



UNIVERSIDAD CARLOS III DE MADRID

MÁSTER UNIVERSITARIO DE ROBÓTICA Y AUTOMATIZACIÓN  
ROBOTS MÓVILES

??

*Autores:*

David Estévez Fernández  
Enrique Fernández Rodicio  
Javier Isabel Hernández

*Profesores:*

Ramón Barber Castaño

31 de mayo de 2015

# Índice general

<b>1. Introducción</b>	<b>3</b>
<b>2. Estado del arte</b>	<b>4</b>
2.1. Navegacion de robots autonomos . . . . .	4
2.2. Impresion 3D aplicada a la robotica . . . . .	6
<b>3. Desarrollo del proyecto</b>	<b>8</b>
3.1. Introduccion al proyecto Beerbot . . . . .	8
3.1.1. Objetivos . . . . .	8
3.1.2. Funcionamiento del sistema . . . . .	9
3.1.3. Escenario a utilizar . . . . .	9
3.2. Hardware . . . . .	10
3.3. Software . . . . .	10
3.3.1. Calibración del sistema de visión . . . . .	10
3.3.2. Procesamiento del entorno . . . . .	11
3.3.3. Planificación de trayectorias . . . . .	11
3.3.4. Control del robot y seguimiento de la trayectoria . . . . .	14
<b>4. Resultados obtenidos</b>	<b>15</b>
<b>5. Conclusion</b>	<b>16</b>

# Capítulo 1

## Introducción

El objetivo de este trabajo es aplicar los conocimientos adquiridos tanto en esta asignatura como a lo largo del Máster en el desarrollo de un caso práctico de estudio. Para ello, en esta memoria se presenta el proyecto Beerbot (Beer Extracting Environment Robot), un sistema compuesto por un robot móvil y una cámara aérea situada sobre el escenario. La cámara se encargará de tomar imágenes del entorno, estas se procesarán para obtener las características deseadas y posteriormente un algoritmo de planificación de trayectorias generará un camino que permita al robot cumplir su misión evitando obstáculos y siguiendo la ruta más directa. La tarea programada es recoger una lata en algún punto del entorno, y devolverla al punto inicial.

Este trabajo se divide en los siguientes apartados. Después de esta breve introducción, se llevará a cabo un breve estudio acerca de la navegación autónoma para robots móviles acerca de la impresión 3D, las diferentes tecnologías existentes y del uso de técnicas de impresión 3D en la fabricación de estos robots.

En la siguiente sección se pasará a hablar del proyecto Beerbot en sí. Se dividirá en tres subapartados. En el primero se presentará el proyecto y los objetivos a conseguir, además de mostrar el entorno de pruebas empleado. El siguiente apartado tratará sobre el hardware empleado, es decir, se comentará el diseño y funcionamiento del robot y el modelo de cámara con el que se va a trabajar. Por último, el último subapartado tratará acerca del software desarrollado, que se puede clasificar en tres etapas. La primera de ellas consiste en el procesamiento de las imágenes tomadas por la cámara, de las que se extraerán los obstáculos presentes en el entorno, la posición y orientación del robot y la posición de la lata (al ser cilíndrica, la orientación de la lata es irrelevante). La segunda etapa es la planificación de la trayectoria. Se usará un planificador probabilístico para muestrear el espacio libre y un método de resolución de grafos para encontrar el camino. Por último, la tercera etapa consiste en el mecanismo de control del robot, que transformará los datos extraídos del planificador en comandos de velocidad para las ruedas, y que se encargará también de asegurar que el robot no se desviará de la ruta seleccionada.

Por último, se mostrarán imágenes del sistema en funcionamiento y se comentarán los resultados obtenidos, además de mostrar algunas posibles mejoras que se han considerado pero no se han podido llegar a implementar, antes de cerrar este trabajo con una conclusión final.

# Capítulo 2

## Estado del arte

En esta sección se realizará un breve estudio acerca de los tres conceptos principales que conforman este trabajo. Se comenzará hablando sobre la navegación autónoma en el campo de la robótica, mostrando las diferentes partes que la conforman para a continuación entrar en el campo de la impresión 3D, centrándose en la tecnología de ámbito doméstico y su aplicación a la fabricación de robots.

