener la transformada de Fourier del entorno;

5 $\mathcal{F}[\bar{A}^*] \leftarrow$ Obtener la transformada de Fourier $\bar{A}^*$;

6 $P = \mathcal{F}[E^*] \cdot \mathcal{F}[\bar{A}^*]$;

7 $CE^* = \mathcal{F}^{-1}[P]$ Obtener la transformada inversa;

8 $\mathcal{F}[AE^*] \leftarrow$ Obtener la transformada de Fourier del entorno aumentado;

9 $P = \mathcal{F}[AE^*] \cdot \mathcal{F}[\bar{A}^*]$;

10 $ACE^* = \mathcal{F}^{-1}[P] \leftarrow$ Obtener la transformada inversa;

11 $\partial_g CE^* \leftarrow$ Obtener la frontera de agarre $((CE^* = 0) \ \wedge \ (ACE^* > 0))$;

12 $Q_{grasp} \leftarrow$ Configuraciones con valor máximo en la frontera de agarre;

independientes de la forma de la pieza y de la geometría de la pinza y dependen solo del tamaño de las matrices de discretización con las que se trabaje. Además, se ha propuesto un criterio de calidad para conseguir inmovilizar la pieza (agarre de fuerza) y otro criterio orientado a la tarea (agarre de precisión) que no incrementa la complejidad de los algoritmos. Por último, situando nuestra propuesta en un escenario con la presencia de obstáculos y objetos que pueden ser agarrados por el robot (objetos movibles), se ha puesto de manifiesto que en los algoritmos se dispone de una estructura de datos para examinar la accesibilidad a las configuraciones de agarre calculadas. De esta forma hemos demostrado como nuestro método de síntesis de agarres permite obtener agarres accesibles ordenados de mayor a menor calidad de acuerdo al criterio de calidad elegido (orientado a inmovilizar la pieza o a la tarea que posteriormente se desea realizar con ella).

Aunque los ejemplos se han planteados con pinzas simples, hay que tener en cuenta que muchos trabajos con pinzas antropomórficas, como las manos humanas, reducen la complejidad de los métodos de análisis y síntesis a considerar dos dedos virtuales ([Ibe87] [Ibe97]) que, posteriormente, se hacen corresponder con los dedos reales. Este sería el caso de la pinza de la figura 4.11(a), donde se podría considerar que hay dos articulaciones en cada uno de los dedos. Variando el grado de libertad asociado a cada articulación, se podría representar el robot A en un conjunto de matrices, que serían la entrada a los algoritmos previos de forma que permitan obtener el conjunto de configuraciones de agarre.

5

Planificación de manipulaciones de un robot paletizador

5.1. Los vehículos guiados automáticamente

Actualmente para lograr aumentar la competitividad, la industria está adoptando Sistemas de Fabricación Flexible (FMS - *Flexible Manufacturing Systems*) como respuesta rápida a cambios en las necesidades de producción. En estos sistemas, además de las máquinas herramientas, los robots manipuladores, etc., se han ido incorporando los Vehículos Guiados Automáticamente (AGV - *Automated Guided Vehicle*) que se dedican al transporte de materiales con el objetivo de ahorrar costes y mejorar la eficiencia.

Estos vehículos basados en computador siguen rutas preestablecidas de manera repetitiva sin la intervención de un conductor. Desde los primeros sistemas filoguiados, se han ido mejorando sus prestaciones hasta cubrir actualmente una cuota de mercado. Entre las soluciones comerciales aparecen el sistema E'gv de la empresa Egemin (`http://www.egemin.com`), el sistema *Skilled LGV* de la empresa Euroimpianti (`http://www.skilled.it`), Odyssey de

AGV Products (http://www.agvp.com) o el sistema P3 de la empresa Handyman (http://www.carltonhandyman.com/p3), entre otros.

Los AGVs presentan un mayor grado de flexibilidad respecto de las cintas de transporte instaladas, que están gobernadas por sistemas basados en autómatas industriales. Los primeros sistemas de guiado basados en un cable enterrado en el suelo o una banda magnética adherida al suelo presentan limitaciones en cuanto a que las rutas seguidas por los vehículos están preestablecidas: la navegación se limita a recorridos fijos, y por tanto a tareas repetitivas.

Para mejorar la flexibilidad, se han desarrollado sistemas de guiado basados en la colocación de balizas activas y pasivas en el entorno. Entre las primeras, se encuentran las balizas de infrarrojos o ultrasonidos, mientras que entre las pasivas están los puntos magnéticos o reflectores para el guiado por láser. Aunque en estas soluciones comerciales es más sencillo modificar las rutas, sin embargo necesitan alterar el entorno y trabajar en un entorno conocido y estructurado.

Además de las plantas de fabricación y de almacenaje especialmente diseñadas y acondicionadas para integrar los AGVs, existen otros tipos de instalaciones donde los AGVs pueden resultar eficaces. Se trata de entornos parcialmente desconocidos y donde las condiciones de trabajo presentan determinadas incertidumbres. Por ello, los trabajos de investigación se están dirigiendo a sistemas de guiado basados en la detección de características bien artificiales, como patrones gráficos, o bien naturales, como columnas, para detectarlos con sistemas de visión. El primer caso supone también una alteración del entorno de trabajo, mientras que en el segundo surge el problema de la asociación de datos.

Si el AGV se integra realmente en un FMS no ideal no sólo hay que considerar la localización del AGV dentro del entorno (el sistema de guiado), sino que también deben contemplarse los aspectos relacionados con la manipulación de objetos no preestablecida. Es por lo que en este capítulo planteamos el siguiente contexto de trabajo, como un intento de flexibilizar también la tarea de manipulación de un AGV.

5.1.1. Planteamiento propuesto para la planificación de manipulaciones

Supongamos un escenario como el que se representa en la figura 5.1, donde existen una serie de objetos presentes en su entorno de trabajo, como columnas, estanterías, etc., con los que el robot puede colisionar y otra serie de objetos movibles, como palets, cajas, etc., que el robot puede trasladar.

Es este escenario donde el robot deberá hacer de forma autónoma la manipulación de

Figura 5.1: Posible entorno de trabajo de un vehículo guiado automáticamente (AGV).

objetos con la mínima intervención del usuario. Más concrétamente, la descripción de la tarea sería: "Coge el objeto X y depositalo en la posición Y". Para ello, el robot tiene que identificar estos objetos, decidir cómo agarrarlos, llevar al robot a la posición de agarre sin colisionar con ninguno de ellos y finalmente trasladar el objeto movible, ya agarrado por el robot, a su posición final. Para realizar esta tarea de forma autónoma o semiautónoma se necesita resolver un conjunto de problemas que comentamos a a continuación junto con el planteamiento propuesto en esta tesis para resolverlos.

El paso crucial que aborda este trabajo de tesis ya expuesto en los capítulos anteriores es la síntesis de agarres, que consiste en determinar las configuraciones de agarre, considerando el objeto a transportar y el robot. Para ello necesitamos una descripción geométrica del robot y del objeto que se traducirá a mapas de bits. Para determinar el conjunto de las configuraciones de agarre se tendrán en cuenta criterios de calidad dirigidos a inmovilizar el objeto y otros orientados a la tarea.

Una vez determinado el conjunto de los mejores agarres se necesita disponer de la posición del robot y de los objetos. Este tema queda fuera del alcance de esta tesis doctoral por lo que suponemos que algún otro procedimiento responde a las preguntas ¿dónde estoy? y ¿donde están los objetos a manipular?.

Otro problema a abordar consiste en determinar el camino/trayectoria de tránsito, por lo que hay que tener en cuenta la descripción cinemática del robot. En nuestro caso, el AGV será una carretilla transportadora, cuya cinemática normalmente será de tipo Ackerman, por lo que está sujeto a restricciones no holónomas en sus movimientos. La búsqueda de métodos generales que permitan modelar la cinemática y la dinámica del movimiento de los robots tipo coche es aún un tema de investigación abierto. Uno de los enfoques consiste

en restringir su conjunto de movimientos a los especificados por unas pocas primitivas de movimiento (Ej. girar a la derecha, ir recto, girar a la izquierda). En nuestro planteamiento de resolución, el cálculo de estas trayectorias de tránsito estará apoyado en las curvas de Dubins. Estas curvas definen la trayectoria óptima, entre una configuración inicial y otra final del robot, con restricciones en cuanto a velocidad constante y radio de giro máximo. Las trayectorias calculadas son óptimas en el dominio temporal, que al trabajar a velocidad constante garantiza que los caminos calculados son también óptimos.

También, se necesita verificar si las rutas de tránsito calculadas están libres de colisiones. Recordemos que, como parte de nuestro método de obtención de los mejores agarres se obtiene también una representación de los objetos en el espacio de las configuraciones del robot. De esta forma el test de colisiones se simplifica enormemente permitiendo descartar las rutas que produzcan colisión.

Una vez que el robot agarre el objeto movible lo traslada, siguiendo un camino de transferencia, a su posición final. Para determinar este camino se calcularán de nuevo las curvas de Dubins entre la posición en la que el robot ha agarrado al objeto y el destino final. Una vez calculadas se evaluará las más cortas y si son o no factibles. Para este último paso será necesario considerar la representación de los objetos en el espacio de las configuraciones, pero no únicamente del robot sino considerando el robot junto con el objeto que transporta. Este planteamiento general será descrito en las siguientes secciones.

5.2. El robot paletizador y su entorno

Para la realización de las tareas de paletización (cargar mercaderías sobre un palet) se suele utilizar una carretilla elevadora. El grupo de Robótica y Sociedad de la Universidad de Salamanca ha robotizado recientemente un vehículo de transporte, por lo que se utilizará como modelo para realizar pruebas de validación de nuestros algoritmos.

La carretilla elevadora puede trabajar tanto en entornos interiores, como pueden ser almacenes, naves, etc., como en exteriores, manejando palés. La necesidad de estandarizar la forma y tamaño de los palets resulta evidente para su fabricación a gran escala, su transporte o en el diseño de herramientas de manipulación, entre las que se encuentran los robots paletizadores.

Existen numerosos organismos internacionales que regulan y certifican las normas relacionadas con los palés que especifican las dimensiones de todas sus piezas y su montaje, peso que pueden soportar, condiciones mínimas para su reutilización, reciclado, etc. Uno de los tipos de palés mas utilizados es el palé Europeo o EUR EPAL (*European Pallet Aso-*

ciation - http://www.epal-pallets.org) definido en las normas de la Unión Europea de Ferrocarriles (UIC - *Union Internationale des Chemins de Fer*) UIC 435-2-0 y UIC 435-4-0 R.

Como se ha comentado queda fuera del alcance de esta tesis doctoral responder a la pregunta ¿donde está el palé a manipular?; pero como vemos, este sector está perfectamente regulado por lo que no es de extrañar que en un futuro muy cercano se puedan etiquetar los palés con marcas estándares reconocibles por los AVGs que permitan responder automáticamente a esta pregunta.

5.3. Síntesis de agarre para un AGV

En este apartado se va a aplicar nuestro método de síntesis de agarres descrito en los capítulos anteriores para determinar el conjunto de los mejores agarres con que nuestro robot paletizador A ha de manipular un palé europeo M_1 (figura 5.2(a)) y posteriormente situaremos al robot en un entorno con objetos fijos y movibles (figura 5.10). Se supondrá que la apertura de la horquilla permanece fija.

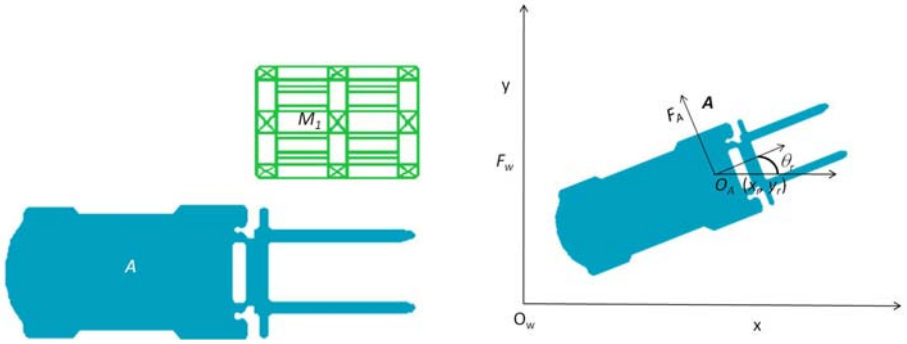
5.3.1. Escenario básico con criterios de calidad en la horquilla

El primer paso para aplicar nuestro método de síntesis de agarres es especificar los sistemas de referencia para el robot y el entorno. En la figura 5.2(b) vemos cómo el origen del sistema de referencia F_A que se mueve con el robot se ha situado en el centro del eje que une las ruedas delanteras motrices de nuestra carretilla elevadora. De esta forma una configuración q de A vendrá especificada por (x_r, y_r, θ_r).

Se utilizará el algoritmo 4.3 donde será necesario especificar las matrices del robot y del entorno así como la zona de agarre de la horquilla.

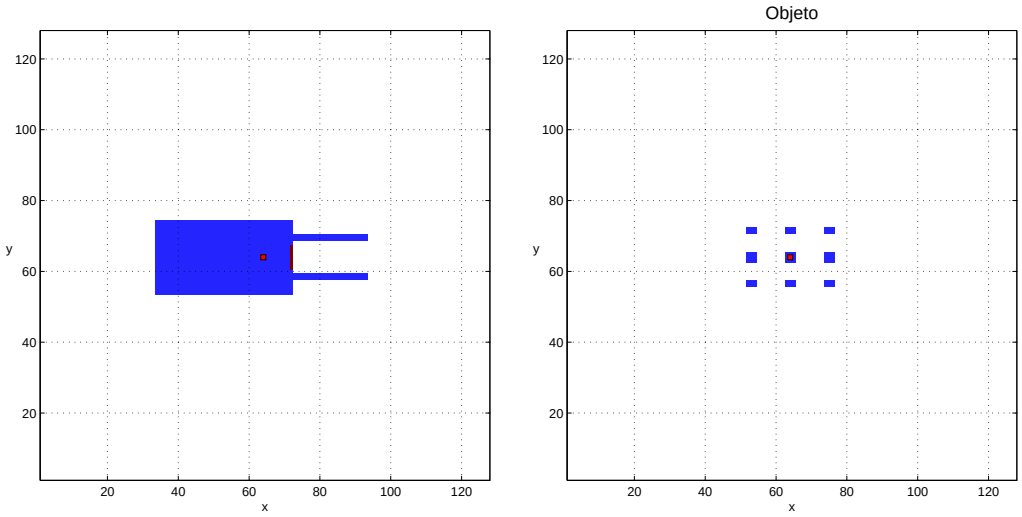
Teniendo en cuenta las dimensiones reales del palé y del robot, se realiza una discretización $N = 128$ para las coordenadas espaciales y $M = 128$ para la coordenada angular trabajando a una escala de 1:50. En el robot para descartar las configuraciones que no son físicamente realizables (aquellas en las que el robot contacta con el objeto en zonas que no se corresponden con la zona interna del efector final, pinza o dedos y palma) se ha seleccionado la zona de agarre (**FA**) en la parte interna central de la horquilla. En la figura 5.3 se pueden ver las matrices del robot y del entorno.

Aplicando el algoritmo 4.3 para el cálculo de configuraciones de agarre se obtienen las configuraciones de los agarres con mayor longitud de contacto que, tal y como puede verse en la figura 5.4, se obtienen en las orientaciones $\frac{\pi}{2}$, $-\frac{\pi}{2}$, 0 y $-\pi$. Para cada una de estas



(a) Escenario (b) Sistemas de referencia

Figura 5.2: Robot paletizador y palé.



(a) Carretilla (b) Palé Europeo

Figura 5.3: Las matrices del robot A^* y del palé europeo ${M_1}^*$.

cuatro orientaciones, también se producen máximos locales en su frontera de agarre (figura 5.5). En esta figura solo se ha representado el perfil de la frontera de agarre para las orientaciones 0 y $\frac{\pi}{2}$ puesto que las de $-\pi$ y $-\frac{\pi}{2}$ serían simétricas. Si llevamos nuestro robot a los puntos calculados tendremos al robot paletizador agarrando el palé en cada una de las configuraciones de agarre obtenidas (figura 5.6).

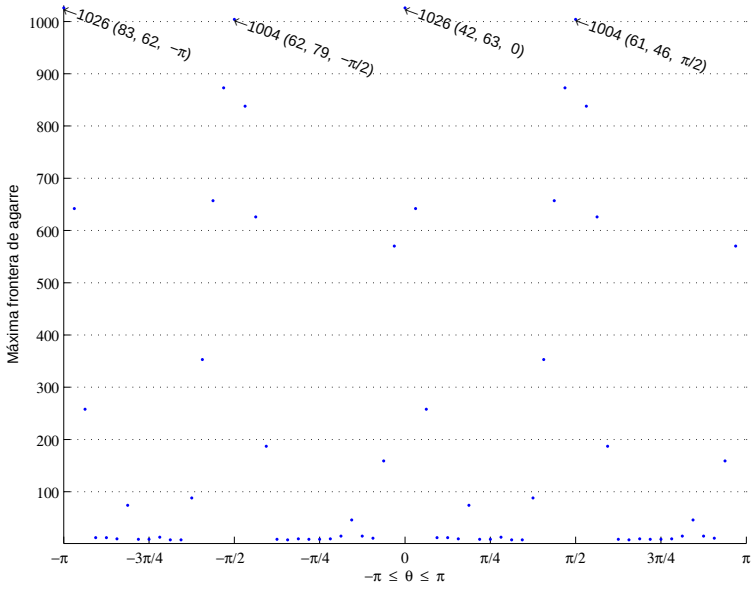


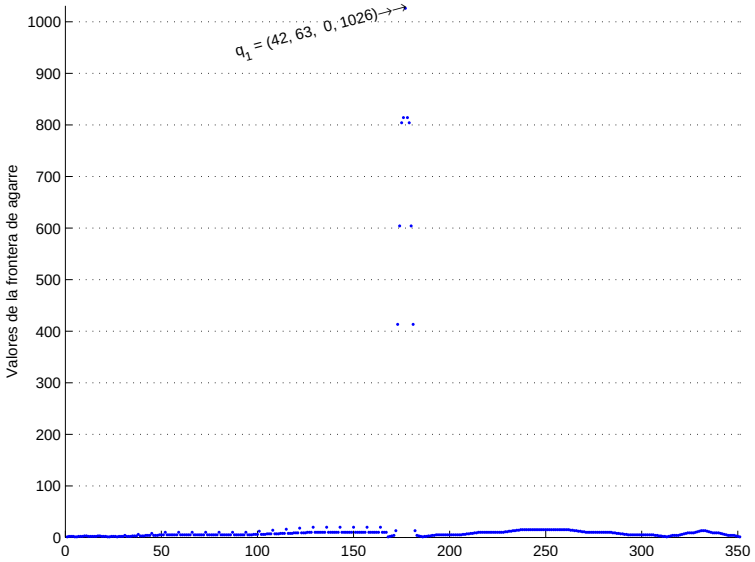
Figura 5.4: Valores máximos de la frontera de agarre para todas las orientaciones.

5.3.2. Escenario básico con criterios de calidad orientados a la tarea

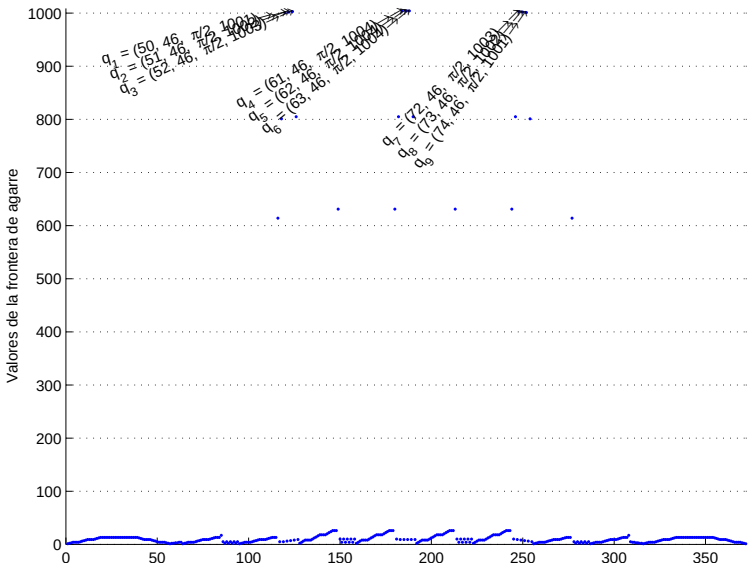
Los resultados previos se han obtenido aplicando el criterio de mayor longitud de contacto. Sin embargo, como se puede ver en la figura 5.6, entre estos agarres se encuentran aquellos donde la parte central de la horquilla contacta con alguno de los tacos del palé, o aquellos en los que uno de los lados de la horquilla queda en la parte exterior del palé. Estos agarres no son los adecuados para que el robot paletizador pueda manipular un palé, sino que las palas han de pasar entre los tacos y lo más centradas posible para así equilibrar la carga.

Para contemplar este aspecto relacionado con la tarea que se desea realizar con el palé, que es su transporte seguro por el robot paletizador, en nuestro algoritmo está prevista la posibilidad de especificar zonas preferidas de agarre (**FM**) en los objetos del entorno. En la manipulación de palés parece claro que la zona preferida de agarre pueden ser los tacos centrales de cada una de las caras del palé. Entonces se especificará esa zona en **FM**.

Con este criterio los agarres se producen todos ellos con la pinza situada en los tacos

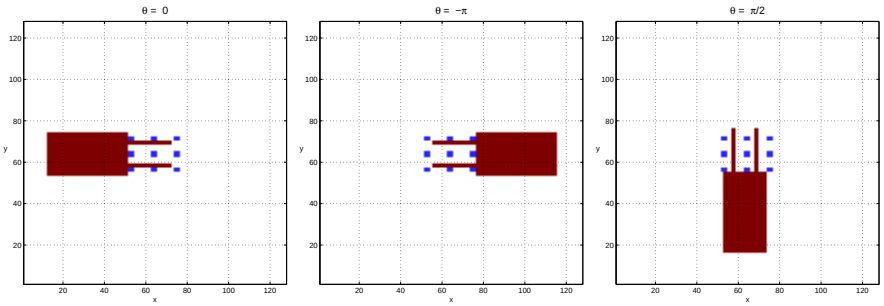


(a) $\theta = 0$

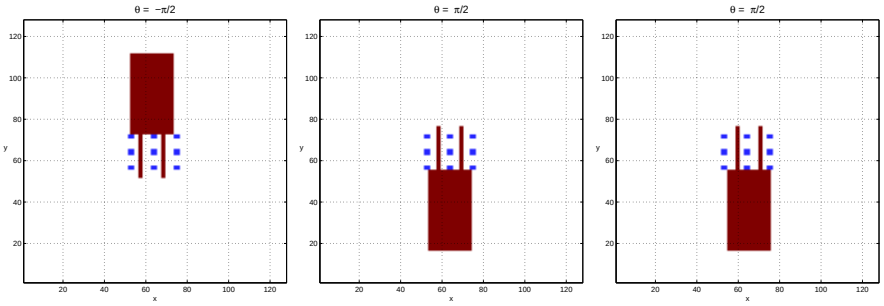


(b) $\theta = \frac{\pi}{2}$

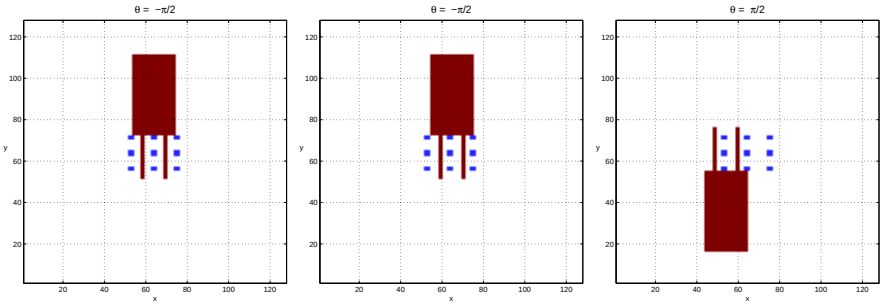
Figura 5.5: Perfil de la frontera de agarre en las orientaciones 0 y $\frac{\pi}{2}$.