### 2.1. Navegación de robots autónomos

Un robot móvil es aquella máquina automática que puede moverse en un determinado entorno. En función de la forma en la que se mueva se pueden dividir en diferentes categorías: robots rodantes (emplean ruedas para moverse), andantes (emplean dos o más extremidades para desplazarse), reptantes (se arrastran por el suelo), nadadores, voladores. . .

En el campo de la robótica móvil, se conoce como navegación al guiado de un robot que parte de un punto inicial, a través de un entorno en el que existen ciertos obstáculos a evitar, para alcanzar el punto final deseado. Para que se pueda cumplir con un cierto éxito esta tarea, es necesario poder resolver cuatro grandes problemas: Generar un mapa, localizar al robot en él, generar una trayectoria que una los puntos inicial y final y, por último, ser capaz de seguir esa trayectoria sin desviarse.

En primer lugar, se necesita proporcionar al robot un mapa del entorno que posea una precisión adecuada. Este puede haber sido realizado de antemano, o generado por el propio robot, empleando para ello los diversos sensores con los que cuenta, al mismo tiempo que avanza. Esta técnica es conocida como Simultaneous Localization And Mapping, o SLAM, que busca que el robot pueda ser capaz de generar el mapa de su entorno y, al mismo tiempo, pueda localizarse con éxito dentro de ese mapa. De entre todas las técnicas empleadas para su solución, las más efectivas son las basadas en técnicas probabilísticas, que parten del teorema de Bayes, para poder tener en cuenta las distintas fuentes de incertidumbre que van apareciendo a lo largo del problema. Los principales algoritmos en este campo son el Filtro Extendido de Kalman, los mapas de ocupación de celdillas o la solución factorizada del filtro de Bayes.

A continuación, el robot debe ser capaz de localizarse dentro del mapa del entorno. Si para generar dicho mapa se empleó la técnica de SLAM, el punto anterior y este se resuelven de forma simultánea. Si el mapa fue generado de forma externa, debe emplearse un sistema que permita al robot conocer su posición. Por lo general, existen tres familias de algoritmos para esta tarea, aquellos basados en el concepto de probabilidad bayesiana, aquellos que plantean la localización del robot como la solución a un problema de optimización, y aquellos que mezclan conceptos de los dos tipos anteriores, conocidos como métodos híbridos.

El teorema de Bayes establece que la probabilidad de que se dé un suceso aleatorio A conociendo que otro suceso B ya ha ocurrido depende de la distribución de probabilidad condicional de B si A ha ocurrido y de la distribución de probabilidad marginal de A. Aplicado a la localización de robots móviles, se calcula la probabilidad a posteriori asociada a una posible posición del robot dentro del espacio de soluciones del problema con los datos extraídos de la odometría y de los sensores y, a continuación, partiendo del paso anterior, se estima

la posición futura del robot. Dentro de esta familia, destacan los métodos de localización de Markov y el filtro de Kalman, aunque este último no permite obtener la posición del robot, simplemente sirve para mantener localizado al robot durante el movimiento.

En cuanto a los métodos de optimización, trabajan con una función de coste que asigna a cada posible posición del robot un valor de coste. Una vez se tiene definida dicha función, se usa uno de los métodos de resolución de problemas de optimización, como pueden ser los algoritmos evolutivos, para tratar de encontrar la solución cuyo valor de coste sea mínimo. En bastantes casos, el coste es la medida que representa la diferencia que hay entre las medidas de los sensores del robot y de las medidas estimadas que se obtendrían si el robot se encontrase en esa posición.

Una vez hecho esto, el robot debe ser capaz de elegir una ruta libre de obstáculos para poder desplazarse desde el punto inicial al punto final. Para resolver este problema, se puede tener en cuenta las características de movimiento del robot a emplear o se puede buscar una solución basada simplemente en el mapa del entorno y los puntos inicial y final. La planificación se puede plantear de dos maneras, de forma explícita, construyendo el espacio libre, aunque esté puede ser excesivamente costoso computacionalmente hablando, o muestreando el espacio libre, sin llegar a definirlo por completo, lo que es suficiente para encontrar un buen camino, usando un número suficiente de muestras.

Un planificador es completo si es capaz de encontrar un camino, en caso de que exista, en un tiempo finito. Los planificadores completos de resolución garantizan una solución si la resolución de la rejilla con la que se representa el espacio libre es lo suficientemente alta. Por último, los planificadores probabilísticos, como el método de Monte Carlo, ofrecen una mayor probabilidad de encontrar el camino cuantas mas iteraciones se realicen.