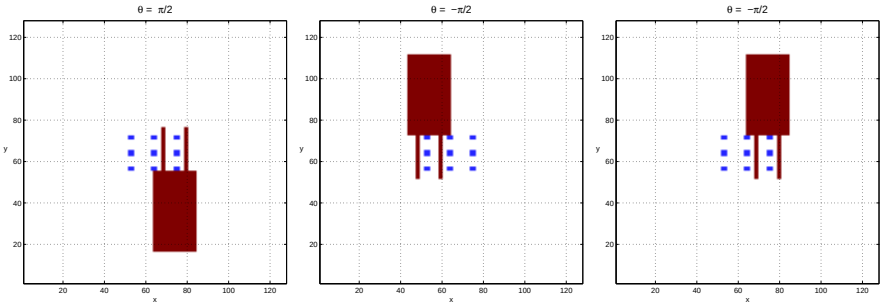
(a) $q_1 = (42, 63, 0, 1026)$ (b) $q_2 = (83, 62, -\pi, 1026)$ (c) $q_3 = (61, 46, \frac{\pi}{2}, 1004)$



(d) $q_4 = (62, 79, -\frac{\pi}{2}, 1004)$ (e) $q_5 = (62, 46, \frac{\pi}{2}, 1004)$ (f) $q_6 = (63, 46, \frac{\pi}{2}, 1004)$



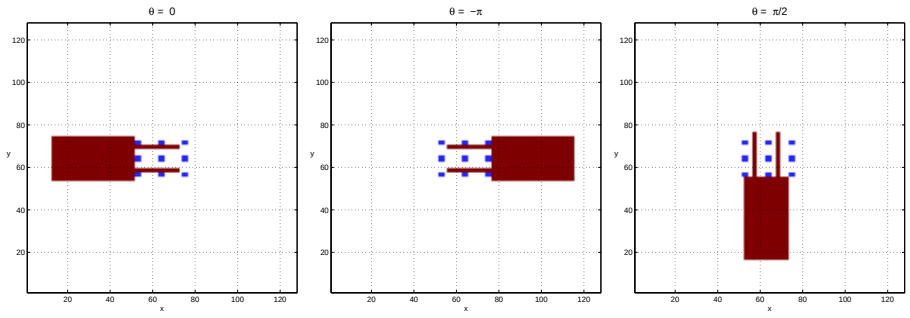
(g) $q_7 = (63, 79, -\frac{\pi}{2}, 1004)$ (h) $q_8 = (64, 79, -\frac{\pi}{2}, 1004)$ (i) $q_9 = (52, 46, \frac{\pi}{2}, 1003)$



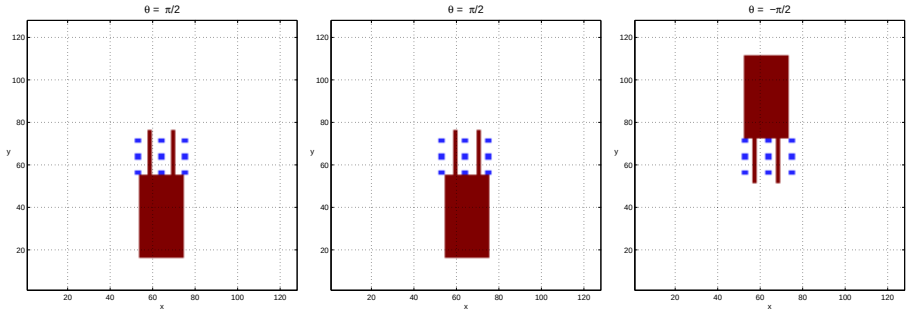
(j) $q_{10} = (72, 46, \frac{\pi}{2}, 1003)$ (k) $q_{11} = (53, 79, -\frac{\pi}{2}, 1003)$ (l) $q_{12} = (73, 79, -\frac{\pi}{2}, 1003)$

Figura 5.6: Robot en las configuraciones de agarre cuando se especifica únicamente la zona de agarre en su horquilla.

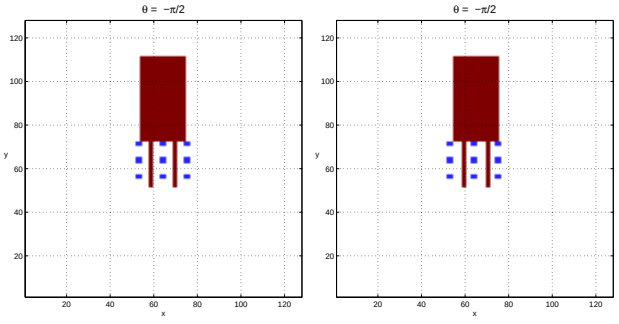
centrales del palé (figura 5.7). Las orientaciones de los mejores agarres siguen siendo las mismas que en el caso anterior (figura 5.8), pero los máximos locales de la frontera de agarre (figura 5.9) en estas orientaciones permiten discriminar claramente los agarres adecuados a la tarea que se desea realizar.



(a) $q_1 = (18, 63, 0, 132026)$ (b) $q_2 = (107, 62, -\pi, 132026)$ (c) $q_3 = (61, 22, \frac{\pi}{2}, 132004)$



(d) $q_4 = (62, 22, \frac{\pi}{2}, 132004)$ (e) $q_5 = (63, 22, \frac{\pi}{2}, 132004)$ (f) $q_6 = (62, 103, -\frac{\pi}{2}, 132004)$



(g) $q_7 = (63, 103, -\frac{\pi}{2}, 132004)$ (h) $q_8 = (64, 103, -\frac{\pi}{2}, 132004)$

Figura 5.7: Robot en las configuraciones de agarre con zona de agarre en la horquilla del robot y en el los tacos centrales del palé.

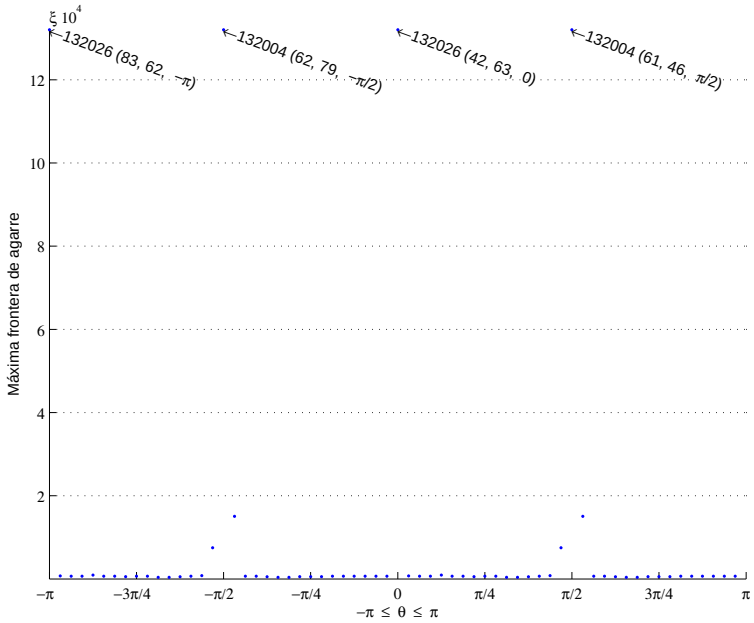


Figura 5.8: Frontera de agarre en todas las orientaciones.

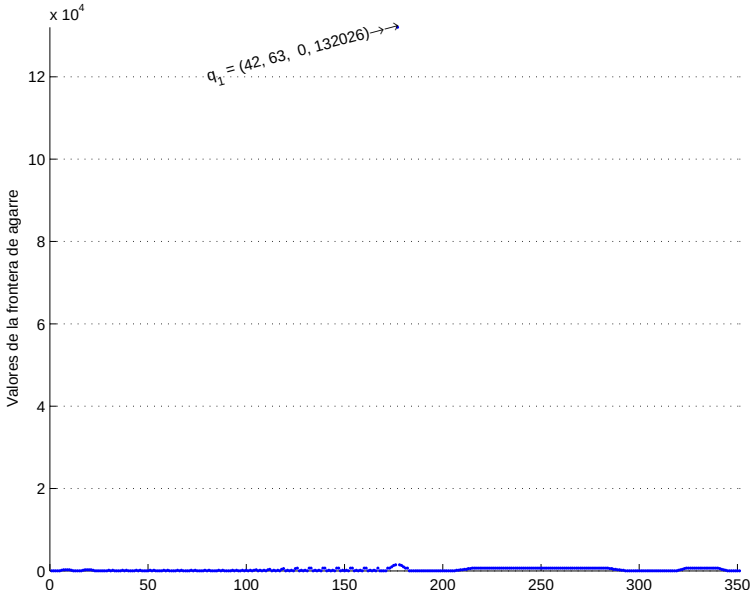
5.3.3. Accesibilidad: cálculo del C-espacio libre

Para completar este apartado situaremos nuestro robot paletizador en un entorno (figura 5.10) en el que existen obstáculos como paredes, columnas, estanterías, cajas o palés, y palés que el robot ha de manipular. Este escenario nos servirá también de base para la planificación de trayectorias que se explicará en la siguiente sección.

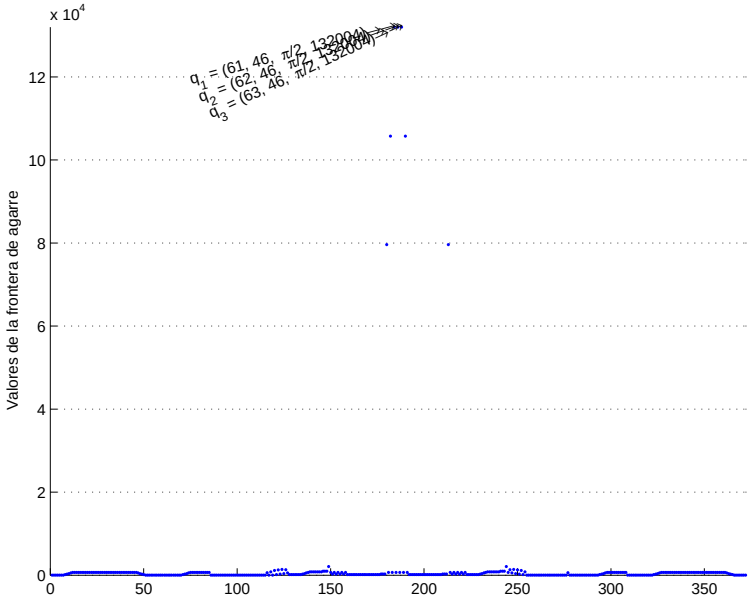
En nuestro método de síntesis de agarres, el hecho de situar el objeto en un escenario se traduce en representar en la matriz del entorno E^* tanto los obstáculos como los objetos movibles. A las celdas ocupadas por los obstáculos se les asignará el valor 1, a las ocupadas por el objeto movible un 2 y a la zona de agarre **FM** el valor V_{FM} en el objeto movible aumentado AE^*. En la matriz del robot A^* se especificará una zona de agarre en el centro de la horquilla.

En el escenario 5.10 el robot, las paredes, las columnas (en azul) y los palés presentes (en negro) se representan con la misma escala que para el robot y el palé a manipular (en verde). Trabajaremos con a una resolución de 256.

Al aplicar nuestro algoritmo de síntesis de agarres (algoritmo 4.3) se obtienen los mejores agarres (figura 5.12) en las orientaciones 0, $-\pi$ y $\frac{\pi}{2}$. No se ha seleccionado ningún agarre en la orientación $-\frac{\pi}{2}$, puesto que el robot no puede adoptar configuraciones de agarre en esa orientación sin colisionar con un palé que está en su entorno. Así, nuestro método de síntesis de agarres cumple con la característica de accesibilidad que es una de las principales



(a) $\theta = 0$



(b) $\theta = \frac{\pi}{2}$

Figura 5.9: Perfil de la frontera de agarre en las orientaciones 0 y $\frac{\pi}{2}$.

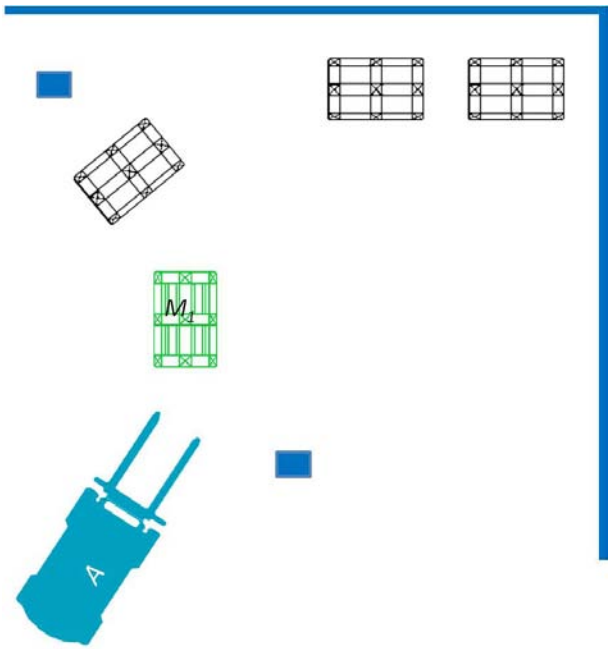
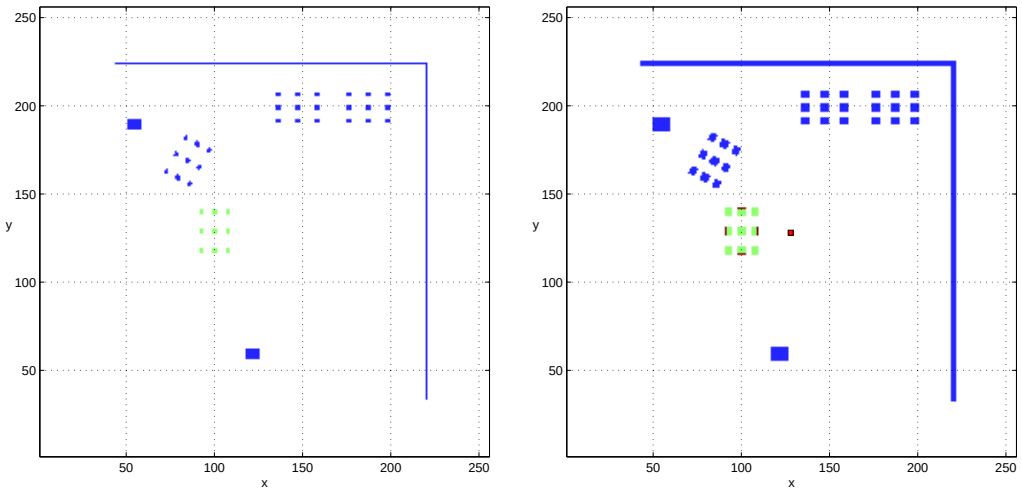


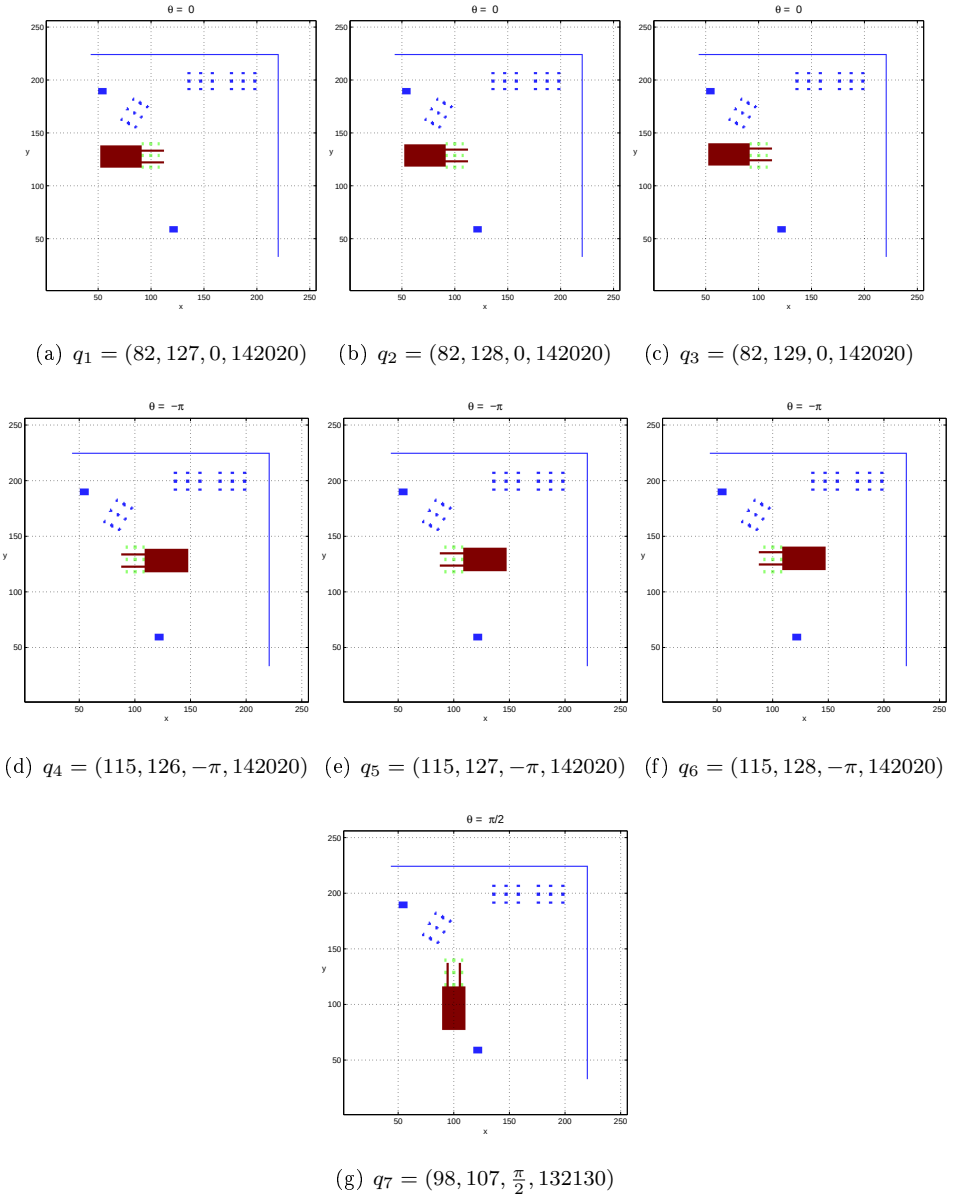
Figura 5.10: Escenario de trabajo del robot paletizador.



(a) E^* (b) AE^*

Figura 5.11: Escenario E^* y escenario aumentado AE^* con obstáculos y un objeto movible.

que se suelen imponer a un agarre ([Cut84], [Ngu86b]) pues no se seleccionará ningún agarre no alcanzable por el robot. Esto no significa que el robot pueda llegar hasta el punto de agarre sin colisionar con ningún obstáculo, es decir, exista una ruta accesible, sino que en la configuración de agarre el robot no colisiona con ningún obstáculo. La tarea de determinar cómo alcanzar la posición de agarre sin colisionar con ningún obstáculo le corresponde al planificador de trayectorias como se verá en las siguientes secciones.



(a) $q_1 = (82, 127, 0, 142020)$ (b) $q_2 = (82, 128, 0, 142020)$ (c) $q_3 = (82, 129, 0, 142020)$

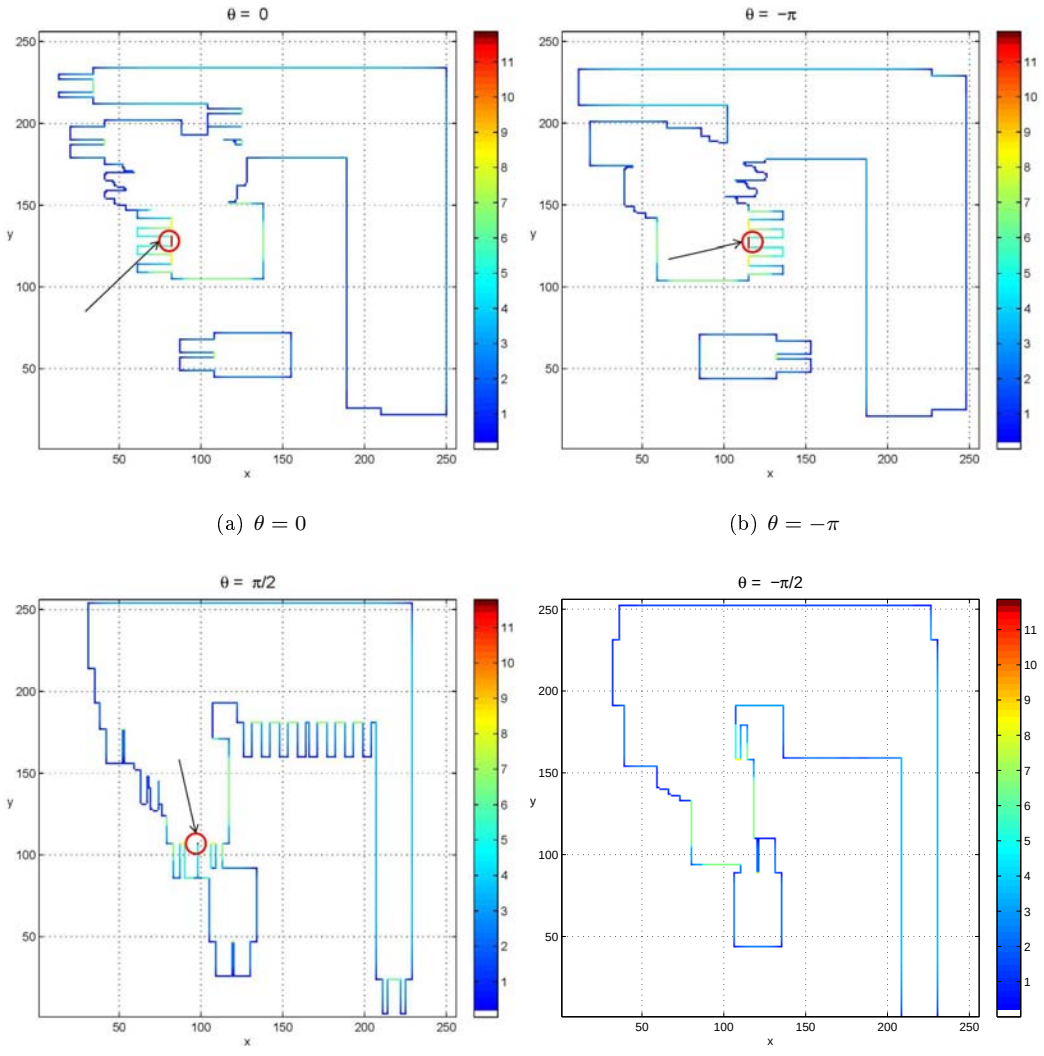
(d) $q_4 = (115, 126, -\pi, 142020)$ (e) $q_5 = (115, 127, -\pi, 142020)$ (f) $q_6 = (115, 128, -\pi, 142020)$

(g) $q_7 = (98, 107, \frac{\pi}{2}, 132130)$

Figura 5.12: Robot en las configuraciones de agarre en un escenario con obstáculos y con objetos movibles.

En la figura 5.13 se muestra en 2D la frontera de agarre en las orientaciones 0, $-\pi$, $\frac{\pi}{2}$

y $-\frac{\pi}{2}$. La frontera de agarre es por definición la frontera exterior de los objetos de nuestro escenario proyectados en el C-espacio. Por tanto, toda la zona interior de esta frontera son configuraciones del C-espacio donde el robot colisiona con los obstáculos del entorno. Para las tres orientaciones donde se realizan agarres los valores más altos de esta frontera (figuras 5.13(a), 5.13(b) y 5.13(c)) se han destacado en color rojo. En la orientación $-\frac{\pi}{2}$ que produce colisión no existe ningún punto en la frontera de agarres (figura 5.11) donde se obtengan valores máximos.



(a) $\theta = 0$ (b) $\theta = -\pi$

(c) $\theta = \frac{\pi}{2}$ (d) $\theta = -\frac{\pi}{2}$

Figura 5.13: Frontera de agarre en las orientaciones 0, $-\pi$ y $\frac{\pi}{2}$ donde se producen los agarres y en $-\frac{\pi}{2}$ donde no es posible agarrar el objeto.

5.4. Planificación de trayectorias

Una vez determinado el conjunto de las configuraciones de agarres (ordenado de mayor a menor calidad), el siguiente paso es el cálculo de la trayectoria a seguir por el robot hasta alcanzar las configuraciones de agarre. Teniendo en cuenta que nuestro robot paletizador no puede girar libremente en el espacio de trabajo (sus ruedas no pueden deslizarse lateralmente) estará por tanto sujeto a restricciones no holónomas en sus movimientos. A esta clasificación se adscriben los vehículos rodados, debido a la imposición de no deslizamiento del punto de contacto de las ruedas.

Existen diferentes enfoques y métodos que tratan de dar respuesta al problema de la planificación de movimientos en sistemas no holónomos y, en particular, a los robots tipo coche. Pero encontrar la solución óptima global se presenta como un problema muy complejo. Por ello, algunos de los métodos para obtener trayectorias en este tipo de vehículos son heurísticos. En este trabajo se utilizará una simplificación del problema que bajo ciertas restricciones permite reducirlo a un problema únicamente geométrico.

En esta sección se expondrán estas restricciones y se explicará el método que vamos a utilizar para calcular los caminos más cortos en sistemas no holónomos . Como estamos realizando una tarea de manipulación, primero aplicaremos el método para calcular las rutas en las que el robot se mueve solo, caminos de tránsito, y una vez que el robot llegue a la configuración de agarre y coja el objeto aplicaremos el mismo método para calcular los caminos de transferencia.

5.4.1. Planificación de trayectorias en sistemas no holónomos

Es conocido que un coche no se puede conducir lateralmente porque las ruedas traseras tendrían que deslizarse en lugar de girar. Si las cuatro ruedas pudieran girar simultáneamente en la dirección del estacionamiento, aparcar un coche sería trivial. Las complicaciones de la maniobra de aparcar un coche están causadas por las restricciones de giro.

Un coche se puede modelar como un cuerpo rígido que se mueve en el plano. Una configuración se denota por $q = (x, y, \theta)$. El origen del sistema de referencia del coche se sitúa en el centro del eje trasero y los puntos del eje x a lo largo del eje principal del coche. s denota la velocidad (con signo) del coche y ϕ el ángulo de la dirección (en la orientación de las ruedas de la figura 5.14 es negativo). La distancia entre los ejes trasero y delantero se representa por L. Si el ángulo

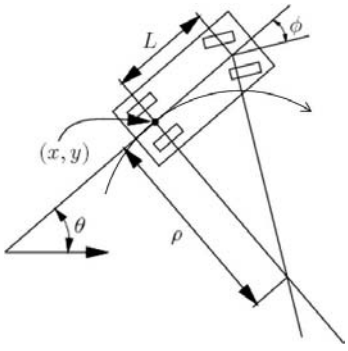


Figura 5.14: El coche sencillo.

de dirección (*steering angle*) se fija en ϕ, el coche gira con un movimiento circular con un radio de giro ρ. Obsérvese que ρ se puede hallar como la intersección de los dos ejes en la figura 5.14 (el ángulo entre estos ejes es $|\phi|$).

Utilizando esta notación el movimiento del coche se representa por un conjunto de ecuaciones de la forma [LaV06]

$$\begin{aligned} \dot{x} &= f_1(x, y, \theta, s, \phi) \\ \dot{y} &= f_2(x, y, \theta, s, \phi) \\ \dot{\theta} &= f_3(x, y, \theta, s, \phi) \end{aligned} \tag{5.1}$$

En un intervalo de tiempo pequeño $\triangle t$, el coche se mueve aproximadamente en la dirección a la que apuntan las ruedas traseras. En el límite cuando $\triangle t$ tiende a 0 se cumple que $\frac{dx}{dy} = \tan\theta$. Como $\frac{dy}{dx} = \frac{\dot{y}}{\dot{x}}$ y $\tan\theta = \frac{\sin\theta}{\cos\theta}$, esta condición se puede escribir como una restricción de Pfaffian:

$$-\dot{x}\sin\theta + \dot{y}\cos\theta = 0 \tag{5.2}$$

Esta condición se cumple si $\dot{x} = \cos\theta$ y $\dot{y} = \sin\theta$. Cualquier múltiplo escalar de la solución es también una solución; el factor de escala se corresponde directamente con la velocidad s del coche. Así las dos primeras ecuaciones para representar el movimiento de un coche son $\dot{x} = s\cos\theta$ y $\dot{y} = s\sin\theta$.

El siguiente paso es obtener $\dot{\theta}$. Sea w la distancia recorrida por el coche (la integral de la velocidad). Como se muestra en la figura 5.14, ρ representa el radio de un círculo que es trazado por el centro en el eje trasero, si el ángulo de la dirección es fijo. Obsérvese que $dw = \rho d\theta$. Por trigonometría, $\rho = \frac{L}{\tan\phi}$, lo que implica

$$d\theta = \frac{\tan\phi}{L} dw \tag{5.3}$$

Dividiendo a ambos lados por dt y utilizando el hecho de que $\dot{w} = s$ se obtiene

$$\dot{\theta} = \frac{s}{L}\tan\phi \tag{5.4}$$

Así se ha modelado el movimiento del coche pero no se han definido las variable de actuación. Supóngase que la velocidad s y el ángulo de la dirección ϕ se especifican directamente con las variable de actuación u_s y u_ϕ, respectivamente. Obsérvese que las variables de actuación se denotan con la letra u y un subíndice. De esta forma es más fácil localizar

las variables de actuación en las ecuaciones. Con esta notación el conjunto de ecuaciones de movimiento de un coche sencillo es

$$\begin{aligned} \dot{x} &= u_s \cos\theta \\ \dot{y} &= u_s \sin\theta \\ \dot{\theta} &= \frac{u_s}{L} \tan u_\phi \end{aligned} \tag{5.5}$$

En las expresiones 5.5 hay que especificar U, las variables de actuación de la forma $u = (u_s, u_\theta)$. Los posibles valores del ángulo de la dirección u_s están en el intervalo $[-\frac{\pi}{2}, \frac{\pi}{2}]$ pero los valores $-\frac{\pi}{2}$ y $\frac{\pi}{2}$ presentan problemas, puesto que las ruedas delanteras no pueden girar tanto, ya que colisionarían con su eje. Por tanto, un coche sencillo tiene un máximo ángulo de dirección, $\phi_{max} < \frac{\pi}{2}$ y como puede observarse en la figura 5.14 un máximo ángulo de dirección implica un mínimo radio de giro ρ_{min}.

En cuanto a la velocidad u_s, un vehículo real tiene una velocidad máxima y su comportamiento cambia drásticamente dependiendo de la velocidad. Por ejemplo, no se intentará girar el mínimo ángulo a 140km/h. Se supone implícitamente que un coche sencillo se mueve a velocidad reducida debido a que en estas condiciones se pueden despreciar los efectos dinámicos. Así, la velocidad se restringe a los valores $u_s \in \{-1, 0, 1\}$ sin que esto afecte a las configuraciones que el coche puede adoptar.

Basándose en estas consideraciones respecto a la velocidad y el ángulo de giro se pueden considerar varios modelos cinemáticos simplificados de vehículos con ruedas, entre ellos:

- Un coche sencillo: $U = [-1, 1] \times [-\phi_{max}, \phi_{max}]$. Imponiendo que $|u_\phi| \leq \phi_{max} < \frac{\pi}{2}$ se obtienen un coche con mínimo radio de giro $\rho_{min} = \frac{L}{\tan\phi_{max}}$.
- El coche de Reeds-Shepp: La velocidad se restringe a los valores $u_s \in \{-1, 0, 1\}$. Intuitivamente en este modelo la velocidad puede tener tres valores: todo hacia atrás, parado y todo hacia adelante.
- El coche de Dubins: La velocidad se restringe a los valores $u_s \in \{0, 1\}$; es decir, respecto al coche de Reeds-Shepp se elimina la marcha atrás.

5.4.2. Caminos más cortos para un robot tipo coche

El estudio de los caminos más cortos en ausencia de obstáculos para un sistema similar a un robot tipo coche fue realizado por Dubins [Dub57]. El control de velocidad lineal u_1 es fijo e igual a 1. Dubins demuestra que los caminos más cortos son curvas de clase C^1 compuestas de arcos de círculos de radio 1 y segmentos de líneas rectas. Reeds y Shepp

[RS90] extienden el trabajo de Dubins considerando que el control de velocidad lineal u_1 puede tener valores positivos y negativos.

En este tipo de vehículos se puede restringir su conjunto de movimientos a los especificados por unas pocas primitivas de movimiento (Ej. girar a la derecha, ir recto, girar a la izquierda). Las curvas de Dubins y/o Reeds y Shepp definen el camino más corto entre una posición inicial y otra final restringiendo los movimientos a una combinación de unas pocas primitivas.

5.4.2.1. El coche de Dubins

En la versión de Dubins de un coche sencillo se supone que el coche se mueve únicamente hacia adelante y a velocidad constante, $u_s = 1$. Otra importante restricción es que cuando gira lo hace con el máximo ángulo de giro de la dirección ϕ_{max}, que resulta en un mínimo radio de giro del vehículo ρ_{min}.

Puesto que la velocidad es constante, el sistema se puede simplificar a:

$$\begin{aligned} \dot{x} &= \cos\theta \\ \dot{y} &= \sin\theta \\ \dot{\theta} &= u \end{aligned} \tag{5.6}$$

donde u se elige en el intervalo $U = [-\tan\theta_{max}, \tan\theta_{max}]$. Por simplicidad, se supone que $\tan\theta = 1$. Los resultados son válidos para cualquier $\phi_{max} \in (0, \frac{\pi}{2})$.

Como se demuestra en [Dub57], entre cuales quiera dos configuraciones, el camino más corto para el coche de Dubins se puede expresar siempre como una combinación de no más de tres primitivas de movimiento. Cada primitiva de movimiento aplica una acción constante durante un intervalo de tiempo. Mas aún, las únicas acciones para seguir el camino más corto son $u \in \{-1, 0, 1\}$. Las primitivas y sus símbolos asociados se muestran en la tabla 5.1. La primitiva S conduce el vehículo en línea recta. Las primitivas L y R producen giros todo a la derecha o todo a la izquierda, respectivamente.

Símbolo	**Dirección:** u
S	0
L	1
R	-1

Tabla 5.1: Las tres primitivas de movimiento con las que construir las curvas óptimas para el coche de Dubins

Utilizando estos símbolos, todos los caminos más cortos se expresan con una secuencia de estos tres símbolos que se corresponden con el orden en el que se ejecutan las primitivas. A

cada una de estas secuencias se les denomina *palabra*. Dubins demostró que sólo seis palabras son óptimas:

$$\{LRL, RLR, LSL, LSR, RSL, RSR\} \tag{5.7}$$

El camino más corto entre dos configuraciones cualesquiera (q_I, q_G) se puede siempre caracterizar por una de estas palabras, que se denominan las *curvas de Dubins*. Para ser más precisos, también se puede expresar la duración de cada primitiva. Para L o R, un subíndice denotará la rotación total que se ha acumulado durante la aplicación de la primitiva. Para S, el subíndice indica la distancia recorrida. Así, las curvas de Dubins se puede caracterizar de una forma más precisa como:

$$\{L_\alpha R_\beta L_\gamma, R_\alpha L_\beta R_\gamma, L_\alpha S_d L_\gamma, L_\alpha S_d R_\gamma, R_\alpha S_d L_\gamma, R_\alpha S_d R_\gamma\} \tag{5.8}$$

donde $\alpha, \gamma \in [0, 2\pi)\,, \beta \in (\pi, 2\pi)$, y $d \geq 0$. La figura 5.15 muestra dos casos. Obsérvese que β debe ser mayor que α (si fuera menor, las curvas óptimas serían otras).

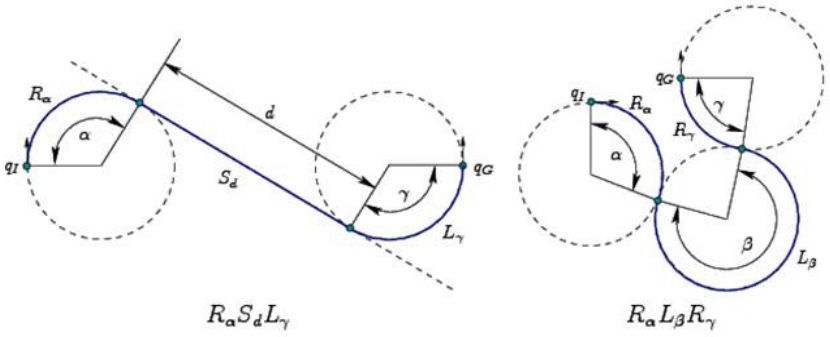


Figura 5.15: Trayectorias de dos de las curvas de Dubins en $W = R^2$

5.4.3. Algoritmo propuesto para el cálculo de las curvas de Dubins

En esta sección se describirá el algoritmo desarrollado en esta tesis doctoral para obtener las curvas de Dubins como método de cálculo del camino más corto entre dos configuraciones en ausencia de obstáculos bajo ciertas restricciones. Estas restricciones, que son las mismas que en el coche de Dubins, son: (1) no es posible retroceder, (2) se avanza siempre a velocidad constante (velocidad nominal) (3) los giros son con radio mínimo. En estas condiciones el problema de la planificación de movimientos en sistemas no holónomos, como los vehículos con ruedas tipo coche, se reduce a un problema geométrico.

El algoritmo que proponemos es el algoritmo 5.1. Los parámetros de entrada son: configuración inicial (q_i), configuración final (q_f), velocidad nominal (v) y radio mínimo de giro

(r). Las configuraciones inicial y final se especificarán por su posición y orientación en el plano. La velocidad es constante y debe fijarse a un valor moderado para que se puedan despreciar los efectos de la dinámica. El mínimo radio de giro del vehículo vendrá dado por el máximo ángulo de giro de la dirección del robot.

El primer lugar se calculan los incrementos correspondientes para los puntos de las rutas (líneas 2 y 3 del algoritmo 5.1) tanto para los movimientos de giro (todo a izquierda o todo a derecha) como en los de línea recta (todo hacia adelante). El siguiente paso consiste en trazar en cada una de las configuraciones dos circunferencias de radio igual al mínimo radio de giro del vehículo, una en el sentido de giro positivo y otra en el negativo (líneas 4 a 7 del algoritmo 5.1). Tomaremos la convención de que los giros positivos son giros a la izquierda (antihorario) y los negativos giros a la derecha (horario).

Una vez trazadas las circunferencias y para cada una de las 4 primitivas posibles: LSL, RSR, LSR y RSL (figura 5.16), se trazan las dos rectas tangentes que unen cada circunferencia de origen con cada circunferencia de destino. De las dos tangentes se determina la que es factible teniendo en cuenta el sentido del giro (líneas 9 y 15 del algoritmo 5.1). Los puntos en salida y en llegada de la tangente seleccionada son los puntos de salida (fin) del movimiento de giro inicial y de llegada (inicio) del movimiento de giro final. En la figura 5.16 se representan las dos tangentes (en línea continua la tangente posible según el sentido del giro y en línea discontinua la que no lo es) para cada una de las primitivas de movimiento.

El siguiente paso es la construcción de la ruta (líneas 16 a 24 del algoritmo 5.1) que estará formada por la concatenación de los movimientos de giro desde la configuración inicial hasta la configuración de salida calculada, de linea recta hasta la configuración de llegada calculada (la tangente posible según el sentido del giro) y de giro desde la configuración de llegada hasta la configuración final.

Todos los cálculos geométricos asociados al algoritmo se detallarán en los siguientes apartados.

5.4.3.1. Validación del algoritmo

Para explicar y validar nuestro algoritmo analizamos un ejemplo. En la figura 5.17 se representa una configuración inicial $q_I = (-30, 30, 0)$ y una final $q_G = (30, -30, \frac{\pi}{4})$. En esta figura las configuraciones se representan como un punto en los ejes x e y del sistema de coordenadas y la orientación θ se representa con un vector de color azul. En cada configuración se han trazado dos circunferencias cuyo radio es un determinado mínimo radio de giro, una para el giro positivo (en color rojo) y otra para el negativo (en color azul).

En las figuras 5.18, 5.19, 5.20 y 5.21 se muestran las curvas LSL, RSR, RSL y LSR

Algoritmo 5.1: Cálculo de las curvas de Dubins

Entrada: Configuración inicial ($q_I = (x_i, y_i, \theta_i)$), configuración final ($q_F = (x_f, y_f, \theta_f)$), velocidad ($v$) y radio mínimo de giro (r)

Salida: Curvas de Dubins ($Ruta_{pp}, Ruta_{nn}, Ruta_{np}, Ruta_{pn}$)

Inicio
- $\Delta s = v * \Delta t \leftarrow$ Obtener el incremento en el desplazamiento;
- $\Delta\theta = (v/r) * \Delta t \leftarrow$ Obtener el incremento angular;
- $CIRC\_POS_{out} \leftarrow$ Construir una circunferencia de radio r en sentido positivo en la configuración de salida q_I;
- $CIRC\_NEG_{out} \leftarrow$ Construir una circunferencia de radio r en sentido negativo en la configuración de salida q_I;
- $CIRC\_POS_{in} \leftarrow$ Construir una circunferencia de radio r en sentido positivo en la configuración de llegada q_F;
- $CIRC\_NEG_{in} \leftarrow$ Construir una circunferencia de radio r en sentido negativo en la configuración de llegada q_F;

```
//curvas posibles: LSL, RSR, LSR y RSL
```

- **para** i *desde* 1 *hasta* 4 **hacer**
  - $\theta_c \leftarrow$ Determina dirección de conexión;
  - $\alpha_1 \leftarrow$ Dirección (en salida) de la primera tangente que une las dos circunferencias;
  - $\alpha_2 \leftarrow$ Dirección (en salida) de la segunda tangente que une las dos circunferencias;
  - $\beta_1 \leftarrow$ Dirección (en llegada) de la primera tangente que une las dos circunferencias;
  - $\beta_2 \leftarrow$ Dirección (en llegada) de la segunda tangente que une las dos circunferencias;
  - $\alpha \leftarrow$ Seleccionar entre las tangentes de salida la más próxima a θ_c;
  - $\beta \leftarrow$ Seleccionar entre las tangentes de entrada la más próxima a θ_c;
  - $A \leftarrow$ Calcular el punto de salida;
  - $B \leftarrow$ Calcular el punto de llegada;
  - $S \leftarrow$ Determina la recta de conexión (recta que une A con B);
  - $\theta_{out} \leftarrow$ Determina orientación de salida;
  - $G_{out} \leftarrow$ Construir el giro de salida desde θ_{out} hasta θ_c;
  - $Ruta_i = concatenar(G_{out}, S)$; `//Añade S (la recta)`
  - $\theta_{in} \leftarrow$ Determina orientación de entrada;
  - $G_{in} \leftarrow$ Construir el giro de llegada desde θ_c hasta θ_{in};
  - $Ruta_i = concatenar(Ruta_i, G_{in})$; `//Añade el giro de llegada`
- Curvas de Dubins $\leftarrow$ Ordenar las rutas ($Rutas_i$) primero la más corta;

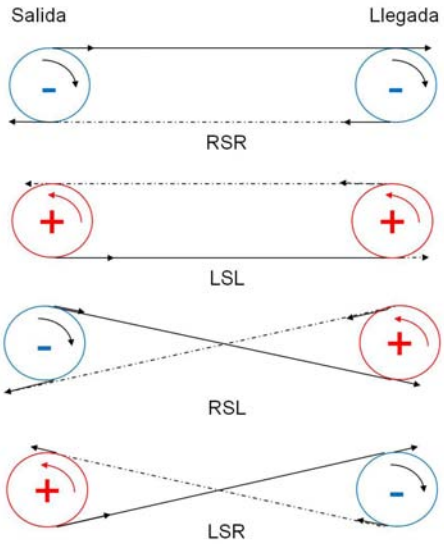


Figura 5.16: Cálculo de las curvas de Dubins.

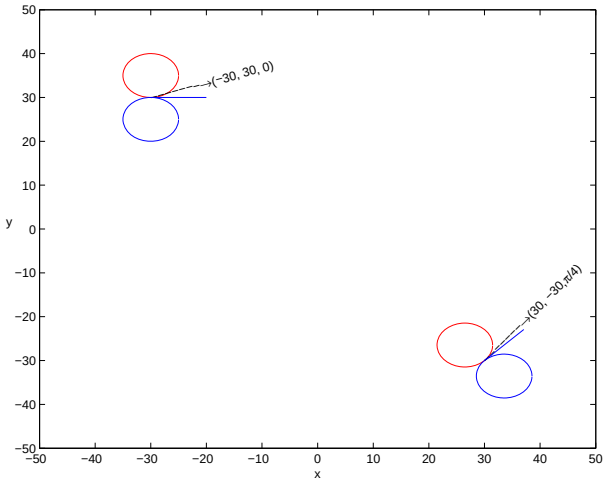
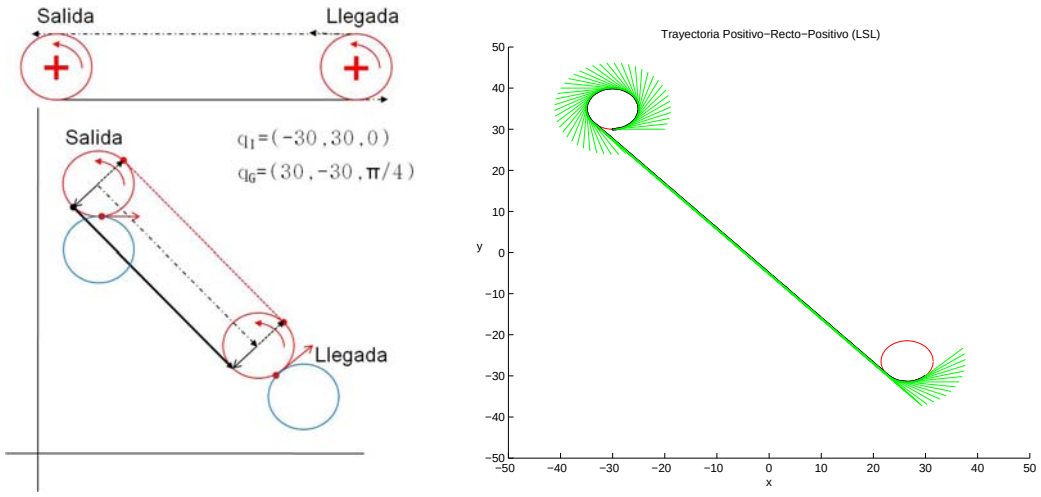


Figura 5.17: Circunferencias con sentido positivo y negativo en la configuración inicial y final.

resultantes al aplicar nuestro algoritmo. En la parte izquierda de las figuras se muestra un esquema de cómo se realiza el cálculo y la parte derecha la representación de las curvas calculadas en una simulación realizada en Matlab. En estas figuras las circunferencias de giro positivo se dibujan en color rojo y las de giro negativo en color azul. La recta tangente que une la circunferencia de origen con la de destino (la tangente posible según el sentido del giro) se representa en color negro y en trazo más grueso. Este es el movimiento en línea recta (S). El movimiento inicial de giro se produce desde la configuración inicial (dibujada en su posición en las figuras con un vector en color rojo que representa su orientación) hasta la configuración de salida en la circunferencia donde comenzaría el movimiento S. El movimiento final de giro se produce desde la configuración de llegada del movimiento en línea recta hasta alcanzar la configuración final deseada (dibujada también en las figuras en su posición con un vector de color rojo en su orientación).



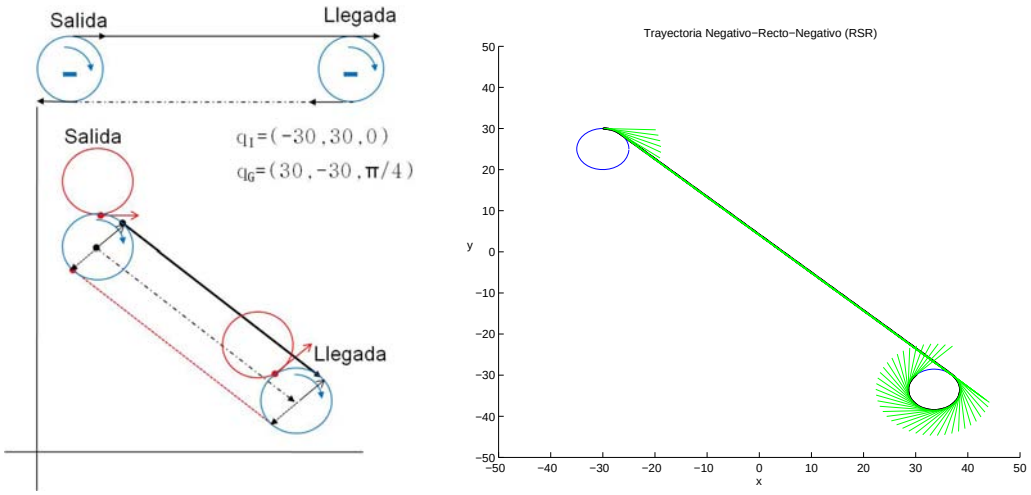
(a) Esquema (b) Representación de la ruta

Figura 5.18: Curva de Dubins LSL.

En la simulación realizada en Matlab (parte derecha de las figuras 5.18, 5.19, 5.20 y 5.21) las coordenadas de las rutas obtenidas se representan como un punto en los ejes de coordenadas (x, y) y su orientación con un vector de color verde.