En general, hay cinco tipos de planificadores. Los planificadores basados en rejilla representan el espacio de configuraciones como una cuadrícula, donde cada punto de esta representa una posible configuración del robot. Para encontrar el espacio libre, se comprobará si el camino entre dos celdas contiguas está despejado, y en caso afirmativo, se añadirá al conjunto de puntos por los que puede pasar el robot. Por último, se usa un método de resolución de grafos para encontrar el camino. Este método no es muy recomendable para entornos muy grandes o no estáticos, ya que la rejilla crece de forma exponencial con el tamaño del escenario, y en caso de que algo cambie, hay que volver a calcular el espacio libre.

Los planificadores basados en intervalos son similares a los que se acaban de ver, solo que en este caso, el tamaño de las celdas no tiene por que ser uniforme. Estas celdas se dividen en dos subconjuntos,  $X^-$  y  $X^+$ , donde  $X^-$  es el espacio por el que el robot se puede mover sin problemas, mientras que  $X^+$  es el espacio máximo del cual no se puede salir. Al igual que antes, se usa un método de resolución de grafos para encontrar la trayectoria, que tiene que estar contenida en  $X^+$  y ser factible en  $X^-$ . Dada su similitud con los planificadores basados en rejilla, también comparte uno de sus problemas, y es que este método tampoco es muy recomendable en entornos muy grandes.

Los planificadores geométricos representan tanto los obstáculos como el robot como polígonos. Después, con grafos de visibilidad o descomposición de celdas, se calcula el camino para que el robot pueda pasar entre los obstáculos.

Los planificadores basados en campos de potencial representan el entorno como un campo de potencial, donde el objetivo definido para el robot genera una fuerza de atracción, mientras que los obstáculos generarán una fuerza de repulsión. Son algoritmos muy poco costosos desde el punto de vista computacional, pero por otra parte, pueden quedarse atrapados en mínimos locales, aunque se han desarrollado métodos dentro de esta familia que solucionan este problema, como es el caso del Fast Marching.

Por último, los planificadores basados en muestreo son bastante efectivos para entornos de muchas dimensiones, son fáciles de implementar y además, son planificadores completos, por lo que siempre encontrarán un camino, si este existe. Se distribuyen una serie de muestras por el entorno (existen diferentes aproximaciones para esto), asegurándose de que no caigan dentro de ningún obstáculo. Después, se unen aquellos puntos entre los que el robot puede moverse sin colisionar. Por último, se usa algún método como el  $A^*$  o el Dijkstra para encontrar el camino. Con estos planificadores, el hecho de no encontrar un camino, no significa que no exista, ya que lo mismo si se aumenta el número de muestras se podría obtener una trayectoria. Dentro de esta familia se encuentran los algoritmos PRM, o Probabilistic Road Map o los Rapidly-expanding Random Trees (RRT).

Por último, el robot debe ser capaz de seguir la ruta que planificó anteriormente. Esto supone resolver la forma de adecuar dicha ruta a los parámetros que caracterizan el movimiento del robot (en caso de no haberlo hecho a la hora de planificar la ruta), además de poder mantener un seguimiento de su localización a lo largo del desplazamiento. Además, debe seguir recibiendo información del entorno a través de los sensores para poder reaccionar adecuadamente a la aparición de un obstáculo imprevisto.

En este trabajo se ha optado por una alternativa poco ortodoxa en cuanto al método para extraer el mapa y la localización del robot, ya que esto se consigue mediante una cámara externa que toma imágenes del entorno y las procesa para extraer los datos deseados. Para la planificación, se ha empleado un algoritmo basado en el muestreo, en concreto el PRM, y el seguimiento de la trayectoria se realiza de forma directa, reorientándose respecto al siguiente punto del camino cada vez que se avanza un poco.