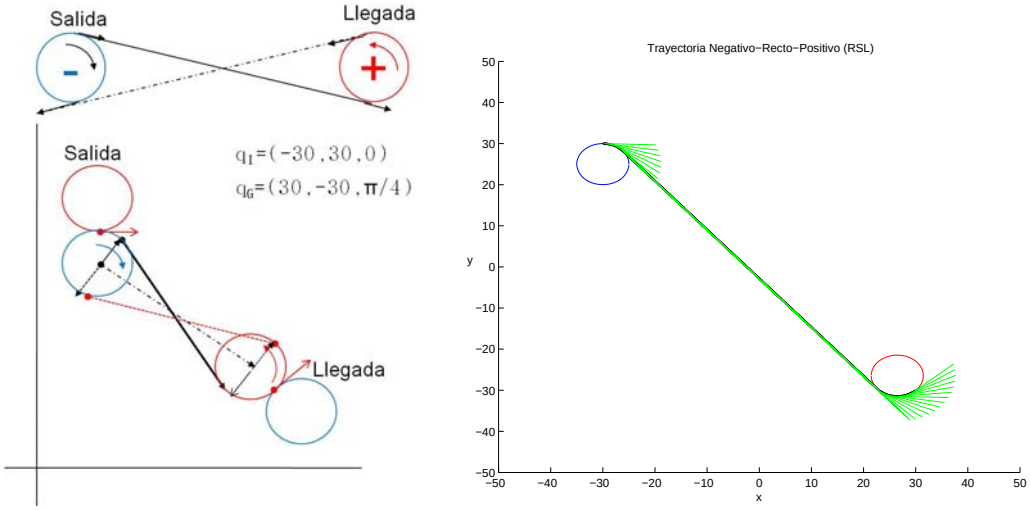
5.4.4. Cálculo geométrico de las curvas de Dubins

En el algoritmo 5.1 se han utilizado una serie de operaciones, como por ejemplo obtener la circunferencia de salida, obtener la tangente posible según el sentido del giro o construir la ruta de giro de entrada, pero no se han aportado las expresiones matemáticas con la que



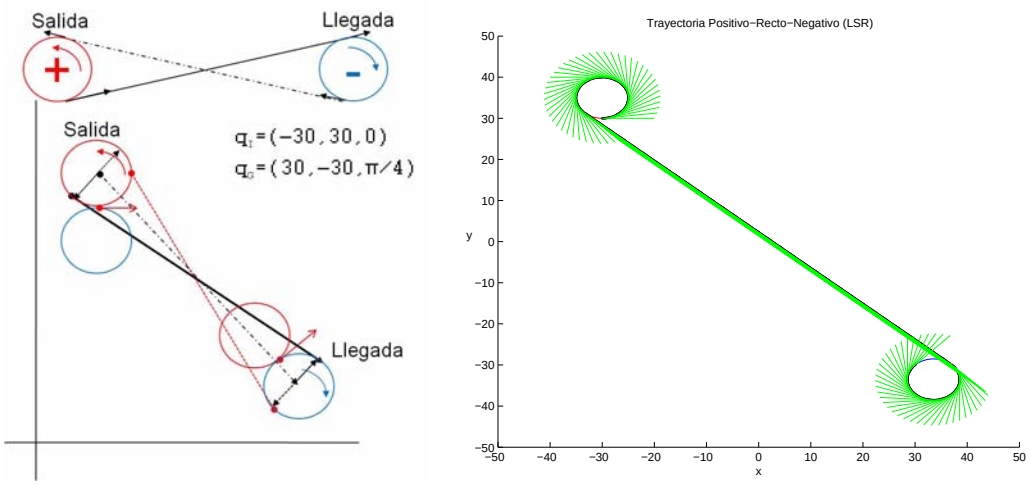
(a) Esquema (b) Representación de la ruta

Figura 5.19: Curva de Dubins RSR.



(a) Esquema (b) Representación de la ruta

Figura 5.20: Curva de Dubins RSL.



(a) Esquema (b) Representación de la ruta

Figura 5.21: Curva de Dubins LSR.

obtener lo deseado. Algunas de estas operaciones merecen una explicación más detallada que abordaremos en esta sección.

En las dos primeras líneas del algoritmo 5.1 se obtienen los incrementos que vamos a considerar tanto para los desplazamientos en línea recta Δs como en los movimientos de giro $\Delta\theta$ con las expresiones

$$\begin{aligned} \Delta s &= v * \Delta t \\ \Delta\theta &= (v/r) * \Delta t \end{aligned} \tag{5.9}$$

donde v es la velocidad constante fijada en función de los parámetros de entrada del algoritmo, r el radio mínimo de giro e Δt es el incremento temporal con el que se trabaja.

El siguiente paso es el cálculo de las circunferencias de radio r: una en sentido positivo y otra en el negativo que se trazan en la configuración inicial y en la final (líneas 4 a 7 del algoritmo 5.1). Para ello, primero obtendremos su centro y luego su ecuación. El centro de las circunferencias se calcula con la expresión

$$X_C = (x_j - r\cos(\theta_j \pm \frac{\pi}{2}),\ y_j - r\sin(\theta_j \pm \frac{\pi}{2})) \tag{5.10}$$

donde (x_j, y_j, θ_j) representa la configuración (inicial y final) desde la que se va a trazar la circunferencia. Aplicando esta expresión, tanto para la configuración inicial como para la final, obtenemos el centro de las circunferencias de sentido positivo y negativo en salida y en llegada. Para las circunferencias de giro positivo aparece $-\frac{\pi}{2}$ y para el giro negativo se tiene $+\frac{\pi}{2}$.

Conocidas las coordenadas del centro, calculamos los puntos de las circunferencias me-

diante

$$CIRC = (X_C + (r\cos\theta,\ r\sin\theta)),,\theta \in [0, 2\pi) \tag{5.11}$$

donde los valores de θ están discretizados en intervalos de $\Delta\theta$.

Para determinar la dirección de conexión (θ_c) entre la posición inicial y la final (línea 9 del algoritmo 5.1) se calcula la dirección del vector unitario que une el centro de la circunferencia de salida con el centro de la circunferencia de llegada con la expresión

$$\begin{aligned} \overrightarrow{u_c} &= \frac{(X_{C\_in} - X_{C\_out})}{\|X_{C\_in} - X_{C\_out}\|} \\ \theta_u &= \arctan(\overrightarrow{u_c}) \end{aligned} \tag{5.12}$$

donde X_{C_in} y X_{C_out} son las coordenadas del centro de la circunferencia de entrada y de salida respectivamente, previamente calculadas con la expresión 5.10. La dirección de conexión en las circunferencias se obtiene con la expresión

$$\theta_c = \theta_u \pm \pi/2 \tag{5.13}$$

con signo positivo para los giros negativos y negativo para los giros positivos.

Para calcular la dirección en salida y en llegada de las dos tangentes que unen las circunferencias (líneas 10 a 13 del algoritmo 5.1) utilizaremos la propiedad de que las tangentes son perpendiculares al radio y por tanto su producto escalar es 0. Primero se determinan los vectores unitarios de los radios con la expresión

$$\overrightarrow{u_r} = \frac{CIRC - X_C}{\|CIRC - X_C\|} \tag{5.14}$$

Las direcciones de las dos tangentes (α_1 y α_2) satisfacen estas ecuaciones

$$\begin{aligned} \overrightarrow{u_r}(\alpha_1) \cdot \overrightarrow{u_c} &= 0 \\ \overrightarrow{u_r}(\alpha_2) \cdot \overrightarrow{u_c} &= 0 \end{aligned} \tag{5.15}$$

De las dos tangentes posibles se selecciona la única posible según el sentido del giro que será la más próxima a la dirección de conexión (líneas 14 y 15 del algoritmo 5.1)

$$\alpha,,|\alpha - \theta_c| = min_{i\in 1,2}|\alpha_i - \theta_c| \tag{5.16}$$

Conocidas ya las direcciones de salida y de llegada, podemos obtener los puntos que hay que unir (líneas 16 y 17 del algoritmo 5.1) y seguir en línea recta. Así, ambos puntos se obtienen con la expresión

$$A \in CIRC,,A = X_C + (r\cos\alpha, r\sin\alpha) \tag{5.17}$$

Para el punto de salida, X_C es el centro de la circunferencia de salida, r el radio y α la dirección de la tangente de salida. Mientras que para el punto de llegada, X_C es el centro

de la circunferencia de llegada y α la dirección de la tangente en la circunferencia de llegada (en el algoritmo 5.1 a las tangentes en llegada las hemos denominado como β) . El recorrido S es la línea recta que une el punto de salida en la circunferencia de salida con el punto de llegada en la circunferencia de llegada.

El siguiente paso es obtener los movimientos circulares desde la configuración de salida hasta el punto de salida de la tangente de conexión y desde el punto de llegada en la tangente a la configuración de destino. En primer lugar, determinaremos la orientación de salida y de entrada en sus respectivas circunferencias con la expresión

$$\theta_{io} = \theta_r \pm \frac{\pi}{2} \quad ,, \theta_{io} \in [0, 2\pi) \tag{5.18}$$

con signo positivo para los giros negativos y negativo para los giros positivos. Para calcular la orientación de salida (línea 19 del algoritmo 5.1) θ_r es la orientación en la configuración inicial (θ_i) y para la orientación de llegada (línea 22 del algoritmo 5.1) θ_r es la orientación en la configuración final (θ_f) .

Las expresiones para determinar los movimientos de giro son diferentes en función del sentido del giro y si se trata de la circunferencia de salida o de la de entrada. Comenzaremos con el giro positivo o a izquierda (L) de salida. Si la orientación de salida es menor que la orientación de conexión $\theta_{io} < \theta_c$, el movimiento de giro positivo va desde la orientación de salida hasta la orientación de conexión.

$$L1 = CIRC\_POS(\theta_{io}, \theta_c) \tag{5.19}$$

En caso contrario, el movimiento de giro positivo es la concatenación de los tramos desde la orientación de salida hasta 2π y desde 0 hasta la orientación de conexión.

$$\begin{aligned} L1_1 &= CIRC\_POS(\theta_{io}, 2\pi) \\ L1_2 &= CIRC\_POS(0, \theta_c) \\ L1 &= concatenar(L1_1, L1_2) \end{aligned} \tag{5.20}$$

Para el mismo giro pero en llegada, si la orientación de conexión es menor que la orientación de la salida $\theta_c < \theta_{oi}$, el movimiento de giro positivo va desde la orientación de conexión hasta la orientación de llegada o final.

$$L2 = CIRC\_POS(\theta_c, \theta_{io}) \tag{5.21}$$

En caso contrario, el movimiento de giro positivo es la concatenación de los tramos desde la orientación de conexión hasta 2π y desde 0 hasta la orientación de llegada.

$$\begin{aligned} L2_1 &= CIRC\_POS(\theta_c, 2\pi) \\ L2_2 &= CIRC\_POS(0, \theta_{io}) \\ L2 &= concatenar(L2_1, L2_2) \end{aligned} \tag{5.22}$$

Para finalizar consideraremos los giros negativos o a derecha (R) diferenciando salida y llegada, al igual que hemos hecho con los giros positivos. En salida, si la orientación de salida es mayor que la orientación de conexión $\theta_{oi} > \theta_c$, el movimiento de giro negativo va desde la orientación de salida hasta la orientación de conexión. Como la circunferencia es de sentido negativo (los ángulos decrecen) el tramo obtenido ha de recorrerse en sentido inverso.

$$R1 = CIRC\_NEG(\theta_{oi}, \theta_c) \tag{5.23}$$

En caso contrario, el movimiento de giro negativo es la concatenación de los tramos desde la dirección de salida hasta 0 y desde 2π hasta la dirección de conexión. Igualmente los recorrido ha de realizarse en sentido inverso.

$$\begin{aligned} R1_1 &= CIRC\_NEG(\theta_{io}, 0) \\ R1_2 &= CIRC\_NEG(2\pi, \theta_c) \\ R1 &= concatenar(R1_1, R1_2) \end{aligned} \tag{5.24}$$

Para el mismo giro en llegada, si la orientación de conexión es mayor que la orientación de llegada $\theta_c > \theta_{io}$, el giro negativo de llegada va desde la orientación de conexión hasta la orientación de salida.

$$R2 = CIRC\_NEG(\theta_c, \theta_{io}) \tag{5.25}$$

En caso contrario, el movimiento de giro negativo es la concatenación del tramo desde la orientación de conexión hasta 0 y el tramo desde 2π hasta la orientación de salida. Todos los recorridos se han de realizar en sentido inverso.

$$\begin{aligned} R2_1 &= CIRC\_NEG(\theta_c, 0) \\ R2_2 &= CIRC\_NEG(2\pi, \theta_{io}) \\ R2 &= concatenar(R2_1, R2_2) \end{aligned} \tag{5.26}$$

5.4.5. Validación de los cálculos geométricos de la curva LSR

Para ilustrar los cálculos descritos en esta sección tomaremos como ejemplo la curva LSR (positivo/recto/negativo) (figura 5.21) y el esquema de la figura 5.22, donde la configuración inicial es $q_i = (-30, 30, 0)$ y la final $q_f = (30, -30, \frac{\pi}{4})$.

En primer lugar obtenemos los incrementos en los desplazamientos en línea recta y en los giros con la expresión 5.9. Con estos valores se discretizarán los movimientos lineales y circulares, respectivamente.

En la configuración inicial se traza la circunferencia positiva (en rojo en la figura 5.22). Para calcular su ecuación, primero obtenemos el centro con la expresión 5.10, considerando $\theta_j + \frac{\pi}{2}$, donde θ_j es la orientación de la configuración inicial. La ecuación de la circunferencia positiva se obtiene con la expresión 5.11. De igual forma se obtiene la ecuación de la

circunferencia negativa (en color azul en la figura 5.22) de llegada. El centro de la circunferencia negativa en llegada se obtiene con la expresión 5.10, considerando $\theta_j - \frac{\pi}{2}$, donde θ_j es la orientación de la configuración final. La ecuación que representa a la circunferencia negativa se obtiene con la expresión 5.11. En ambas circunferencias los valores de θ están discretizados en intervalos de $\Delta\theta$.

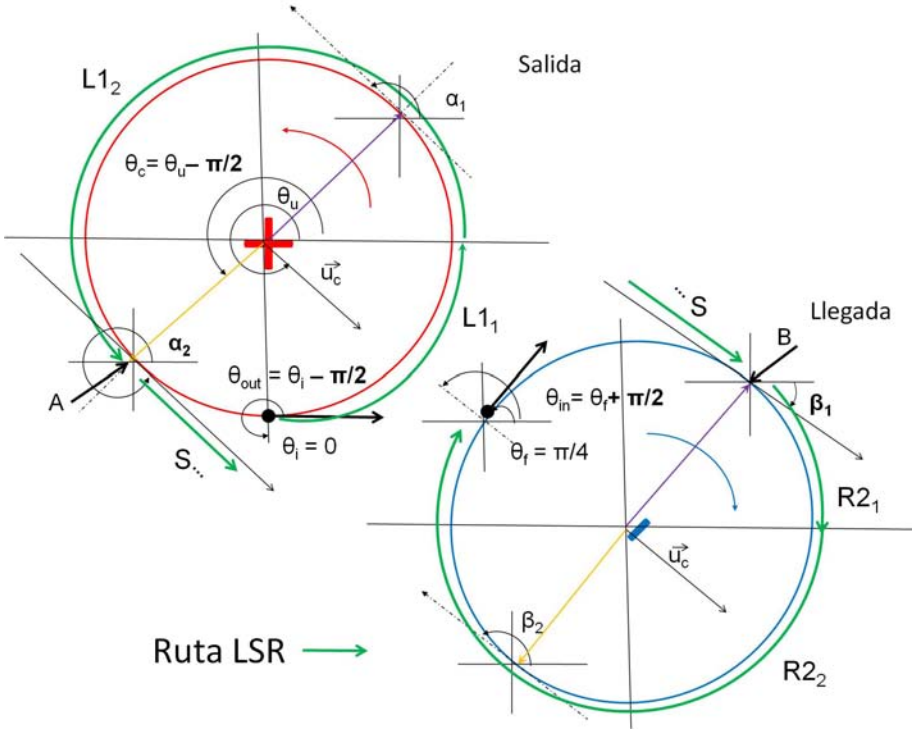


Figura 5.22: Esquema del cálculo de la curva de Dubins LSR

La dirección de la conexión de ambas circunferencias (θ_c en la figura 5.22) se calcula a partir de la expresión 5.12 y posteriormente restando $\frac{\pi}{2}$ en la expresión 5.13 al ser el giro de salida positivo.

El movimiento en línea recta es la tangente que une el punto de salida en la circunferencia de salida y el punto de entrada en la circunferencia de llegada cuando recorremos las circunferencias en el sentido del giro. Estos puntos aparecen etiquetados como A y B, respectivamente, en la figura 5.22 y han sido obtenidos con la expresión 5.17. Para determinar A aplicando la expresión 5.17, X_C son las coordenadas del centro de la circunferencia de salida y α es α_2 que es, de las orientaciones donde se producen las tangentes (α_1 y α_2) en la circunferencia de salida, la más próxima al sentido del giro. Para determinar B con la misma expresión, X_C son las coordenadas del centro de la circunferencia de llegada y α es β_1 que es, de las orientaciones donde se producen las tangentes (β_1 y β_2) en la circunferencia de llegada, la más próxima al sentido del giro.

El movimiento circular de salida (figura 5.22) es la concatenación de dos tramos $L1_1$ y $L1_2$ que se calculan con la expresión 5.20 . El primero desde la orientación de salida θ_{out} hasta 2π y el segundo desde 0 hasta la orientación de conexión θ_c. θ_{out} se ha calculado con la expresión 5.18, siendo θ_r la orientación de la configuración inicial y considerando $-\frac{\pi}{2}$ al tratarse de un giro positivo.

El movimiento circular de llegada (figura 5.22) es la concatenación de dos tramos $R2_1$ y $R2_2$ que se calculan con la expresión 5.26 . El primero desde la orientación de conexión θ_c hasta 0 y el segundo desde 2π hasta la orientación de llegada θ_{in}. θ_{in} se ha calculado con la expresión 5.18, siendo θ_r la orientación de la configuración final y considerando $+\frac{\pi}{2}$ al tratarse de un giro negativo. Ambos tramos han de recorrerse en sentido inverso al tratase de giros negativos donde el ángulo decrece.

El movimiento completo dibujado en color verde en la figura 5.22 estaría formado por la concatenación del movimiento circular positivo de salida ($L1_1$ y $L1_2$), el tramo en línea recta (S) y el movimiento circular negativo de llegada ($R2_1$ y $R2_2$). En el apéndice A se detallan los algoritmos para cada una de las cuatro curvas de Dubins utilizando las expresiones matemáticas descritas en esta sección. Además se aporta un esquema similar al de la figura 5.22 que utilizando la notación del algoritmo permite entender las operaciones geométricas realizadas.

5.5. Planificación de manipulaciones

Una vez expuestos los procedimientos de determinación de agarres, de obtención de trayectorias en robots con restricciones no holónomas y teniendo en cuenta que como parte de nuestro método de síntesis de agarres obtenemos también la representación de los obstáculos en el espacio de las configuraciones, en esta sección se va a describir el procedimiento completo de manipulación integrando todos estos aspectos.

5.5.1. Algoritmo de manipulación propuesto

Una tarea de manipulación consiste en dado un entorno donde existen objetos fijos y objetos movibles, el robot debe agarrar un objeto y transportarlo a una determinada configuración final. El algoritmo propuesto (algoritmo 5.2) consta de los siguientes pasos:

(1) determina las configuraciones de agarre, (2) calcula los caminos de tránsito desde una configuración de partida hasta alcanzar una configuración de agarre, y (3) calcula los caminos de transferencia desde la configuración de agarre hasta la configuración final deseada considerando que el robot transporta el objeto.

Algoritmo 5.2: Algoritmo de manipulación

Entrada: Configuración inicial (q_I), configuración final (q_G), matriz del robot (A^*), matriz del entorno (E^*), zona de agarre de la pinza (**FA**), zona preferida de agarre del objeto (**FM**), factor de agarre (k), umbral (u), resolución (N), velocidad (v) y radio mínimo de giro del robot (r)

Salida: Caminos de transferencia ($C_{transferencia}$), caminos de tránsito ($C_{transito}$)

```
1  Inicio
2      (Q_grasp, CE*) = calculaAgarres(A*, E*, FA, FM, k, u, N) ;   /*Determinar las
       configuraciones de agarre ordenadas de mayor a menor calidad*/
3      noFinAgarres = true;
4      i = 1 ;                                                      /*Caminos de tránsito*/
5      mientras ∃Q_grasp(i) ∧ noFinAgarres hacer
6          C_dubins = calculaDubins(q_I, Q_grasp(i), v, r) ;   /*Calcular curvas Dubins ordenadas
           primero las mas cortas*/
7          noFinCurvas = true;
8          j = 1;
9          mientras ∃C_dubins(j) ∧ noFinCurvas hacer
10             si noHayColisiones(CE*, C_dubins(j)) entonces
11                 noFinCurvas = false;
12                 C_transito = C_dubins(j)
13             si no
14                 j = j + 1;
15         si noFinCurvas entonces
16             i = i + 1 ;      /*Ninguna de las curvas son realizable, comprobar el siguiente
               agarre*/
17         si no
18             noFinAgarres = false;
19     si noFinAgarres entonces
20         C_transito = ∅ ;                          /*Ninguno de los agarres es posible*/
21     si no
           /*Caminos de transferencia                                                    */
22         A_t ← Obtener la matriz que representa al robot con el objeto agarrado;
23         E_t ← Obtener la matriz del entorno sin el objeto agarrado por el robot;
24         CE_t = obtenerCObstaculo(A_t, E_t);
25         C_dubins = calculaDubins(Q_grasp(i), q_G, v, r);
26         noFinCurvas = true;
27         j = 1;
28         mientras ∃C_dubins(j) ∧ noFinCurvas hacer
29             si noHayColisiones(CE_t, C_dubins(j)) entonces
30                 noFinCurvas = false;
31                 C_transferencia = C_dubins(j);
32             si no
33                 j = j + 1;
34         si noFinCurvas entonces
35             C_transferencia = ∅ ;                 /*Ninguna de las curvas son realizables*/
```

En la línea 2 del algoritmo 5.2 se calculan las configuraciones de agarre con el algoritmo adecuado en cada caso, según lo expuesto en el capítulo 4. El algoritmo de agarres toma como entradas las matrices que representa al robot (A^*) y al entorno (E^*). Si se desean especificar zonas de agarre en robot y/o en el objeto movible se especificarán, respectivamente, en **FA** y en **FM**. El algoritmo del cálculo de agarres devuelve el conjunto Q_{grasp} formado por las configuraciones ordenadas de mayor a menor calidad en las que el robot puede agarrar el objeto movible. Además, y como parte del proceso de determinación de los mejores agarres, se obtiene también CE^* que es la proyección de los obstáculos en el C-espacio. CE^* se utilizará como entrada en el test de colisiones dentro del planificador de movimientos.

El siguiente paso de nuestro planificador será hallar el camino de tránsito desde la posición actual del robot hasta la configuración de agarre. En este camino el robot se mueve sin carga hasta la configuración en la que agarrará al objeto. Para determinar los caminos de tránsito se hallarán las curvas de Dubins (línea 6 del algoritmo) desde la configuración actual del robot hasta la configuración de agarre. Para el cálculo de las curvas se utiliza el algoritmo A.1 que devuelve las 4 curvas posibles ordenadas según la distancia recorrida (primero la más corta). En el siguiente paso (líneas 9 a 14) del planificador de caminos se evalúan si las curvas son o no factibles (si colisionan o no con algún obstáculo). Para ello se realiza un test de colisiones utilizando CE^*. Se verifica para cada una de las configuraciones que forman las curvas si su valor de CE^* es 0 (no hay colisión) o mayor que 0 (hay colisión). Este proceso se repite hasta que se encuentre el primer agarre realizable con la ruta más corta posible o sin resultados si ninguno de los agarres es realizable. Así se completaría la obtención de los caminos de tránsito (líneas 5 a 14 del algoritmo).

Si se ha obtenido un camino de tránsito, el siguiente paso es hallar los caminos de transferencia. Es decir, los caminos en los que el robot traslada el objeto a la configuración de destino. El proceso de obtención de los caminos de transferencia se realizan entre las líneas 22 y 33 del algoritmo de manipulación. En primer lugar se obtiene la matriz que representa al robot con el objeto agarrado que formará el nuevo robot. En la matriz que representa el entorno no se considera el objeto trasladado por el robot. Con estas matrices se determina CE^*, que al igual que en la determinación de los caminos de tránsito, se utilizará en el test de colisiones. Las curvas de Dubins se calculan ahora entre la configuración de agarre y la configuración final donde hay que depositar el objeto. Y para finalizar, y de igual forma que para los caminos de tránsito, se verifica si todas las configuraciones que conforman la ruta colisiona o no con un obstáculo, consultando para cada una de ellas su valor en CE^*. El proceso termina al encontrar la curva más corta que sea realizable (no colisione con los obstáculos) o sin resultados si ninguna de las curvas es realizable.