## 2.2. Impresión 3D aplicada a la robotica

La impresión 3D es una técnica de fabricación basado en adición de material, donde se modela el objeto a fabricar usando algún software de diseño CAD en 3D y se van añadiendo capas de material para obtener el resultado. Es una tecnología que comenzó a utilizarse a mediados de los 80 y que recientemente se ha popularizado gracias a kits de menor tamaño y coste que permiten aplicaciones domésticas. Las diferentes tecnologías empleadas en este campo pueden separarse según el modo de depositar las capas de material y según el tipo de material empleado. Atendiendo al primer criterio, se encuentran los siguientes tipos:

1. **Impresión por extrusión:** El material, por lo general filamentos termoplásticos o de metal, se hace pasar por un extrusor que aplica calor, el cual, al combinarse con la presión debida a la diferencia de sección que existe entre el filamento y la boquilla del extrusor, provoca el paso del material de sólido a líquido. El cabezal se va moviendo para dar forma a la capa y, al depositarse, el material se enfría y vuelve a ser sólido. Un ejemplo se puede ver en la figura 2.2

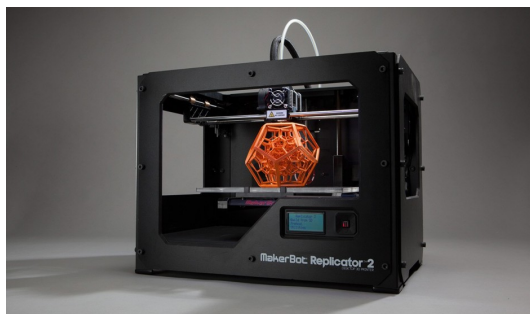


Figura 2.1: Impresora 3D por extrusión

2. **Impresión por fusión de material granulado:** Al igual que en el caso anterior, el objeto a obtener se fabrica también a base de ir añadiendo capas, solo que en este caso lo que se hace es depositar sobre la base de trabajo un material fusible y fundir aquellas partes que interesan. Una vez formada la capa, esta baja y se extiende por encima una nueva capa de material, y así sucesivamente hasta que se termine la pieza. Dependiendo del método empleado para fundir el material, se puede encontrar la sinterización selectiva por láser, que calienta el fundente hasta que las partículas se agrupan debido al proceso de sinterización, la fusión selectiva por láser, que funde completamente el material para formar la capa, y la fusión por rayos de electrones, que se usa para la fabricación de piezas metálicas, fundiendo las partículas de metal usando un rayo de electrones en el vacío. En la figura 2.2 se ve una impresora de este tipo



Figura 2.2: Impresora 3D por fusión de material

3. **Impresión por laminación:** Técnica consistente en la fabricación de piezas 3D en papel, cortando la forma de la capa en cada hoja y posteriormente uniéndolas todas, aplicando un adhesivo y presión.
4. **Impresión por fotopolimerización:** Se fabrica la pieza sometiendo a un polímero fotosensible a una emisión controlada de luz que provoca que se solidifique la capa. Una vez hecho esto, la base de impresión descende, y se pasa a trabajar sobre la siguiente capa, continuando con este proceso hasta que se haya terminado el objeto. Una impresora de este tipo se puede ver en la figura 2.3. Para imprimir piezas con una gran precisión (para aplicaciones de microfabricación) se puede usar la técnica de fotopolimerización por absorción de fotones. Se usa un láser para repasar la pieza sobre un bloque de resina, la cual se endurece solo en aquellas zonas donde el láser ha hecho contacto.

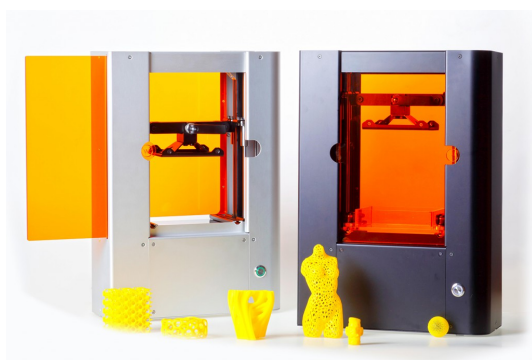


Figura 2.3: Impresora 3D por fotopolimerización

Aunque esta tecnología lleva años en uso, desde hace unos años se ha producido una gran expansión con la aparición de sistemas de impresión 3D que pueden ser contruídos de forma casera a un bajo coste. Esto ha dado un empujón de forma indirecta a la robótica, ya que permite diseñar y fabricar prototipos de forma rápida y cómoda, acercando este campo a todos los públicos. Usando un software de diseño asistido por ordenador, se generan unos planos en 3D de las piezas necesarias para el montaje, se imprimen y, con un controlador adecuado (hoy en día existen opciones de uso bastante general, como puede ser el caso de Aruino), servomotores y sensores comunes, se obtiene un robot simple sobre el que se puede seguir trabajando y desarrollando proyectos.