5.5.2. Resultados de validación del algoritmo

Como ejemplo para validar nuestro algoritmo de manipulación de objetos por un robot paletizador (un sistema no holónomo) que se mueve en un entorno con la presencia de paredes, estanterías, columnas, cajas, etc., se ha realizado una simulación en Matlab.

Sea un escenario como el que se muestra en la Figura 5.23, en el que existen cinco objetos fijos en color azul (una pared, dos columnas y 3 palés) y uno movible en color verde (un palé). El objeto movible es un palé europeo. Todos ellos se han representado a escala 1:50, donde cada pixel representa 10cm. Este escenario es el mismo que se ha utilizado en el apartado de determinación de las configuraciones de agarre de este mismo capítulo.

El primer paso es la determinación de las configuraciones de agarre del palé M_1 cuando se utiliza la carretilla paletizadora A. El procedimiento descrito en la sección 5.3.3 determina para este escenario 7 configuraciones de agarre, ordenadas por el valor de la frontera de agarre (según nuestro criterio de calidad de agarres). Estas configuraciones trasladadas a nuestro escenario se marcan con asteriscos de color rojo en la figura 5.23. Las tres primeras configuraciones, las situadas a la derecha del palé, al igual que las tres siguientes, situadas a la izquierda, están muy próximas. Por esta razón, solo están representadas en la figura por un asterisco. Los valores de las configuraciones de agarre en el sistema de referencia de nuestro escenario son los que se muestran en la primera columna de la tabla 5.2 ordenados de mayor a menor calidad. Las tres primeras están a la izquierda del palé y se corresponden con el agarre por la parte ancha del mismo; los tres siguientes a la derecha también por la parte ancha; y el último en la parte inferior donde el robot agarraría al palé por la parte estrecha. El otro agarre por la parte estrecha superior no es posible puesto que en ese punto el robot colisiona con el palé B_3.

La tarea de manipulación consistirá en que el robot situado en la configuración inicial $q_I = (-8.0, -8.0, \frac{\pi}{2})$ agarre el palé M_1 y lo traslade a la configuración final $q_G = (6, 2.5, \frac{\pi}{2})$ (marcada en color verde en la figura 5.23).

El siguiente paso es la planificación de los caminos de tránsito para un robot no holónomo como es el robot paletizador. Los pasos del planificador son (1) calcular las curvas de Dubins y (2) verificar si son realizables; es decir, no existe colisión con ningún obstáculo (líneas 5 a 18 del algoritmo 5.2).

Para cada una de las configuraciones de agarre se calculan las curvas de Dubins desde la configuración inicial a cada una de las configuraciones de agarre calculadas. El procedimiento del cálculo de las curvas de Dubins devuelve las cuatro rutas posible ordenadas primero la más corta. Las rutas están formadas por las configuraciones $q_i(x_i, y_i, \theta_i)$ por las que ha

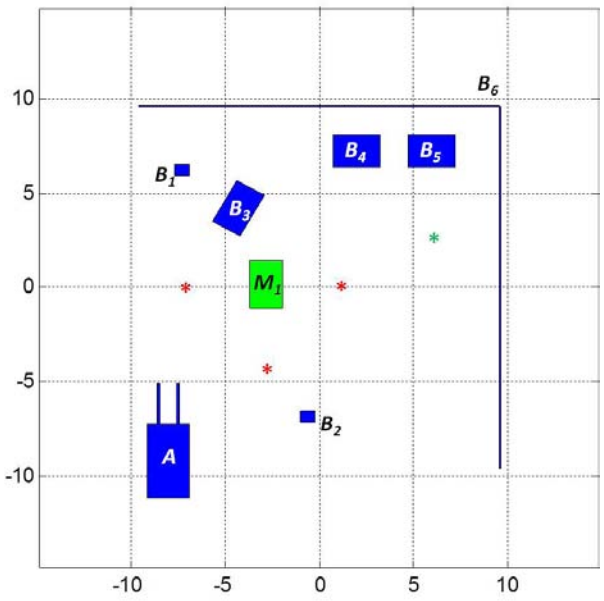


Figura 5.23: Escenario con un objeto movible y seis objetos fijos.

de pasar la carretilla considerando una velocidad constante v y su radio mínimo de giro ($r = 171cm$).

Para evaluar si una curva es factible, para todas las configuraciones que forman la ruta, se verifica si están libres o no (consultando su valor la estructura que contiene la proyección de los objetos en el C-espacio). Recordamos que esta estructura CE se obtuvo en el procedimiento de determinación de agarres y está formada por 256 matrices (número de niveles de discretización de la orientación θ) de dimensión 256×256 (coordenadas x, y) cada una.

	Orden			
q_G	1	2	3	4
(-6.70, 0.10, 0.00)	LSR	RSL	RSR	LSL
	Factible	(-5.35, 0.23, 1.18)	Factible	(-5.41, 0.00, 1.26)
(-6.70, -0.00, 0.00)	LSR	RSL	RSR	LSL
	Factible	(-5.38, 0.24, 1.18)	Factible	(-5.44, 0.00, 1.26)
(-6.70, -0.10, 0.00)	RSL	RSR	LSL	LSR
	Factible	(-5.38, 0.24, 1.18)	Factible	(-5.41, 0.17, 1.26)
(0.80, -0.20, $-\pi$)	RSR	RSL	LSR	LSL
	(-5.16, -3.63, 0.91)	factible	(-4.65, -3.54, 0.80)	factible
(0.80, -0.10, $-\pi$)	RSR	RSL	LSR	LSL
	(-5.18, -3.62, 0.91)	factible	(-4.67, -3.53, 0.80)	factible
(0.80, -0.00, $-\pi$)	RSR	RSL	LSR	LSL
	(-5.20, -3.61, 0.91)	factible	(-4.67, -3.51, 0.80)	factible
(-3.00, -4.20, $\pi/2$)	RSL	LSL	RSR	LSR
	factible	factible	(-1.12, -2.50,-0.10)	(-1.12, -2.50,-0.10)

Tabla 5.2: Rutas factibles (marcadas en rojo) para agarrar el palé europeo en el escenario de la figura 5.23 siendo $q_I = (-8.0, -8.0, \frac{\pi}{2})$. Las rutas marcadas en rojo no son realizables porque colisionan con un obstáculo en la configuración que se muestra en la celda.

En la tabla 5.2 se muestran para todos los agarres y en el orden en que han sido evaluados, las rutas factibles (en color verde) y las que colisionan con algún obstáculo en alguna

configuración de la ruta (en color rojo) y en el cuál se produce la colisión. Aunque el planificador devolvería sólo el primer agarre realizable (Figura 5.24(a)), con la curva más corta que se inicia en la configuración $q_I = (-8.0, -8.0, \frac{\pi}{2})$ y llega a la mejor configuración de agarre $q_G = (-6.7, -0.1, 0.0)$ con la ruta LSR o lo que es lo mismo, girando todo hacia la izquierda (giro positivo), seguir recto y girar todo hacia la derecha (giro negativo). Mostraremos también, para cada agarre, el resto de curvas y si son o no factibles. De esta forma demostramos la corrección de nuestros algoritmos de cálculo de curvas y de detección de colisiones.

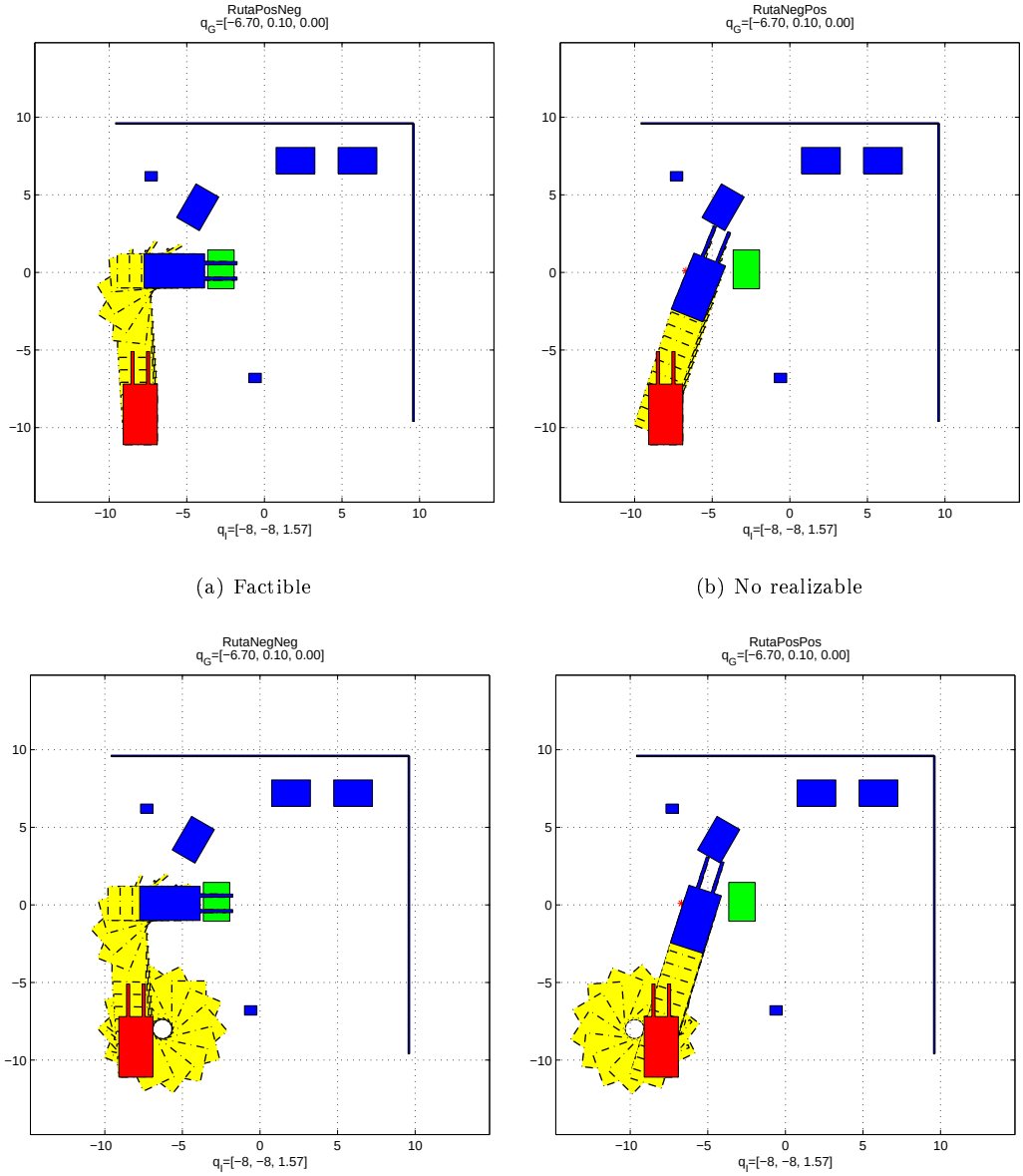
Como se puede observar en la tabla 5.2 , aunque los agarres son 7, podemos agruparlos en 3: (1) los agarres por la parte ancha a la izquierda del palé (los tres primeros), (2) los agarres a la derecha del palé por la parte ancha (los tres siguientes) y (3) el agarre por abajo en la parte estrecha del palé. Los resultados del cálculo de las rutas más cortas y si son o no factibles para estos tres agarres se muestran en las figuras 5.24, 5.25 y 5.26.

En la figura 5.24 vemos cómo se evalúan primero las curvas más cortas. La primera curva evaluada (figura 5.24(a)) es la que seleccionaría el planificador, puesto que es la primera más corta realizable para el mejor agarre. En la figura 5.25(c) la curva RSR es también realizable, pero es claramente más larga que la primera, puesto que el giro a derecha es casi completo, el camino recto es el mismo y el giro final a derecha es igual. Las dos combinaciones que faltan RSL y LSL no son realizables, pues ambas acaban con un giro positivo que no es posible puesto que se produce colisión con el palé situado en la parte superior del palé a manipular.

En el segundo grupo de agarres (los situados a la derecha del palé) las curvas que acaban en giro a la derecha (figuras 5.25(a) y 5.25(c)) no son posibles, puesto que la carretilla colisiona con el propio palé a agarrar. De las que acaban en giro a la izquierda, ambas son posibles, pero la RSL es más corta (5.25(b)) y por ello se evaluaría primero.

En el tercer grupo de agarres (agarre de la parte inferior) las rutas que acaban en giro a derecha no son posibles pues se produce colisión con el propio palé (figura 5.26(c) y (d). Las que acaban en giro a izquierda son factibles y la más corta es la primera evaluada (figura 5.26(a)).

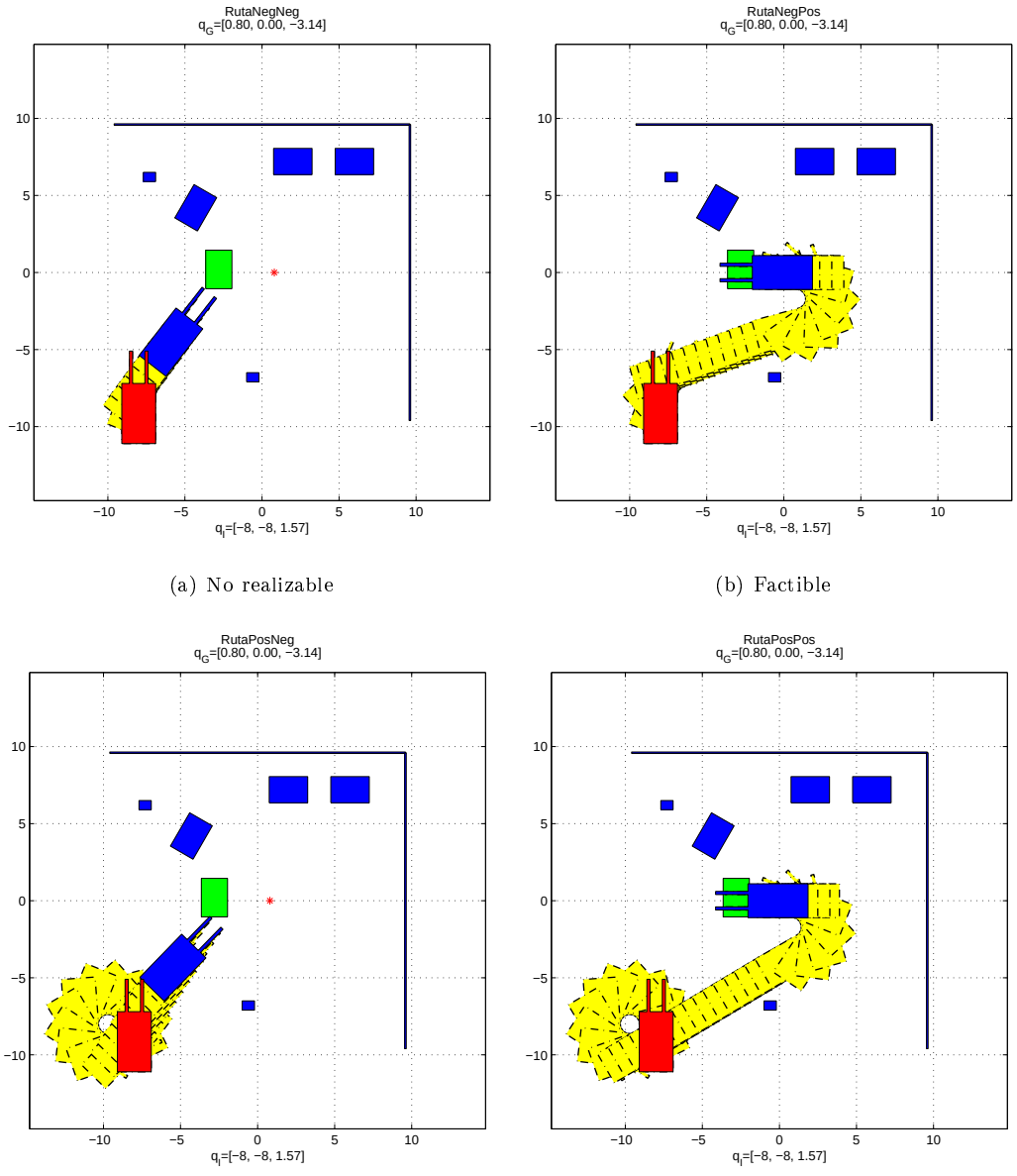
Para el cálculo de los caminos de transferencia, se considerará que en la matriz A^* se representa al robot con el palé en la configuración de agarre. En las figuras 5.27(a) y 5.27(b) se muestran las matrices que representan al robot y al palé, situando el centro del sistema de referencia en el centro del eje que une las ruedas delanteras (ruedas directrices), dependiendo de si el palé es agarrado por su parte ancha o por la estrecha. El cálculo de las curvas de Dubins se realiza de igual forma que para los caminos de tránsito, siendo la configuración inicial el agarre seleccionado y la final la posición en la que se ha de depositar el palé. En cuanto al test de colisiones es necesario calcular de nuevo la representación de los objetos



(a) Factible (b) No realizable

(c) Factible (d) No realizable

Figura 5.24: Rutas del primer grupo de configuraciones de agarre en el orden en el que han sido evaluadas. El camino de tránsito seleccionado por el planificador es el representado en (a).



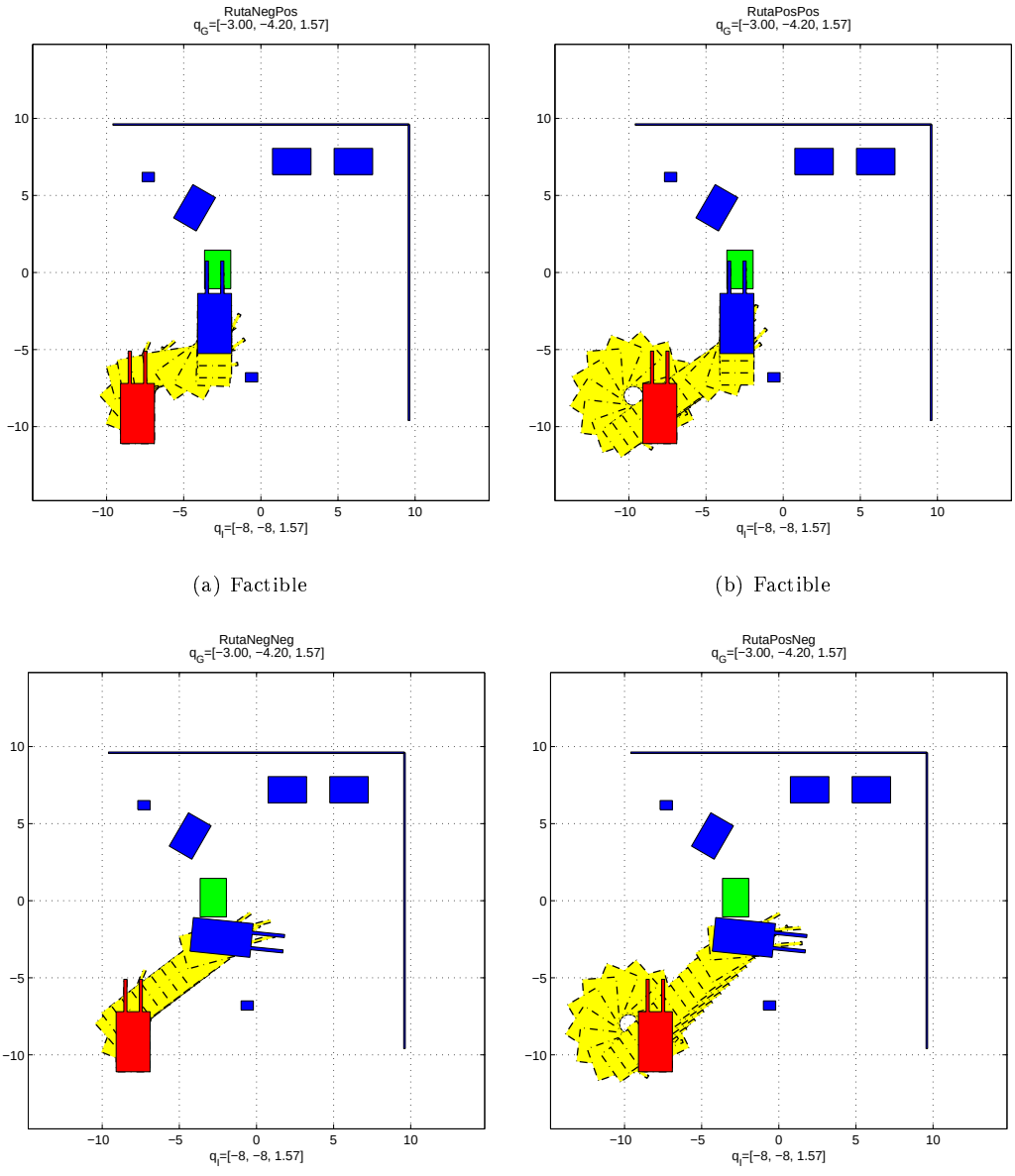
(a) No realizable

(b) Factible

(c) No realizable

(d) Factible

Figura 5.25: Rutas para el segundo grupo de configuraciones de agarre.

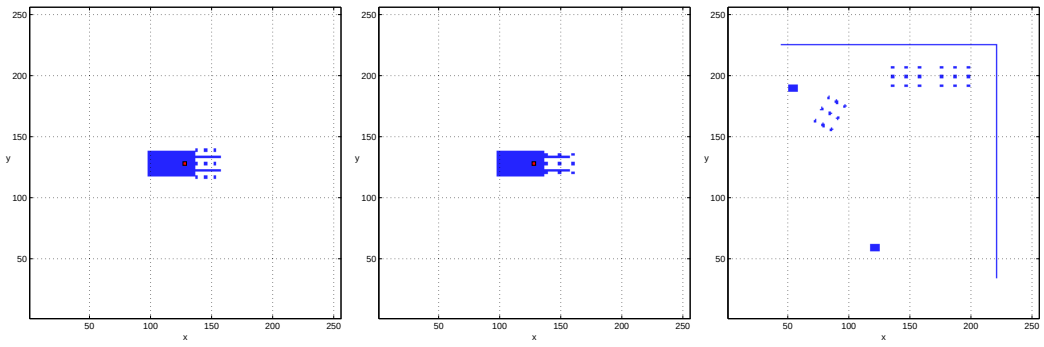


(a) Factible (b) Factible

(c) No realizable (d) No realizable

Figura 5.26: Rutas para la última configuración de agarre.

en el C-espacio, teniendo en cuenta que el robot transporta el palé (A^* en la figura 5.27(a) o (b)) y que en el escenario (E^*) no se considera el palé que se transporta (figura 5.27(c)).



(a) Matriz A^* incluye carretilla y palé

(b) Matriz A^* incluye carretilla y palé

(c) Nuevo escenario E^*

Figura 5.27: Robot y Escenario para los caminos de transferencia.

En la tabla 5.3, se muestran para cada uno de los agarres realizables y en el orden en el que han sido evaluados, las cuatro curvas de Dubins posibles y si son o no realizables. En caso de no ser realizables se muestra la configuración en la que se produciría el choque. Es importante recordar que el planificador devolverá el camino de tránsito hacia la primera celda de esta tabla y posteriormente el camino de transferencia: la ruta LSL que partiendo de $q_I = (-6.7, -0.1, 0.0)$ primero hay que girar toda a la izquierda seguir recto y girar toda a izquierda hasta alcanzar la posición final $q_G = (6.0, 2.5, \frac{\pi}{2})$ (figura 5.28(a)). Al igual que hemos hecho con los caminos de tránsito hemos decidido mostrar los resultados de evaluar todas las curvas posibles para mostrar así la validez de nuestros algoritmos de cálculo de curvas y de detección de colisiones.

	Orden			
q_I	1	2	3	4
(-6.70, -0.10, 0.00)	LSL	LSR	RSL	RSR
	Factible	(6.2, 3.2, 0.25)	Factible	(6.3, 3.9, 0.27)
(0.80, -0.00, $-\pi$)	LSL	RSL	RSR	LSR
	factible	factible	(6.2, 4.1, 0.00)	(6.6, 3.8, 0.73)
(-3.00, -4.20, $\pi/2$)	RSL	LSL	RSR	LSR
	factible	factible	(6.0, 3.4,0.63)	(6.0, 3.3,-0.62)

Tabla 5.3: Rutas factibles (marcadas en rojo) para trasladar el palé a la configuración $q_G = (6.0, 2.5, \frac{\pi}{2})$. Las rutas marcadas en rojo no son realizable porque colisionan con un obstáculo en la configuración mostrada.