## Capítulo 3

# Desarrollo del proyecto

### 3.1. Introduccion al proyecto Beerbot

En esta sección se presenta la parte central del trabajo, el proyecto Beerbot, acrónimo de Beer Extracting Environment Robot. Se trata de un sistema formado por un robot móvil con una pinza y una cámara situada sobre el escenario. La cámara se conectará con un ordenador, que será el encargado de ejecutar el algoritmo desarrollado, desde el procesamiento de la imagen a la planificación de la trayectoria y el control de alto nivel del robot. Este ordenador se comunicará con el microcontrolador del robot a través de un módulo bluetooth, enviándole la velocidad de giro de cada rueda.

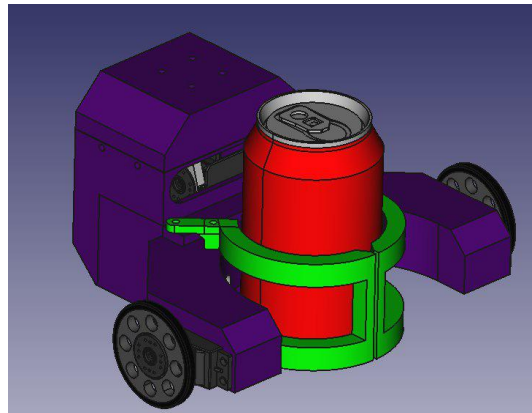


Figura 3.1: Modelo 3D del robot

#### 3.1.1. Objetivos

El objetivo principal de este proyecto es desarrollar un robot que sea capaz de navegar por un entorno con obstáculos estáticos para localizar una lata, recogerla con la pinza y llevarla al punto inicial. Para ello se trabajará con algoritmos de procesamiento de imágenes, algoritmos de planificación de trayectorias, e impresión 3D, para la fabricación del robot.

Además de este objetivo, con este trabajo también se pretende diseñar una aplicación real y física que permita poner en práctica los conceptos adquiridos no solo en la asignatura en sí, si no a lo largo del máster. Se han llevado a cabo otros proyectos acerca de planificación, pero en entornos simulados, por lo que la fabricación del robot y el uso de un entorno real son la parte clave, ya que con esto se podrá estudiar como las características del entorno afecta a los resultados obtenidos.

Respecto a la parte de procesamiento de imágenes, el hecho de tener un escenario con iluminación no controlada obligará a tener que explorar diferentes alternativas para combatir los cambios de iluminación y poder conseguir



los resultados esperados, especialmente en la etapa de segmentación de la imagen. Por otra parte, a diferencia de las simulaciones, el robot presentará errores de posición que será necesario corregir añadiendo un mecanismo de seguimiento de la trayectoria, para asegurarse que se llegue al punto al que realmente se está tratando de llegar.

### 3.1.2. Funcionamiento del sistema

El sistema funciona de la siguiente manera. El robot se sitúa en el escenario elegido, en un punto cualquiera. La cámara aérea captura una imagen del escenario, que se procesa para extraer de ella los obstáculos presentes en el entorno, la posición y orientación del robot y la posición del objetivo, es decir, la lata. Todos estos datos se envían al algoritmo de planificación, que muestrea el espacio libre por el que puede pasar el robot y calcula la ruta óptima por la que se pueda alcanzar el objetivo sin colisionar con los obstáculos. Conociendo los puntos por los que pasa la trayectoria, se va controlando la posición del robot en cada instante y se convierte la diferencia entre dicha posición y el siguiente punto del camino a comandos que le indiquen al robot la velocidad que debe llevar cada rueda. Una vez se alcanza una posición cercana a la lata, se abre la pinza, se coge la lata y el robot hace el mismo camino en la otra dirección (se supone que al ser un entorno estático, la trayectoria óptima debería seguir siendo la misma, además de simplificar el proceso, al ahorrarse un ciclo de planificación) hasta llegar al punto del que había partido, donde dejará la lata. En la siguiente sección de la memoria se comentarán más en detalle las diferentes partes del sistema.