En la figura 5.28 se muestran los resultados de evaluar las cuatro curvas de Dubins para el primer grupo de agarres que se produce cuando el robot agarra el palé por la parte ancha desde la izquierda del palé. En la simulación realizada en Matlab se ha decidido representar el palé sólo con sus tacos y de otro color, únicamente para diferenciarlo mejor del robot. Para

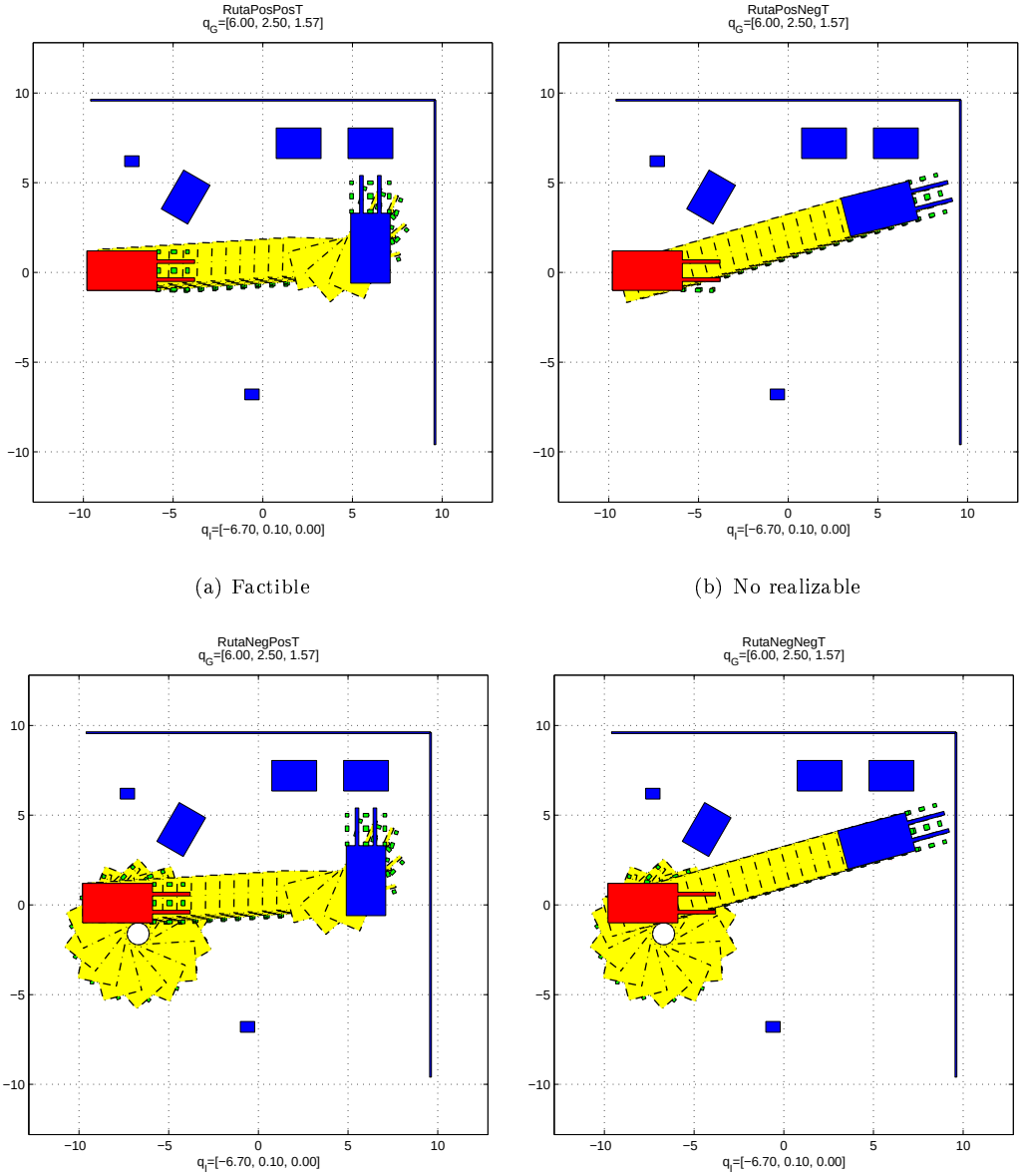
la evaluación de si este grupo de agarres es o no realizable en el algoritmo para el cálculo de la proyección de los objetos en el C-espacio se ha de considerar el robot formado por la carretilla y el palé agarrado por su parte ancha que es el representado en la figura 5.27(a). Observando globalmente la figura 5.28 podemos concluir que las curvas que acaban en giro a derecha (*SR) no son realizables pues se produce colisión con la pared de la derecha (figura 5.28(b) y (d)). Además la curva más corta es en efecto la evaluada en primer lugar y es la que se produce girando levemente a izquierda, siguiendo recto hasta girar finalmente a izquierda y alcanzar la configuración final (figura 5.28(a)). Como ya se ha comentado esta curva formaría el camino de transferencia elegido por nuestro planificador.

Cuando consideramos el segundo grupo de agarres, el realizado por la carretilla desde la derecha y que agarra el palé por su parte ancha, obtenemos la figura 5.29 donde el robot ha de girar completamente, bien a izquierda (figura 5.29(a)) o bien a derecha (5.29(b)), para luego ir recto y alcanzar la posición final girando a izquierda. En este caso la proyección de los objetos en el C-espacio es la calculada teniendo en cuenta el robot y el palé agarrado por su parte ancha (figura 5.27(a)), al igual que para el primer grupo de agarres.

Respecto al tercer y último agarre, destacar que para el calculo de la proyección de los objetos en el C-espacio necesaria en el test de colisiones, se ha de considerar el robot con el palé agarrado por su parte estrecha, que es el representado en la figura 5.27(b). Como puede observarse en la figura 5.30, y al igual que sucede para los anteriores grupos de agarre, las rutas que finalizan en giro a izquierda no son realizables, pues el robot colisiona con la pared de la derecha (figuras 5.30(c) y (d)). La curva que se inicia con un giro a derecha (figura 5.30(a)) es la más corta y, por ello, se evalúa en primer lugar.

5.6. Aportaciones

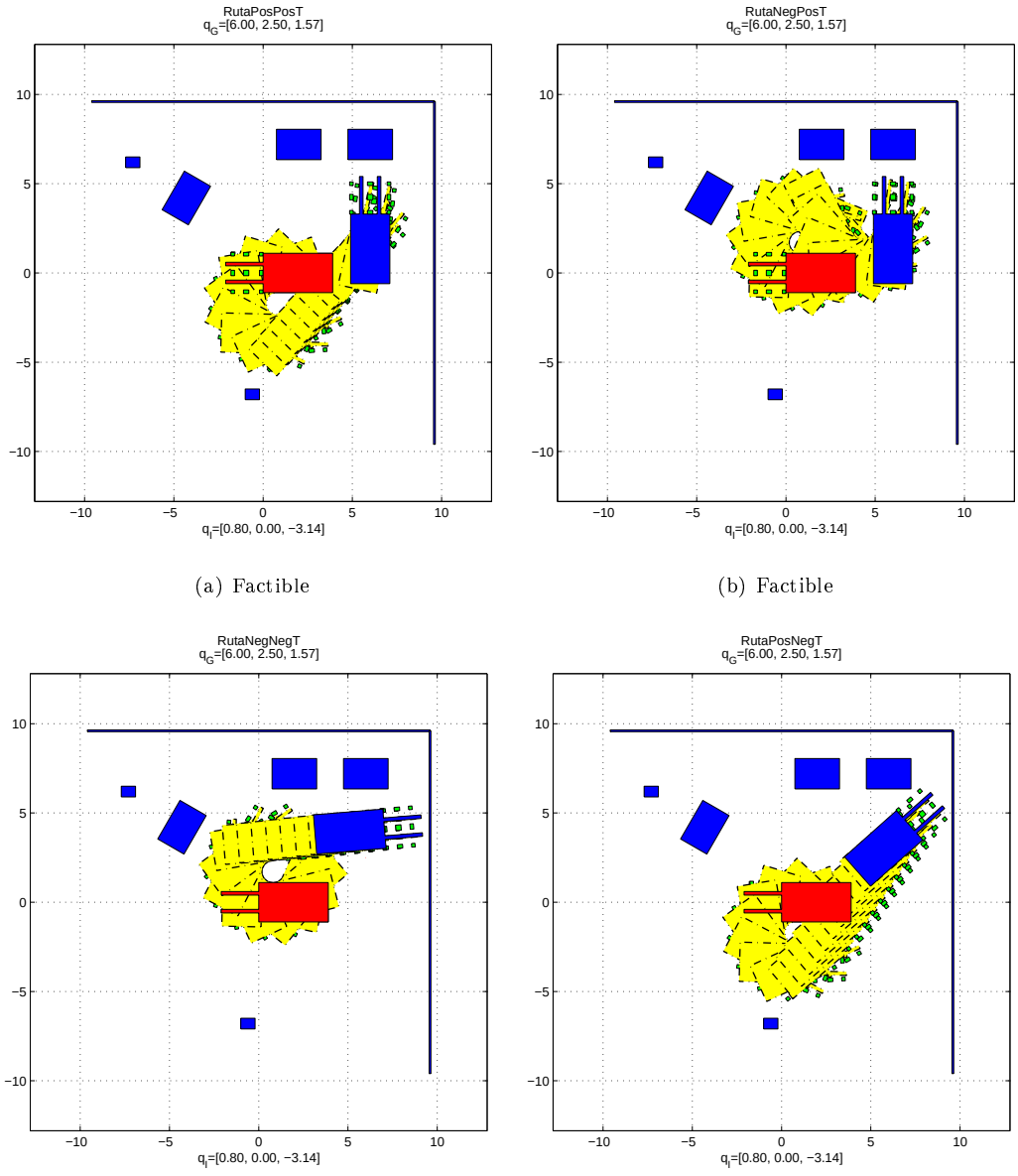
En este capítulo se han aplicado los resultados de nuestra propuesta de síntesis de agarres a la manipulación que realiza un vehículo paletizador que ha sido robotizado por el grupo de Robótica y Sociedad de la Universidad de Salamanca. Para la planificación de trayectorias en un sistema no holónomo como es nuestro robot, hemos propuesto un nuevo algoritmo para el cálculo de las curvas de Dubins que permite obtener trayectorias óptimas y realizables. Además hemos propuesto y validado mediante simulación un algoritmo completo de manipulación que integra las tareas de síntesis de agarres y planificación de caminos. Calcula los agarres de mejor calidad y los caminos de tránsito y transferencia para cumplir la tarea de manipulación.



(a) Factible (b) No realizable

(c) Factible (d) No realizable

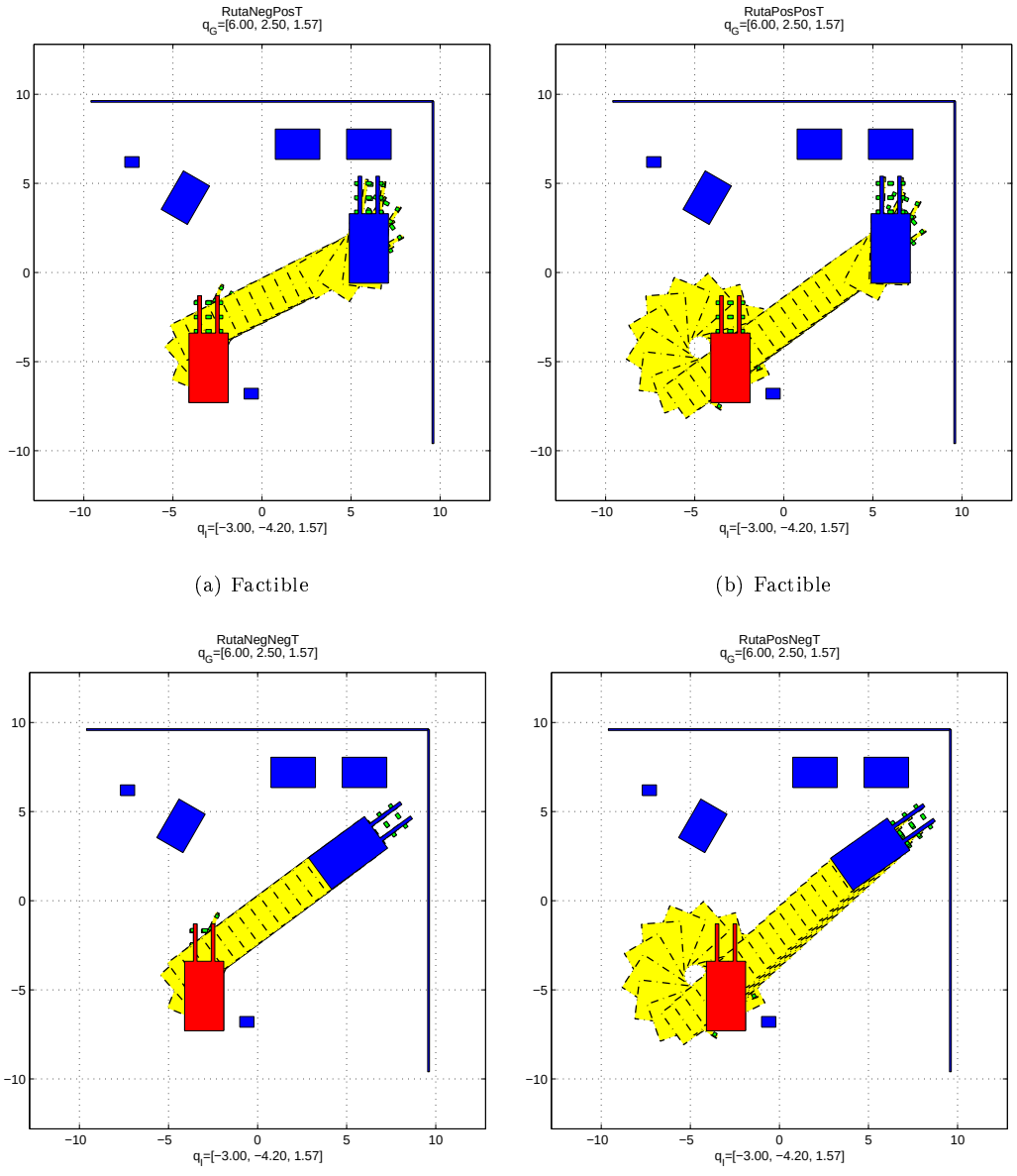
Figura 5.28: Caminos de transferencia desde el primer grupo de agarres en el orden en el que han sido evaluados.



(a) Factible (b) Factible

(c) No realizable (d) No realizable

Figura 5.29: Caminos de transferencia desde el segundo grupo de agarres.



(a) Factible (b) Factible

(c) No realizable (d) No realizable

Figura 5.30: Caminos de transferencia desde el último grupo de agarres.

Puesto que la carga computacional de nuestros algoritmos de manipulación no depende de la geometría y del número de objetos en el entorno, sino únicamente de la resolución con que se trabaje, es posible aplicarlos a la manipulación de objetos no preestablecida en entornos no estructurados flexibilizando así la tarea de manipulación de un AGV.

6

Conclusiones y trabajos futuros

Las principales aportaciones y conclusiones de este trabajo de tesis son las siguientes:

- Se ha propuesto un nuevo formalismo matemático para la síntesis de agarres, en el espacio de las configuraciones de un robot, según determinados criterios de calidad. Las configuraciones de agarre de la pinza se encuentran en la denominada *frontera de agarre*. Las expresiones centrales de este formalismo permiten calcular la frontera de agarre con dos funciones CE y ACE. La primera de ellas se calcula mediante la integral, extendida sobre el espacio de trabajo, del producto de dos funciones: una que describe al robot y la otra al entorno de trabajo donde existen objetos movibles o manipulables por el robot y obstáculos (objetos fijos). Con esta integral se obtiene la representación de los objetos en el C-espacio. La segunda función ACE calcula el *C-espacio aumentado* como el producto de dos funciones: una que describe al robot y la otra al entorno de trabajo considerando los objetos movibles y los obstáculos junto con su frontera.

  Además, las funciones que permiten obtener la *frontera de agarre* se pueden definir

para que las configuraciones de agarre calculadas cumplan determinados criterios de calidad de agarre o en función de la tarea que se va a realizar con la pieza.

- Basado en el formalismo matemático se ha presentado un método de trabajo que con la elección adecuada de los sistema de referencia y las coordenadas de trabajo, directamente relacionados ambos con la estructura mecánica de la pinza, deriva en que las dos funciones CE y ACE, que permiten calcular la *frontera de agarre* mediante la convolución.

  El método se ha aplicado a varios robots/pinzas con diferentes grados de libertad y se han obtenido las expresiones matemáticas que permiten calcular ambas funciones mediante la convolución.

  Esto permitirá reducir la complejidad computacional de los algoritmos que se deriven de las expresiones matemáticas, pues permitirá utilizar como herramienta matemática de la transformada de Fourier en lugar de la convolución y, como solución algorítmica la Transformada Rápida de Fourier.

- Se ha puesto de manifiesto que la complejidad computacional en todos los casos es independiente de la forma geométrica de los objetos que constituyen el robot o de los que están presentes en su entorno de trabajo. Con ello se ha logrado que la complejidad computacional no dependa del número de vértices de los objetos ni de si son cóncavos o convexos, como sucede en la mayoría de los trabajos existentes sino únicamente de la resolución con la que se realice la discretización.

- Se han propuesto y se han implementado los algoritmos para la obtención de los *mejores* agarres, según diferentes criterios de calidad, para robots/pinzas con distintas configuraciones cinemáticas. Los algoritmos propuestos trabajan con mapas de bits que representan al robot y al entorno de trabajo, que incluye los objetos movibles y los obstáculos. Además de los mejores agarres, obtiene un mapa de bits que representa al espacio de las configuraciones. El tiempo de ejecución de estos algoritmos depende, en cada caso, únicamente de la resolución con la que se trabaje y es independiente del número y de la forma geométrica de los objetos movibles y de los obstáculos.

  Debido al carácter paralelizable de los algoritmos que calculan la transformada rápida de Fourier y a la fácil paralelización de los algoritmos propuestos se puede reducir de forma drástica el tiempo de cálculo, pudiendo aplicarse a entornos de trabajo que varían dinámicamente.

- El método de síntesis de agarres permite la incorporación al planificador de agarres de

información procedente de métodos de análisis de agarres, fijando zonas preferidas de agarre tanto en el robot como en el objeto, sin que esto suponga incremento alguno en su carga computacional. La especificación de zonas de agarre en el robot vendrán típicamente determinadas por la forma de su efector final, mientras que en el objeto dependerá de la tarea que se desee realizar con él. De esta forma, las configuraciones de agarre obtenidas con nuestro método cumplirán los criterios de calidad impuestos en el análisis.

- Los agarres calculados con nuestro método de síntesis son accesibles. Solo los agarres que no colisionen con los obstáculos del entorno serán seleccionados. En el proceso de cálculo de los mejores agarres se obtiene además una estructura de datos con la representación de los obstáculos del entorno en el espacio de las configuraciones o C-Obstáculo. Esta estructura servirá al planificador de caminos para realizar un test de colisiones sin coste computacional, pues la determinación de si las configuraciones que integran un camino colisionan o no con algún obstáculo, se reduce únicamente a comprobar si en dicha estructura la configuración correspondiente tiene un valor 0 ó 1.

- Además del nuevo método de síntesis de agarres, en este trabajo de investigación también se ha abordado el tema de la planificación de trayectorias. Así, se ha propuesto un nuevo algoritmo para el cálculo de las curvas de Dubins que permite obtener trayectorias óptimas para un robot sujeto a restricciones no holónomas, como son los robots móviles tipo coche.

- Como resultado final, se ha propuesto un algoritmo de planificación de manipulaciones que integra la determinación de las configuraciones de agarre, la planificación de trayectorias y un test de colisiones que verifica si las curvas de Dubins son o no realizables. Este algoritmo se ha planteado para un robot tipo coche, y más concretamente para una carretilla paletizadora que ha sido robotizada por el grupo de Robótica.

  El planificador de manipulaciones, a partir de las configuraciones de agarre que cumplen los criterios de calidad propuesto, calcula tanto caminos de tránsito, donde el robot viaja solo desde la configuración inicial a la configuración de agarre, como los caminos de transferencia donde el robot transporta consigo la pieza. Dado que existen distintas soluciones, es decir, distintas curvas de Dubins, el planificador determina las trayectorias óptimas que son realizables.

  Para realizar el test de colisiones sólo es necesario consultar en el C-espacio si las configuraciones que componen las curvas de Dubins pertenecen al espacio libre o al ocupado

por los obstáculos. Como el cálculo del C-espacio es parte del proceso de determinación de los mejores agarres, el planificador de trayectorias no necesita calcular de nuevo el C-espacio que se utilizará para la detección de colisiones. Esta es una de las ventajas de nuestro método de síntesis de agarres que no se encuentra en otros métodos, donde la planificación de agarres y de trayectorias son procesos completamente diferentes.

- El entorno de trabajo donde el robot paletizador ha de manipular palés de forma autónoma ha sido simulado en Matlab. En esta simulación se ha incluido nuestro planificador de manipulaciones, permitiendo validar de forma integrada nuestra propuesta de síntesis de agarres y de planificación de trayectorias libres de colisiones.

- Hemos realizado una propuesta integral de manipulación robótica para robots autónomos y, más concretamente, para vehículos guiados automáticamente (AGV). Puesto que la carga computacional de nuestros algoritmos de manipulación no depende de la geometría y del número de objetos en el entorno, sino únicamente de la resolución con que se trabaje, se pueden aplicar a la manipulación de objetos no preestablecida en entornos no estructurados, flexibilizando así la tarea de manipulación de un AGV.

A partir de las conclusiones se pueden destacar algunos puntos en los que es posible extender el trabajo presentado:

- Implementar los algoritmos sobre dispositivos de cálculo especializados de alto rendimiento.

- Proponer un algoritmo para planificar las trayectorias con las curvas de Reeds-Shepp que se diferencian de las de Dubins en que el robot puede ir también marcha atrás.

- Integrar el planificador de manipulaciones propuesto en la arquitectura de control de la carretilla real y en entornos de simulación en 3D como por ejemplo Gazebo.

Apéndices

A

Algoritmos detallados para cálculo de las curvas de Dubins

Algoritmo A.1: Algoritmo detallado para la obtención de las curvas de Dubins

Entrada: Configuración inicial ($q_I = (x_i, y_i, \theta_i)$), configuración final ($q_F = (x_f, y_f, \theta_f)$), velocidad ($v$) y radio mínimo de giro (r)

Salida: Curvas de Dubins ($Ruta_{pp}, Ruta_{nn}, Ruta_{np}, Ruta_{pn}$)

1 **Inicio**

2 $\Delta s = v * \Delta t \leftarrow$ Obtener el desplazamiento;

3 $\Delta\theta = (v/r) * \Delta t \leftarrow$ Obtener el angulo de giro;

`//Construir la circunferencia del giro positivo de salida`

4 $X_{POS\_C\_out} = (x_i - r\cos(\theta_i - \frac{\pi}{2}),\ y_i - r\sin(\theta_i - \frac{\pi}{2}))$; `//El centro`

5 $CIRC\_POS_{out} = (X_{POS\_C\_out} + (r\cos\theta,\ r\sin\theta)),, \theta \in [0, 2\pi)$; `//La circunferencia`

`//Los valores de` θ `son valores discretizados con incrementos de` $\Delta\theta$

`//Construir la circunferencia del giro positivo de llegada`

6 $X_{POS\_C\_in} = (x_f - r\cos(\theta_f - \frac{\pi}{2}),\ y_f - r\sin(\theta_f - \frac{\pi}{2}))$; `//El centro`

7 $CIRC\_POS_{in} = (X_{POS\_C\_in} + (r\cos\theta,\ r\sin\theta)),, \theta \in [0, 2\pi)$; `//La circunferencia`

`//Construir la circunferencia del giro negativo de salida`

8 $X_{NEG\_C\_out} = (x_i - r\cos(\theta_i + \frac{\pi}{2}),\ y_i - r\sin(\theta_i + \frac{\pi}{2}))$; `//El centro`

9 $CIRC\_NEG_{out} = (X_{NEG\_C\_out} + (r\cos\theta,\ r\sin\theta)),, \theta \in [0, 2\pi)$; `//La circunferencia`

`//Construir la circunferencia del giro negativo de llegada`

10 $X_{NEG\_C\_in} = (x_f - r\cos(\theta_f + \frac{\pi}{2}),\ y_f - r\sin(\theta_f + \frac{\pi}{2}))$; `//El centro`

11 $CIRC\_NEG_{in} = (X_{NEG\_C\_in} + (r\cos\theta,\ r\sin\theta)),, \theta \in [0, 2\pi)$; `//La circunferencia`

12 $Ruta_{pp}$ = LSL();

13 $Ruta_{nn}$ = RSR();

14 $Ruta_{pn}$ = LSR();

15 $Ruta_{np}$ = RSL();

16 Curvas de Dubins $\leftarrow$ Ordenar las rutas ($Ruta_{xx}$) primero la mas corta;

Algoritmo A.2: Cálculo de la curva de Dubins LSL (figura A.1)

Salida: $Ruta_{pp}$

Inicio

`//Determina la dirección de conexión`

$\overrightarrow{u_{pp}} = \frac{(X_{POS\_C\_in} - X_{POS\_C\_out})}{\left\|X_{POS\_C\_in} - X_{POS\_C\_out}\right\|}$;

$\theta_{pp} = \arctan(\overrightarrow{u_{pp}})$;

$\theta_{pp\_out} = \theta_{pp} - \pi/2$;

`//Determina las tangentes a las circunferencias (`α `salida,` β `entrada)`

$\overrightarrow{u_{pp_r}} = \frac{(CIRC\_POS_{out} - X_{POS\_C\_out})}{\left\|CIRC\_POS_{out} - X_{POS\_C\_out}\right\|}$; `//Los radios`

$\overrightarrow{u_{pp_r}}(\alpha_{pp\_1}) \cdot \overrightarrow{u_{pp}} = 0;$

$\overrightarrow{u_{pp_r}}(\alpha_{pp\_2}) \cdot \overrightarrow{u_{pp}} = 0;$

`//Selecciona la tangente de conexión como la más próxima a` θ_{pp_out}

$\alpha_{pp}, , |\alpha_{pp} - \theta_{pp\_out}| = min_{i\in 1,2}|\alpha_{pp\_i} - \theta_{pp\_out}|;$

$\overrightarrow{u_{pp_r}}(\beta_{pp\_1}) \cdot \overrightarrow{u_{pp}} = 0;$

$\overrightarrow{u_{pp_r}}(\beta_{pp\_2}) \cdot \overrightarrow{u_{pp}} = 0;$

`//Selecciona la tangente de conexión como la más próxima a` θ_{pp_out}

$\beta_{pp}, , |\beta_{pp} - \theta_{pp\_out}| = min_{i\in 1,2}|\beta_{pp\_i} - \theta_{pp\_out}|;$

`//Determina recta de conexión`

$A_{pp} \in CIRC\_POS_{out}, , A_{pp} = X_{POS\_C\_out} + (r\cos\alpha_{pp}, r\sin\alpha_{pp});$

$B_{pp} \in CIRC\_POS_{in}, , B_{pp} = X_{POS\_C\_in} + (r\cos\beta_{pp}, r\sin\beta_{pp});$

$S = \overline{A_{pp}B_{pp}}$; `//Puntos de la recta que une` A_{pp} `y` B_{pp}

`//Construye trayectoria`

`//Determina orientación de salida`

$\theta_{out} = \theta_i - \frac{\pi}{2} \ , , \theta_{out} \in [0, 2\pi)$; `//Posición angular en la circunferencia de salida`

`//Construye L (giro positivo o a izquierda) de salida`

if $\theta_{out} < \theta_{pp\_out}$ **then**

  $L1 = CIRC\_POS_{out}(\theta_{out}, \theta_{pp\_out});$

else

  $L1_1 = CIRC\_POS_{out}(\theta_{out}, 2\pi);$

  $L1_2 = CIRC\_POS_{out}(0, \theta_{pp\_out});$

  $L1 = concatenar(L1_1, L1_2);$

$Ruta_{pp} = concatenar(L1, S)$; `//Añade S (la recta)`

`//Determina orientación de entrada`

$\theta_{in} = \theta_f - \frac{\pi}{2} \ , , \theta_{in} \in [0, 2\pi)$; `//Posición angular en la circunferencia de entrada`

`//Construye L (giro a izquierdas o positivo) de llegada`

if $\theta_{pp\_out} < \theta_{in}$ **then**

  $L2 = CIRC\_POS_{in}(\theta_{pp\_out}, \theta_{in});$

else

  $L2_1 = CIRC\_POS_{in}(\theta_{pp\_out}, 2\pi);$

  $L2_2 = CIRC\_POS_{in}(0, \theta_{in});$

  $L2 = concatenar(L2_1, L2_2);$

$Ruta_{pp} = concatenar(Ruta_{pp}, L2)$; `//Añade L (giro positivo o a izquierda) de llegada`

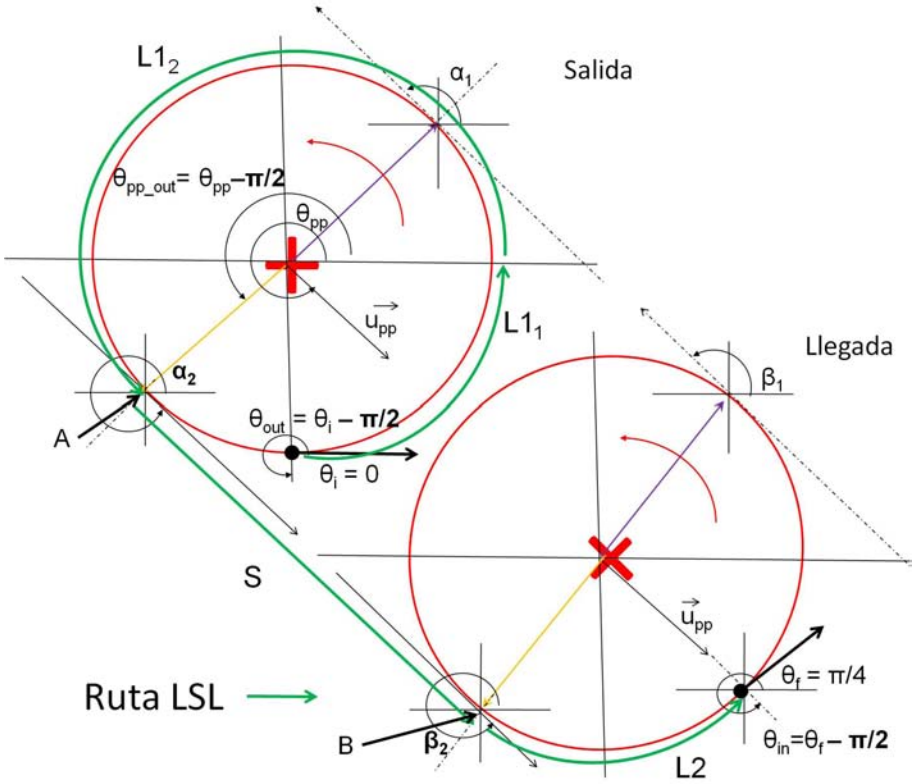


Figura A.1: Esquema del cálculo de la curva de Dubins LSL.

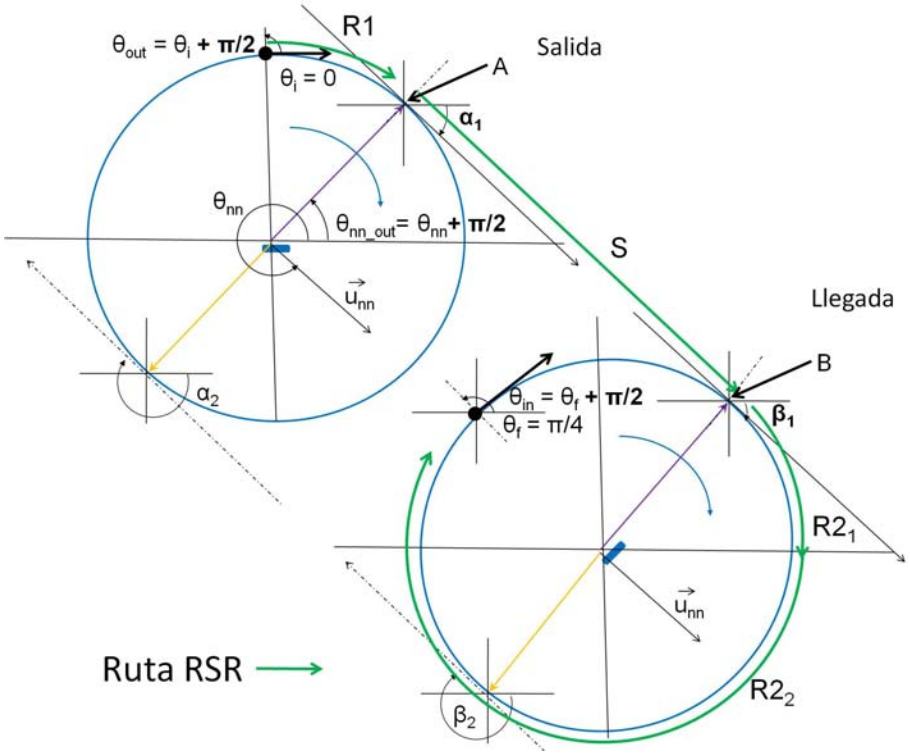


Figura A.2: Esquema del cálculo de la curva de Dubins RSR.

Algoritmo A.3: Cálculo de la curva de Dubins RSR (figura A.2)

Salida: $Ruta_{nn}$

Inicio

`//Determina dirección de conexión`

$\overrightarrow{u_{nn}} = \frac{(X_{NEG\_C\_in} - X_{NEG\_C\_out})}{\left\| X_{NEG\_C\_in} - X_{NEG\_C\_out} \right\|}$;

$\theta_{nn} = \arctan(\overrightarrow{u_{nn}})$;

$\theta_{nn\_out} = \theta_{nn} + \pi/2$;

`//Determina las tangentes a las circunferencias (` α `salida,` β `entrada)`

$\overrightarrow{u_{nn_r}} = \frac{(CIRC\_NEG_{out} - X_{NEG\_C\_out})}{\left\| CIRC\_NEG_{out} - X_{NEG\_C\_out} \right\|}$; `//Los radios`

$\overrightarrow{u_{nn_r}}(\alpha_{nn\_1}) \cdot \overrightarrow{u_{nn}} = 0;$

$\overrightarrow{u_{nn_r}}(\alpha_{nn\_2}) \cdot \overrightarrow{u_{nn}} = 0;$

`//Selecciona la tangente de conexión como la más próxima a` θ_{nn_out}

$\alpha_{nn}, , |\alpha_{nn} - \theta_{nn\_out}| = min_{i \in 1,2} |\alpha_{nn\_i} - \theta_{nn\_out}|$

$\overrightarrow{u_{nn_r}}(\beta_{nn\_1}) \cdot \overrightarrow{u_{nn}} = 0;$

$\overrightarrow{u_{nn_r}}(\beta_{nn\_2}) \cdot \overrightarrow{u_{nn}} = 0;$

`//Selecciona la tangente de conexión como la más próxima a` θ_{nn_out}

$\beta_{nn}, , |\beta_{nn} - \theta_{nn\_out}| = min_{i \in 1,2} |\beta_{nn\_i} - \theta_{nn\_out}|;$

`//Determina recta de conexión`

$A_{nn} \in CIRC\_NEG_{out}, , A_{nn} = X_{NEG\_C\_out} + (r \cos \alpha_{nn}, r \sin \alpha_{nn});$

$B_{nn} \in CIRC\_NEG_{in}, , B_{nn} = X_{NEG\_C\_in} + (r \cos \beta_{nn}, r \sin \beta_{nn});$

$S = \overline{A_{nn} B_{nn}}$; `//Puntos de la recta que une` A_{nn} `y` B_{nn}

`//Construye trayectoria`

`//Determina orientación de salida`

$\theta_{out} = \theta_i + \frac{\pi}{2} \ , , \theta_{out} \in [0, 2\pi)$; `//Posición angular en la circunferencia de salida`

`//Construye R (giro negativo o a derecha) de salida`

if $\theta_{out} > \theta_{nn\_out}$ **then**

  $R1 = CIRC\_NEG_{out}(\theta_{out}, \theta_{nn\_out});$ `//Recorrido inverso`

else

  $R1_1 = CIRC\_NEG_{out}(\theta_{out}, 0);$ `//Recorrido inverso`

  $R1_2 = CIRC\_NEG_{out}(2\pi, \theta_{nn\_out});$ `//Recorrido inverso`

  $R1 = concatenar(R1_1, R1_2);$

$Ruta_{nn} = concatenar(R1, S)$; `//Añade S (la recta)`

`//Determina orientación de entrada`

$\theta_{in} = \theta_f + \frac{\pi}{2} \ , , \theta_{in} \in [0, 2\pi)$; `//Posición angular en la circunferencia de entrada`

`//Construye R (giro a derechas o negativo) de llegada`

if $\theta_{nn\_out} > \theta_{in}$ **then**

  $R2 = CIRC\_NEG_{in}(\theta_{nn\_out}, \theta_{in});$ `//Recorrido inverso`

else

  $R2_1 = CIRC\_NEG_{in}(\theta_{nn\_out}, 0);$ `//Recorrido inverso`

  $R2_2 = CIRC\_NEG_{in}(2\pi, \theta_{in});$ `//Recorrido inverso`

  $R2 = concatenar(R2_1, R2_2);$

$Ruta_{nn} = concatenar(Ruta_{nn}, R2)$; `//Añade R (giro negativo o a derecha) de llegada`

Algoritmo A.4: Cálculo de la curva de Dubins RSL (figura A.3)

Salida: $Ruta_{np}$

Inicio

//Determina dirección de conexión

$\overrightarrow{u_{np}} = \frac{(X_{POS\_C\_in} - X_{NEG\_C\_out})}{\left\| X_{POS\_C\_in} - X_{NEG\_C\_out} \right\|}$;

$\overrightarrow{u_{np_r}} = \frac{(CIRC\_NEG_{out} - X_{NEG\_C\_out})}{\left\| CIRC\_NEG_{out} - X_{NEG\_C\_out} \right\|}$; //Los radios

$\theta_{np} = \arctan(\overrightarrow{u_{np}})$;

$\theta_{np\_out} = \theta_{np} + \pi/2$;

//Determina las tangentes a las circunferencias (α salida, β entrada)

$\overrightarrow{u_{np_r}}(\alpha_{np\_1}) \cdot \overrightarrow{u_{np}} = 0;$

$\overrightarrow{u_{np_r}}(\alpha_{np\_2}) \cdot \overrightarrow{u_{np}} = 0;$

//Selecciona la tangente de conexión como la más próxima a θ_{np_out}

$\alpha_{np}, , |\alpha_{np} - \theta_{np\_out}| = min_{i \in 1,2} |\alpha_{np\_i} - \theta_{np\_out}|$

$\overrightarrow{u_{np}}(\beta_{np\_1}) \cdot \overrightarrow{u_{np}} = 0;$

$\overrightarrow{u_{np}}(\beta_{np\_2}) \cdot \overrightarrow{u_{np}} = 0;$

//Selecciona la tangente de conexión como la más próxima a θ_{np_out}

$\beta_{np}, , |\beta_{np} - \theta_{np\_out}| = min_{i \in 1,2} |\beta_{np\_i} - \theta_{np\_out}|;$

//Determina recta de conexión

$A_{np} \in CIRC\_NEG_{out}, , A_{np} = X_{NEG\_C\_out} + (r \cos \alpha_{np}, r \sin \alpha_{np});$

$B_{np} \in CIRC\_POS_{in}, , B_{np} = X_{POS\_C\_in} + (r \cos \beta_{np}, r \sin \beta_{np});$

$S = \overline{A_{np} B_{np}}$; //Puntos de la recta que une A_{np} y B_{np}

//Construye trayectoria

//Determina orientación de salida

$\theta_{out} = \theta_i + \frac{\pi}{2} \ , , \theta_{out} \in [0, 2\pi)$; //Es la posición angular en la circunferencia de salida del robot

//Construye R (giro negativo o a derecha) de salida

if $\theta_{out} > \theta_{np\_out}$ **then**

  $R = CIRC\_NEG_{out}(\theta_{out}, \theta_{np\_out});$ //Recorrido inverso

else

  $R_1 = CIRC\_NEG_{out}(\theta_{out}, 0);$ //Recorrido inverso

  $R_2 = CIRC\_NEG_{out}(2\pi, \theta_{np\_out});$ //Recorrido inverso

  $R = concatenar(R_1, R_2);$

$Ruta_{np} = concatenar(R, S)$; //Añade S (la recta)

//Determina orientación de entrada

$\theta_{in} = \theta_f - \frac{\pi}{2} \ , , \theta_{in} \in [0, 2\pi)$; //Es la posición angular en la circunferencia de entrada del robot

//Construye L (giro a izquierdas o positivo) de llegada

if $\theta_{np\_out} < \theta_{in}$ **then**

  $L = CIRC\_POS_{in}(\theta_{np\_out}, \theta_{in});$

else

  $L_1 = CIRC\_POS_{in}(\theta_{np\_out}, 2\pi);$

  $L_2 = CIRC\_POS_{in}(0, \theta_{in});$

  $L = concatenar(L_1, L_2);$

$Ruta_{np} = concatenar(Ruta_{np}, L)$; //Añade L (giro positivo o a izquierda) de llegada

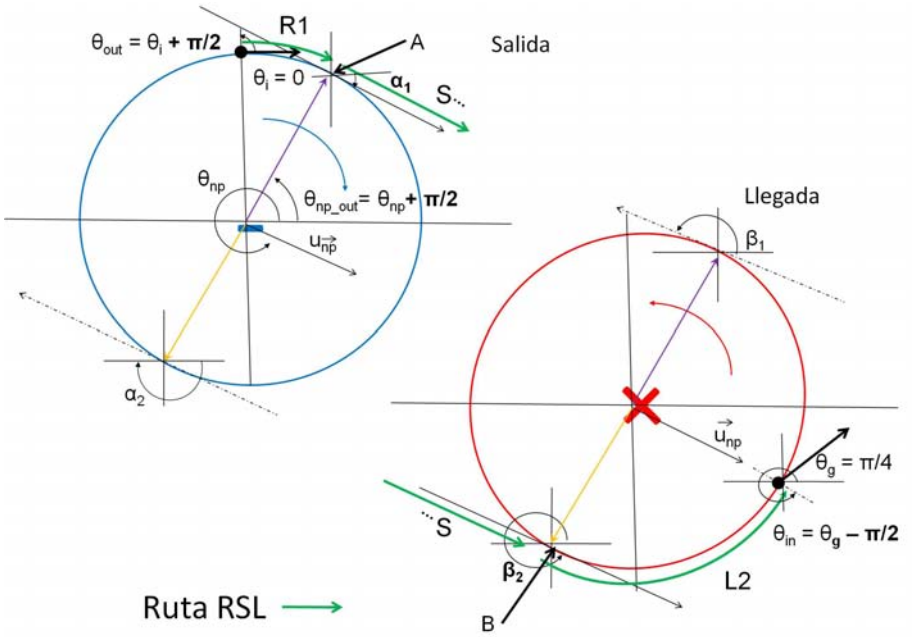


Figura A.3: Esquema del cálculo de la curva de Dubins RSL.

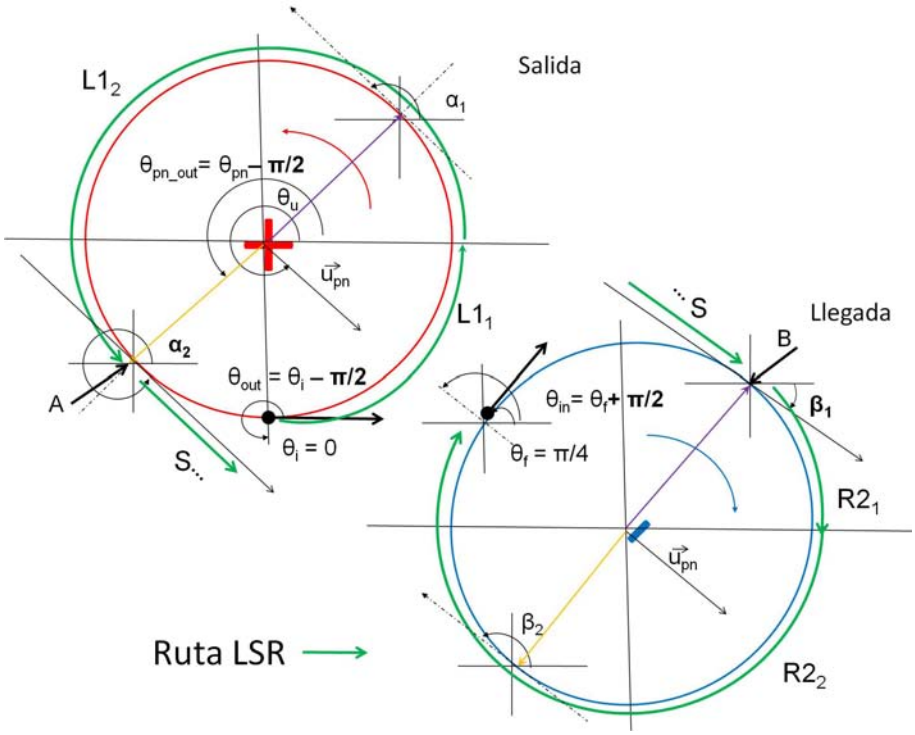


Figura A.4: Esquema del cálculo de la curva de Dubins LSR.

Algoritmo A.5: Cálculo de la curva de Dubins LSR (figura A.4)

Salida: $Ruta_{pn}$

1 **Inicio**

//Determina dirección de conexión

2 $\overrightarrow{u_{pn}} = \frac{(X_{NEG\_C\_in} - X_{POS\_C\_out})}{\left\| X_{NEG\_C\_in} - X_{POS\_C\_out} \right\|}$;

3 $\overrightarrow{u_{pn_r}} = \frac{(CIRC\_POS_{out} - X_{POS\_C\_out})}{\left\| CIRC\_POS_{out} - X_{POS\_C\_out} \right\|}$; //Los radios

4 $\theta_{pn} = \arctan(\overrightarrow{u_{pn}})$;

5 $\theta_{pn\_out} = \theta_{pn} - \pi/2$;

//Determina las tangentes a las circunferencias (α salida, β entrada)

6 $\overrightarrow{u_{pn_r}}(\alpha_{pn\_1}) \cdot \overrightarrow{u_{pn}} = 0$;

7 $\overrightarrow{u_{pn_r}}(\alpha_{pn\_2}) \cdot \overrightarrow{u_{pn}} = 0$;

//Selecciona la tangente de conexión como la más próxima a θ_{pn_out}

8 $\alpha_{pn}, , |\alpha_{pn} - \theta_{pn\_out}| = min_{i \in 1,2} |\alpha_{pn\_i} - \theta_{pn\_out}|$

9 $\overrightarrow{u_{pn_r}}(\beta_{pn\_1}) \cdot \overrightarrow{u_{pn}} = 0$;

10 $\overrightarrow{u_{pn_r}}(\beta_{pn\_2}) \cdot \overrightarrow{u_{pn}} = 0$;

//Selecciona la tangente de conexión como la más próxima a θ_{pn_out}

11 $\beta_{pn}, , |\beta_{pn} - \theta_{pn\_out}| = min_{i \in 1,2} |\beta_{pn\_i} - \theta_{pn\_out}|$;

//Determina recta de conexión

12 $A_{pn} \in CIRC\_POSout, , A_{pn} = X_{POS\_C\_out} + (r \cos \alpha_{pn}, r \sin \alpha_{pn})$;

13 $B_{pn} \in CIRC\_NEG_{in}, , B_{pn} = X_{NEG\_C\_in} + (r \cos \beta_{pn}, r \sin \beta_{pn})$;

14 $S = \overline{A_{pn} B_{pn}}$; //Puntos de la recta que une A_{pn} y B_{pn}

//Construye trayectoria

//Determina orientación de salida

15 $\theta_{out} = \theta_i - \frac{\pi}{2} \ , , \theta_{out} \in [0, 2\pi)$; //Posición angular en la circunferencia de salida

//Construye L (giro positivo o a izquierda) de salida

16 **if** $\theta_{out} < \theta_{pn\_out}$ **then**

17 $L = CIRC\_POS_{out}(\theta_{out}, \theta_{pn\_out})$;

18 **else**

19 $L_1 = CIRC\_POS_{out}(\theta_{out}, 2\pi)$;

20 $L_2 = CIRC\_POS_{out}(0, \theta_{pp\_out})$;

21 $L = concatenar(L_1, L_2)$;

22 $Ruta_{pn} = concatenar(L, S)$; //Añade S (la recta)

//Determina orientación de entrada

23 $\theta_{in} = \theta_f + \frac{\pi}{2} \ , , \theta_{in} \in [0, 2\pi)$; //Posición angular en la circunferencia de entrada

//Construye R (giro a derechas o negativo) de llegada

24 **if** $\theta_{pn\_out} > \theta_{in}$ **then**

25 $R = CIRC\_NEG_{in}(\theta_{nn\_out}, \theta_{in})$; //Recorrido inverso

26 **else**

27 $R_1 = CIRC\_NEG_{in}(\theta_{pn\_out}, 0)$; //Recorrido inverso

28 $R_2 = CIRC\_NEG_{in}(2\pi, \theta_{in})$; //Recorrido inverso

29 $R = concatenar(R_1, R_2)$;

30 $Ruta_{pn} = concatenar(Ruta_{pn}, R)$; //Añade R (giro negativo o a derecha) de llegada

Bibliografía

[AGM98] Juan-Manuel Ahuactzin, Kamal Gupta, and Emmanuel Mazer. Manipulation planning for redundant robots: A practical approach. *The International Journal of Robotics Research*, 17(7):731–747, 1998.