### 3.1.3. Escenario a utilizar

Se va a trabajar sobre una pista cuadrada desarrollada para el concurso de robótica humanoide CEABOT. Tiene unas medidas de y un suelo de color verde, que facilita las tareas de segmentación. Para los obstáculos, se han utilizado objetos diversos que se han encontrado en la zona de trabajo, ya que era lo que se tenía a mano. Además con esto se consigue otra cosa: mostrar que no es necesario que los obstáculos tengan un color, tamaño o forma determinada, ya que se pueden detectar como tal sin problema. En cuanto a la lata, tendrá un tamaño estándar, pero no es necesario que tenga un color determinado, lo que permite que el sistema no esté limitado a un único tipo de objetivo. En cuanto a la cámara, se han utilizado dos listones rígidos unidos por un cable como estructura para soportarla. En la figura 3.2 se puede observar el escenario durante la etapa de montaje de la estructura que soporta la cámara.



Figura 3.2: Escenario empleado

Esta solución presenta varios problemas, siendo uno bastante importante las vibraciones que se generan al temblar el cable. Esto se podría solucionar fabricando una estructura rígida que soporte mejor la cámara. Otro problema por utilizar este escenario, quizás el más grave de todos, es que está situado en una zona muy amplia que, a pesar de ser interior, recibe una gran cantidad de luz natural proveniente de grandes ventanas situadas en el techo, que unido a la luz artificial presente, provoca que no se consiga una iluminación constante a lo largo del día, generándose gran cantidad de sombras en algunas zonas, o brillos en otras (esto también se debe al tipo de suelo que presenta el entorno), lo que dificulta en gran medida el correcto procesamiento de la imagen.

## 3.2. Hardware

## 3.3. Software

### 3.3.1. Calibración del sistema de visión

Para una correcta percepción del entorno es necesario realizar una calibración de la cámara para corregir las distorsiones producidas en la imagen por la lente y el sensor de la misma. La cámara posee dos tipos de distorsión: la radial, generada por la lente y que produce la curvatura de las líneas rectas de la imagen, y la tangencial, producida por un desalineamiento de la lente con el plano del sensor, de forma que ambos no se encuentran paralelos. La distorsión tangencial provoca que ciertas zonas de la imagen parezcan más cercanas de lo que realmente están.

Las fórmulas que modelan la distorsión radial son las siguientes:

$$x_{corrected} = x(1 + k_1r^2 + k_2r^4 + k_3r^6)$$

$$y_{corrected} = y(1 + k_1r^2 + k_2r^4 + k_3r^6)$$

La distorsión tangencial se modela con las siguientes fórmulas:

$$x_{corrected} = x + [2p_1xy + p_2(r^2 + 2x^2)]$$

$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2xy]$$

En total, la distorsión la modelaremos con 5 parámetros:  $(k_1 \ k_2 \ p_1 \ p_2 \ k_3)$

También es necesario obtener los parámetros intrínsecos, que son propios de la cámara, como la distancia focal y el centro óptico, así como los parámetros extrínsecos de la cámara, posición y orientación respecto de un sistema de coordenadas 3D.

La matriz de la cámara expresa estos parámetros intrínsecos en una sola matriz:

$$camera \ matrix = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

Donde  $(f_x, f_y)$  son las distancias focales en los ejes x e y, y las coordenadas del centro óptico son  $(c_x, c_y)$ .

Para realizar la calibración de la cámara es necesario un patrón de calibración como el de la figura ??, de dimensiones conocidas. Las bibliotecas OpenCV poseen funciones capaces de reconocer y detectar automáticamente los vértices de los cuadrados y, conociendo su posición dentro del patrón y las dimensiones del mismo en la realidad, calcular los parámetros de la cámara.

El proceso es el siguiente. En primer lugar se coloca la cámara fija y se realizan una serie de fotos del patrón de calibración en distintas posiciones y orientaciones. Se extraen de estas imágenes los puntos correspondientes a los vértices de los cuadrados del patrón (figura ??), que OpenCV usa para calcular tanto los parámetros de la distorsión como los parámetros intrínsecos de la cámara.

Una vez obtenidos estos parámetros, se almacenan en un archivo de forma que puedan ser cargados posteriormente. Usando estos archivos, es posible rectificar la imagen para eliminar la distorsión usando la función `undistort` de OpenCV, y posteriormente se recorta la región de interés de la imagen.