[ALS95] Rachid Alami, Jean-Paul Laumond, and Thierry Simeón. Two manipulation planning algorithms. In *WAFR: Proceedings of the workshop on Algorithmic foundations of robotics*, pages 109–125, Natick, MA, USA, 1995. A. K. Peters, Ltd.

[ASL90] Rachid Alami, Thierry Simeón, and Jean-Paul Laumond. A geometrical approach to planning manipulation tasks. the case of discrete placements and grasps. In *The Fifth International Symposium on Robotics Research*, pages 453–463, Cambridge, MA, USA, 1990. MIT Press.

[BATM93] Pierre Bessiere, Juan-Manuel Ahuactzin, El-Ghazali Talbi, and Emmanuel Mazer. The ariadne's clew algorithm: Global planning with local mehtods. In *Proceeding of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Yokohama, Japan, 1993.

[BDH96] C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996.

[BFH99] C. Borst, M. Fischer, and G. Hirzinger. A fast and robust grasp planner for arbitrary 3d objects. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, pages 1890–1896, May 1999.

[BFH03] C. Borst, M. Fischer, and G. Hirzinger. Grasping the dice by dicing the grasp. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 3692–3697, October 2003.

[BFH04] C. Borst, M. Fischer, and G. Hirzinger. Grasp planning: how to choose a suitable task wrench space. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 1, pages 319–325, April 2004.

[Bic95] Antonio Bicchi. On the closure properties of robotic grasping. *International Journal of Robotics Research*, 14(4):319–334, August 1995.

[Bic00] Antonio Bicchi. Hands for dexterous manipulation and robust grasping: a difficult road toward simplicity. *IEEE Transactions on Robotics and Automation*, 16(6):652–662, December 2000.

[BK00] Antonio Bicchi and Vijay Kumar. Robotic grasping and contact: a review. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 348–353, 2000.

[Bla03] Francisco Javier Blanco. *Cálculo del C-espacio de un robot móvil: convolución jerárquica*. PhD thesis, Dpto. de Informática y Automática. Universidad de Salamanca, jun 2003.

[BM90] Randy C. Brost and Matthew T. Mason. Graphical analysis of planar rigid-body dynamics with multiple frictional contacts. In *The Fifth International Symposium on Robotics Research*, pages 293–300, Cambridge, MA, USA, 1990. MIT Press.

[Bro89] Randy C. Brost. Computing metric and topological properties of configuration-obstacles. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 170–176, 1989.

[BS08] Dmitry Berenson and S. S. Srinivasa. Grasp synthesis in cluttered environments for dexterous hands. In *IEEE-RAS International Conference on Humanoid Robots*, pages 189–196, December 2008.

[BSD04] Erin Boivin, Inna Sharf, and Michel Doyon. Optimum grasp of planar and revolute objects with gripper geometry constraints. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 1, pages 326–332, April 2004.

[CB93a] I-Ming Chen and J. W. Burdick. A qualitative test for n-finger force-closure grasps on planar objects with applications to manipulation and finger gaits. In *Proceedings of IEEE International Conference on Robotics and Automation, 1993*, pages 814–820 vol.2, May 1993.

[CB93b] I-Ming Chen and Joel W. Burdick. Finding antipodal point grasps on irregularly shaped objects. *IEEE Transactions on Robotics and Automation*, 9(4):507–512, August 1993.

[CFMdP03] Eris Chinellato, Robert B. Fisher, Antonio Morales, and Angel Pascual del Pobil. Ranking planar grasp configurations for a three-finger hand. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 1, pages 1133–1138, September 2003.

[CLH+05] Howie Choset, Kevin M. Lynch, Seth A. Hutchinson, George A. Kantor, W. Burgard, Lydia E. Kavraki, and Sebastian Thrun. *Principles of robot motion: Theory, Algorithms and Implementations.* MIT Press, June 2005.

[CM97] Belén Curto and Vidal Moreno. Mathematical formalism for the fast evaluaton of the configuration space. In *Porceeding of the 1997 IEEE International Symposium On Comp. Intelligence in Robotics and Automation*, pages 194–199, 1997.

[CMB02] Belén Curto, Vidal Moreno, and Francisco Javier Blanco. A general method for c-space evaluation and its application to articulated robots. *IEEE Transactions on Robotics and Automation*, 18(1):24–31, February 2002.

[CMFdP05] Eris Chinellato, Antonio Morales, Robert B. Fisher, and Angel Pascual del Pobil. Visual quality measures for characterizing planar robot grasps. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 35(1):30–41, February 2005.

[Cut84] Mark R. Cutkosky. Mechanical properties for the grasp of a robotic hand. Technical Report CMU-RI-TR-84-24, Robotics Institute, Carnegie Mellon University, September 1984.

[CvdS05] Jae-Sook Cheong and A. Frank van der Stappen. Output-sensitive computation of all form-closure grasps of a semi-algebraic set. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 772–778, April 2005.

[CVM98] Belén Curto, Pastora Vega, and Vidal Moreno. Fast procedure to obstacle representation in the configuration space for mobile robots. In *Proceeding of Intelligent Components for Vehicles - ICV'98*, pages 437–442, 1998.

[CW86] Mark R. Cutkosky and Paul K. Wright. Modeling manufacturing grips and correlations with the design of robotic hands. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, pages 1533–1539, April 1986.

[DLW00] Dan Ding, Yun-Hui Liu, and Shuguo Wang. Computing 3-d optimal form-closure grasps. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 4, pages 3573–3578, April 2000.

[DLW01] Dan Ding, Yun-Hui Lee, and Shuguo Wang. Computation of 3-d form-closure grasps. *IEEE Transactions on Robotics and Automation*, 17(4):515–522, August 2001.

[Dub57] Lester E. Dubins. On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal position and tangents. *American Journal of Mathematics*, 79(3):497–516, July 1957.

[FC92] Carlo Ferrari and John F. Canny. Planning optimal grasps. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, pages 2290–2295, May 1992.

[FP91] Bernard Faverjon and Jean Ponce. On computing two-finger force-closure grasps of curved 2d objects. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 1, pages 424–429, April 1991.

[FP05] Cesar Fernández Peris. *Aprendizaje Mediante Árboles de Decisión para la Síntesis de Agarres Robóticos*. PhD thesis, Universidad Miguel Hernández, January 2005.

[FW91] Steven Fortune and Gordon Wilfong. Planning constrained motion. *Annals of Mathematics and Artificial Intelligence*, 3:21–82, 1991.

[GRS83] L. Guibas, L. Ramshaw, and J. Stolfi. A kinetic framework for computational geometry. In *Proceeding of the IEEE Symposium on Foundations of Computer Science*, pages 100–111, 1983.

[HA77] H. Hanafusa and H. Asada. A robotic hand with elastic fingers and its application to assembly processes. In *Proceeding of FAC Symp. on Information Control Problems in Production Engineering*, pages 127–138, October 1977. Reescrito en Brady, M., Hollerbach, Johnson T.L., Lozano-Perez, Tomás and

Mason, Matthew T. Robot Motion: Planning and Control. Cambridge, MA, MIT Press, pp. 322-335, 1982.

[HeKT99] R. D. Hester, Müjdat Çetin, Chetan Kapoor, and Delbert Tesar. A criteria-based approach to grasp synthesis. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pages 1255–1260, May 1999.

[HRSF99] A. Hauck, J. Ruttinger, M. Sorg, and G. Farber. Visual determination of 3d grasping points on unknown objects with a binocular camera system. In *Intelligent Robots and Systems, 1999. IROS '99. Proceedings. 1999 IEEE/RSJ International Conference on*, volume 1, pages 272–278 vol.1, 1999.

[HSSR05] Robert Haschke, Jochen J. Steil, I. Steuwer, and Helge Ritter. Task-oriented quality measures for dextrous grasping. In *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 689–694, June 2005.

[Ibe87] Thea Iberall. The nature of human prehension: Three dextrous hands in one. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 4, pages 396–401, March 1987.

[Ibe97] Thea Iberall. Human prehension and dexterous robot hands. *International Journal Robotics Research*, 16(3):285–299, 1997.

[Jia95] Yan-Bin Jia. On computing optimal planar grasps. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. 'Human Robot Interaction and Cooperative Robots'*, volume 3, pages 427–434, August 1995.

[Jia02] Yan-Bin Jia. Curvature-based computation of antipodal grasps. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pages 1571–1577, May 2002.

[Jia04] Yan-Bin Jia. Computation on parametric curves with an application in grasping. *The International Journal of Robotics Research*, 23(7-8):827–857, 2004.

[Kav94] Lydia E. Kavraki. *Random networks in configuration space for fast path planning.* PhD thesis, Stanford University, 1994.

[Kav95] Lydia E. Kavraki. Computation of configuration-space obstacles using the fast fourier transform. *IEEE Transaction on Robotics and Automation*, 11(3):408–413, 1995.

[KC92] Imin Kao and Mark R. Cutkosky. Quasistatic manipulation with compliance and sliding. *International Journal of Robotic Research*, 11(1):20–40, 1992.

[Kim04] Byoung-Ho Kim. Non-dimensionalized performance indices-based optimal grasping for multi-fingered hands. *Mechatronics*, 14(3):255–280, April 2004.

[KMY92] David Kirkpatrick, Bhubaneswar Mishra, and Chee-Keng Yap. Quantitative steinitzś theorems with applications to multifingered grasping. *Discrete & Computational Geometry*, 7(3):295–318, 1992.

[KOYS01] Byoung-Ho Kim, Sang-Rok Oh, Byung-Ju Yi, and Il Hong Suh. Optimal grasping based on non-dimensionalized performance indices. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 949–956, 2001.

[LA88] Jean-Paul Laumond and Rachid Alami. A new geometrical approach to manipulation task planning: the case of a circular robot amidst polygonal obstacles and a movable circular object. Technical Report 88314, LAAS/CNRS, October 1988.

[Lat91a] Jean-Claude Latombe. A fast path planner for a car-like indoor mobile robot. In *Proceedings of the 9th Nat. Conference on Artificial Intelligence*, pages 659–665, 1991.

[Lat91b] Jean-Claude Latombe. *Robot motion planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[Lau98] Jean-Paul Laumond, editor. *Robot motion planning and control*. Lecture Notes in Control and Information Sciences, 1998.

[LaV06] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.

[LFP07] Ying Li, Jiaxin L. Fu, and Nancy S. Pollard. Data-driven grasp synthesis using shape matching and task-based pruning. *IEEE Transactions on Visualization and Computer Graphics*, 13(4):732–747, July 2007.

[Liu98] Yun-Hui Liu. Computing n-finger force-closure grasps on polygonal objects. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, pages 2734–2739, May 1998.

[LJTM94] Jean-Paul Laumond, Paul E. Jacobs, Michel Taix, and Richard M. Murray. A motion planner for a nonholonomic mobile robot. *IEEE Transactions on Robotics and Automation*, 10(5):577–593, October 1994.

[LLC03] Jia-Wei Li, Hong Liu, and He-Gao Cai. On computing three-finger force-closure grasps of 2-d and 3-d objects. *IEEE Transactions on Robotics and Automation*, 19(1):155–161, February 2003.

[LLD04] Yun-Hui Liu, Miu-Ling Lam, and D. Ding. A complete and efficient algorithm for searching 3-d form-closure grasps in the discrete domain. *IEEE Transactions on Robotics*, 20(5):805–816, 2004.

[LP83] Tomás Lozano-Pérez. Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, C-32(Iss.2):108–120, February 1983.

[LP87] Tomás Lozano-Pérez. A simple motion-planning algorithm for general robot manipulators. *IEEE Journal of Robotics and Automation*, 3(3):224–238, 1987.

[LP90] Tom Lozano-Pérez. *Foreword: Mobile Robots and Robotics*. Springer-Verlag, cox, i. j. and wilfong, gordon edition, 1990.

[LPJM+87] Tomás Lozano-Pérez, Joseph L. Jones, Emmanuel Mazer, Patrick A. O'Donnell, W. Eric L. Grimson, Pierre Tournassoud, and Alain Lanusse. Handey: A robot system that recognizes, plans, and manipulates. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on, Vol.4, Iss.*, volume 4, pages 843–849, March 1987.

[LS87a] Zexiang Li and S. Shankar Sastry. Task oriented optimal grasping by multifingered robot hands. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 4, pages 389–394, March 1987.

[LS87b] Vladimir J. Lumelsky and Alexander A. Stepanov. Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *ALGORITHMICA*, pages 403–430, 1987.

[LS90] Vladimir J. Lumelsky and T. Skewis. Incorporating range sensing in the robot navigation function. *Systems, Man and Cybernetics, IEEE Transactions on*, 20(5):1058–1069, 1990.

[LW98] Yun-Hui Liu and Mei Wang. Qualitative test and force optimization of 3d frictional force-closure grasps using linear programming. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 4, pages 3335–3340, May 1998.

[LXWL04] Guanfeng Liu, Jijie Xu, Xin Wang, and Zexiang Li. On quality functions for grasp synthesis, fixture planning, and coordinated manipulation. *IEEE Transactions on Automation Science and Engineering*, 1(2):146–162, October 2004.

[Lyo85] Damian M. Lyons. A simple set of grasps for a dextrous hand. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pages 588–593, March 1985.

[LYT02] Yim Li, Yong Yu, and Showzow Tsujio. An analytical grasp planning on given object with multifingered hand. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 4, pages 3749–3754, 2002.

[MA99] Andrew T. Miller and Peter K. Allen. Examples of 3d grasp quality computations. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pages 1240–1246, 1999.

[MA00] Andrew T. Miller and Peter K. Allen. Graspit!: A versatile simulator for grasp analysis. In *Proceedings ASME International Mechanical Engineering Congress and Exposition*, November 2000.

[MA04] Andrew T. Miller and Peter K. Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics and Automation Magazine*, 11(4):110–122, December 2004.

[MB00] Alessia Marigo and Antonio Bicchi. Rolling bodies with regular surface: controllability theory and applications. *IEEE Transactions on Automatic Control*, 45(9):1586–1599, Sep 2000.

[MC94] Brian Mirtich and John F. Canny. Easily computable optimum grasps in 2-d and 3-d. In *Proceedings of IEEE International Conference on Robotics and Automation, 1994*, volume 1, pages 739–747, May 1994.

[MF93] A. A. Maciejewski and J. J. Fox. Path planning and the topology of configuration space. *IEEE Transaction on Robotics and Automation*, 9(4):444– 456, 1993.

[MiERSdP01] Antonio Morales i Escrig, G. Recatala, P.J. Sanz, and Angel Pascual del Pobil. Heuristic vision-based computation of planar antipodal grasps on unknown objects. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 1, pages 583–588, 2001.

[MiESP04] Antonio Morales i Escrig, Pedro J. Sanz, and Angel P. Del Pobil. How can i, robot, pick up that object with my hand? In *European Conf. on Artificial Intelligence*, 2004.

[Mis95] Bud Mishra. Grasp metrics: Optimality and complexity. In *Algorithmic Foundations of Robotics*, pages 137–165. A.K. Peters, 1995.

[MKCA03] Andrew T. Miller, S. Knoop, H.I. Christensen, and Peter K. Allen. Automatic grasp planning using shape primitives. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pages 1824–1829, September 2003.

[MLS94] Richard M. Murray, Zexiang Li, and S. Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1 edition, 1994.

[MNP90] Xanthippi Markenscoff, Luqun Ni, and Christos H. Papadimitriou. The geometry of grasping. *The International Journal of Robotics Research*, 9(1):61–74, 1990.

[Mon88] David J. Montana. The kinematics of contact and grasp. *International Journal of Robotic Research*, 7(3):17–32, 1988.

[Mon95] David J. Montana. The kinematics of multi-fingered manipulation. *IEEE Transactions on Robotics and Automation*, 11(4):491–503, Aug 1995.

[MP89] Xanthippi Markenscoff and Christos H. Papadimitriou. Optimum grip of a polygon. *International Journal of Robotics Research*, 8(2):17–29, 1989.

[MSS87] Bud Mishra, Jacob T. Schwartz, and Micha Sharir. On the existence and synthesis of multifinger positive grips. *Algorithmica*, 2(1):541–558, November 1987.

[Nap56] J. Napier. The prehensile movements of the human hand. *The Journal of bone and joint surgery*, 4(38B):902–913, November 1956.

[NB91] W. Newman and M. Branicky. Real-time configuration space transforms for obstacle avoidance. *The International Journal of Robotics Research*, 6:650–667, oct 1991.

[Ngu86a] Van-Duc Nguyen. Constructing force-closure grasps. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, pages 1368–1373, April 1986.

[Ngu86b] Van-Duc Nguyen. The synthesis of stable force-closure grasps. Technical Report AI-TR-905, MIT Artificial Intelligence Laboratory, 1986.

[Ngu87] Van-Duc Nguyen. Constructing force-closure grasps in 3d. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 4, pages 240–245, March 1987.

[Ngu88] Van-Duc Nguyen. Constructing force-closure grasps. *International Journal of Robotics Research*, 7(3):3–16, 1988.

[Ngu89] Van-Duc Nguyen. Constructing stable grasps. *International Journal of Robotic Research*, 8(1):26–37, 1989.

[Nil69] N.J. Nilson. A mobile automation: An application of artificial intelligence techniques. In *Proceedings of the Fith International Joint Conference on Artificial Intelligence*, pages 509–520, Washintong D.C., 1969.

[NK00] Christian L. Nielsen and Lydia E. Kavraki. A two level fuzzy prm for manipulation planning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1716–1721, 2000.

[OS95] Mark H. Overmars and Petr Svestka. A probabilistic learning approach to motion planning. In *WAFR: Proceedings of the Workshop on Algorithmic Foundations of Robotics*, pages 19–37, Natick, MA, USA, 1995. A. K. Peters, Ltd.

[OSC00] Allison M. Okamura, Niels Smaby, and Mark R. Cutkosky. An overview of dexterous manipulation. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 1, pages 255–262, 2000.

[PF91] Jean Ponce and Bernard Faverjon. On computing three-finger force-closure grasps of polygonal objects. In *Fifth International Conference on Advanced Robotics, 'Robots in Unstructured Environments'*, volume 2, pages 1018–1023, Jun 1991.

[PF95] Jean Ponce and Bernard Faverjon. On computing three-finger force-closure grasps of polygonal objects. *IEEE Transactions on Robotics and Automation*, 11(6):868–881, December 1995.

[PGS03] Ricardo Prado-Gardini and Raúl Suárez. Heuristic approach to construct 3-finger force-closure grasps for polyhedral objects. In *Proceedings of the 7th IFAC Symposium on Robot Control*, pages 387–392, september 2003.

[Pol96] Nancy S. Pollard. Synthesizing grasps from generalized prototypes. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, pages 2124–2130, April 1996.

[Pol04] Nancy S. Pollard. Closure and quality equivalence for efficient synthesis of grasps from examples. *The International Journal of Robotics Research*, 23(6):595–613, 2004.

[PS90] Young C. Park and Gregory P. Starr. Grasp synthesis of polygonal objects. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, pages 1574–1580, May 1990.

[PSBM93] Jean Ponce, Steve Sullivan, Jean Daniel Boissonnat, and Jean-Pierre Merlet. On characterizing and computing three- and four-finger force-closure grasps of polyhedral objects. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pages 821–827, May 1993.

[PSS+96] Jean Ponce, Steve Sullivan, Attawith Sudsang, Jean-daniel Boissonnat, and Jean-Pierre Merlet. On computing four-finger equilibrium and force-closure grasps of polyhedral objects. *International Journal of Robotics Research*, 16:11–35, 1996.

[RB96] Elon Rimon and Joel W. Burdick. On force and form closure for multiple finger grasps. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pages 1795–1800, April 1996.

[RS90] J. A. Reeds and L. A. Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 154(2):367–393, 1990.

[Sal82] J. Kenneth Salisbury. *Kinematic and Force Analysis of Articulated Hands*. PhD thesis, Stanford University, 1982.

[SCLS04] Thierry Simeón, Juan Cortés, Jean-Paul Laumond, and A. Sahbani. Manipulation planning with probabilistic roadmaps. *The International Journal of Robotics Research*, 23(7-8):729–746, 2004.

[SCR06] Raúl Suárez, Jordi Cornellà, and Máximo Roa. Grasp quality measures. Technical Report IOC-DT-P 2006-10, Universidad Politécnica de Cataluña. Instituto de Organización y Control de Sistemas Industriales, March 2006.

[Shi96] K. B. Shimoga. Robot grasp synthesis algorithms: a survey. *International Journal of Robotics Research*, 15(3):230–266, 1996.

[SLG+99] Gordon Smith, Eric Lee, Ken Goldberg, Karl-Friedrich Böhringer, and John Craig. Computing parallel-jaw grids. In *Proceedings of IEEE International Conference on Robotics and Automation*, May 1999.

[SP05] Attawith Sudsang and Thanathorn Phoka. Geometric reformulation of 3-fingered force-closure condition. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 2338–2343, April 2005.

[SPF92] Darrell Stam, Jean Ponce, and Bernard Faverjon. A system for planning and executing two-finger force-closure grasps of curved 2d objects. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 210–217, July 1992.

[SS83a] Jacob T. Schwartz and Micha Sharir. On the piano mover's problem i: the case of a two-dimensional rigid polygonal body moving amidst polygonal barriers. *Communications on Pure and Applied Mathematics*, 36:345–398, 1983.

[SS83b] Jacob T. Schwartz and Micha Sharir. On the piano mover's problem. ii. general techniques for computing topological properties of real algebraic manifolds. *Advances in Applied Mathematics*, 4:298–351, 1983.

[Sta91] S. A. Stansfield. Robotic Grasping of Unknown Objects: A Knowledge-based Approach. *The International Journal of Robotics Research*, 10(4):314–326, 1991.

[SVR03] Raul Suárez, Israel Vázquez, and Jose M. Ramírez. Planning four grasping points from images of planar objects. In *Proceedings of the IEEE International Symposium on Assembly and Task Planning*, pages 169–174, July 2003.

[Tei96] M. Teichmann. A grasp metric invariant under rigid motions. In *Proceedings of IEEE International Conference on Robotics and Automation, 1996*, volume 3, pages 2143–2148, April 1996.

[The02] R. Therón. *Cálculo Paralelo del Espacio de las Configuraciones para Robots Redundantes.* PhD thesis, Dpto. de Informática y Automática. Universidad de Salamanca, jun 2002.

[Tri92] Jeffrey C. Trinkle. On the stability and instantaneous velocity of grasped frictionless objects. *IEEE Transactions on Robotics and Automation*, 8(5):560–572, October 1992.

[vdSWO00] A. Frank van der Stappen, Chantal Wentink, and Mark H. Overmars. Computing immobilizing grasps of polygonal parts. *The International Journal of Robotics Research*, 19(5):467–479, 2000.

[WF95] David Wren and Robert B. Fisher. Dextrous hand grasping strategies using preshapes and digit trajectories. In *IEEE International Conference on Systems, Man and Cybernetics, Intelligent Systems for the 21st Century*, volume 1, pages 910–915, October 1995.

[Wil88] Gordon Wilfong. Motion planning in the presence of movable obstacles. In *Proceedings of the Fourth Annual Symposium on Computational Geometry*, pages 279–288, New York, NY, USA, 1988. ACM Press.

[ZDW03] Xiangyang Zhu, Han Ding, and Jun Wang. Grasp analysis and synthesis based on a new quantitative measure. *IEEE Transactions on Robotics and Automation*, 19(6):942–953, 2003.

[ZQ05] Yu Zheng and Wen-Han Qian. Simplification of the ray-shooting based algorithm for 3-d force-closure test. *IEEE Transactions on Robotics*, 21(3):470–473, June 2005.

[ZW03] Xiangyang Zhu and Jun Wang. Synthesis of force-closure grasps on 3-d objects based on the q distance. *IEEE Transactions on Robotics and Automation*, 19(4):669–679, August 2003.