### 3.3.2. Procesamiento del entorno

El primer paso del algoritmo de planificación y control es la extracción de información del entorno a través de visión artificial. Al no poseer ningún tipo de sensor, toda la información que posee el controlador del robot se obtiene a través del sistema de visión.

El entorno es captado a través de una cámara cenital, fijada a una estructura de soporte (figura ??). El suelo del entorno, un escenario del concurso CEABOT, está pintado de un llamativo tono de verde. Para su localización, el robot lleva en su parte superior un marcador fiduciario formado por un rectángulo de dos colores, negro y azul (figura ??). Al poseer dos colores es posible extraer tanto la posición como la orientación del robot a través de la cámara.

Para la detección de obstáculos se utiliza una segmentación del color verde, aplicando una umbralización en el espacio de color HSV. La máscara resultante se procesa con una apertura usando un kernel rectangular de 5x5 píxeles, para reducir el ruido, y se lleva a cabo un etiquetado de objetos sobre la imagen binaria filtrada. Estos contornos, posteriormente, son también filtrados por tamaño y área, y simplificados.

A partir de los contornos de estos obstáculos se genera una imagen binaria que es utilizada como mapa para el algoritmo de planificación.

El marcador del robot se extrae mediante una segmentación del color azul, llevada a cabo también umbralizando en el espacio HSV y realizando el mismo filtrado que en el caso del color verde. Para una detección robusta del color negro del marcador, y para evitar reflejos producidos por una iluminación no controlada, el marcador completo se detecta buscando entre los obstáculos detectados el que posea un rectángulo azul en su interior, que es identificado y etiquetado como robot.

La posición del robot se obtiene calculando el centro del contorno del robot, y la orientación usando el vector que conecta el centro de dicho contorno con el centro del contorno azul, midiendo su orientación con respecto al eje X.

Por último, para la detección de la lata, se comprueban todos los obstáculos buscando aquellos que tengan forma circular. Esta forma circular se calcula teniendo en cuenta la proporción entre el área del contorno y el área del menor círculo que incluye el contorno, que es aproximadamente 1 en el caso de que el contorno sea totalmente circular. Para que la segmentación no etiquete obstáculos circulares como latas, se realiza también un filtrado de candidatos por tamaño, reduciendo así el número de posibles falsos positivos.

### 3.3.3. Planificación de trayectorias

La planificación de trayectorias toma como entrada toda la información obtenida a través del algoritmo de procesamiento de imágenes: el mapa del entorno, la posición inicial del robot y la posición del objetivo, la lata.

El algoritmo elegido para la planificación de la ruta es un Quasi-Randomized Probabilistic Roadmap (Q-PRM), una variante del PRM que utiliza una distribución cuasi-aleatoria de puntos para calcular la hoja de ruta.

El algoritmo PRM tiene dos fases principales. En la primera, se genera el PRM propiamente dicho mediante un muestreo del espacio libre a través de una distribución aleatoria de puntos, que se conectan siguiendo un criterio de

distancias, procurando siempre que , tanto los nodos generados como las aristas que los conectan se encuentren en el espacio libre (no exista colisión con obstáculos. En la segunda, se realizan peticiones de caminos al mapa, conectando un punto inicial y uno final al grafo y calculando el camino más corto mediante un algoritmo de búsqueda en grafos, tal como el Djisktra o el  $A^*$ .

La mejora introducida por la variante Q-PRM es el uso de una distribución de puntos cuasi-aleatoria en lugar de aleatoria, lo que permite elegir una distribución de puntos que rellene el espacio de forma más óptima, siguiendo unos criterios de discrepancia y dispersión.

La discrepancia para un conjunto  $P$  formado por  $N$  muestras de  $d$ -dimensiones en  $[0, 1]^d$  viene dada por la ecuación 3.1:

$$D_N(P) = \sup_j \left| \frac{A(J)}{N} - \mu(J) \right| \quad (3.1)$$

Donde  $J$  es cualquier subconjunto  $n$ -rectangular perteneciente a  $[0, 1]^d$ ,  $\mu(J)$  es su medida  $n$ -dimensional y  $A(J)$  es el número de puntos que pertenecen a la unión entre  $P$  y  $J$ . En cuanto a la dispersión, hace referencia a la máxima distancia a la que cada punto de un conjunto puede estar respecto al punto mas cercano perteneciente a la misma secuencia. Por lo general, los conjuntos de puntos que presentan una baja discrepancia también poseerán una baja dispersión.

Estas distribuciones de puntos tienen la ventaja de ocupar el espacio libre de forma más eficiente, por lo que con menos cantidad de puntos se puede abarcar un área mayor y, por tanto, más partes delicadas del mapa. Estas partes delicadas pueden ser, por ejemplo, estrechamientos del espacio libre entre obstáculos.

Para el muestreo de puntos cuasi-aleatorios se han usado dos distribuciones de puntos: el conjunto de Hammersley y el conjunto de Halton. Estas distribuciones se generan a partir de una semilla para un número arbitrario de dimensiones. Estas distribuciones se pueden calcular de la siguiente manera:

- **Conjunto de Hammersley:** Dados  $d - 1$  números primos distintos  $p_1, p_2, \dots, p_{d-1}$  el  $i$ -ésimo punto del conjunto es dado por la expresión:

$$\left( \frac{i}{N}, r_{p_1}(i), \dots, r_{p_{d-1}}(i) \right), \quad i = 0, 1, \dots, N - 1$$

- **Conjunto de Halton** Dados  $d$  números primos distintos  $p_1, p_2, \dots, p_d$  el  $i$ -ésimo punto del conjunto es dado por la siguiente expresión:

$$(r_{p_1}(i), r_{p_2}(i), \dots, r_{p_d}(i))$$

En ambos casos, la función  $r_p(i)$  se obtiene escribiendo los dígitos de la notación basada en  $p$  en orden inverso. Por ejemplo, para la expresión  $i = a_0 + a_1p + a_2p^2 + a_3p^3 + \dots$  donde  $a_j \in \{0, 1, \dots, p - 1\}$  la función  $r_p(i)$  sería:

$$r_p(i) = \frac{a_0}{p} + \frac{a_1}{p^2} + \frac{a_2}{p^3} + \frac{a_3}{p^4} + \dots$$

La figura 3.3 muestra una comparativa de los distintos métodos de generación de puntos de muestreo sobre el escenario del CEABOT usado como entorno de trabajo.

### Cambiar figura 3.3

La distribución elegida fue la de Hammersley (figura 3.3b), debido a una mayor dispersión, de forma que la conectividad de la hoja de ruta era mayor.

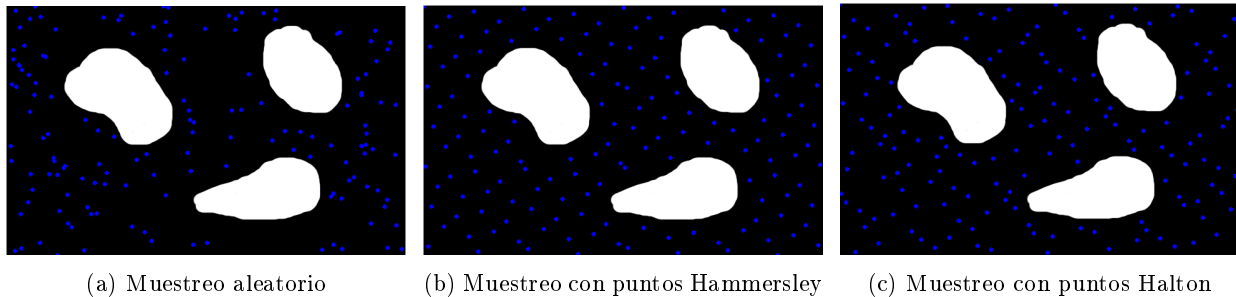


Figura 3.3: Comparativa de los distintos modos de muestreo del espacio libre

Una vez generados los puntos para el muestreo del espacio libre, se comprueba cada uno de los puntos comprobando colisiones con los obstáculos. Para ello se utiliza un detector de colisiones basado en convolución: para reducir el tiempo de cómputo de colisiones con el modelo del robot, se aplica primero la convolución del círculo que representa al robot sobre toda la imagen, de forma que los obstáculos se dilatan. De esta forma se puede usar la detección de colisión simple, mucho más rápida, sobre los obstáculos dilatados.

Tras eliminar los nodos que colisionan con obstáculos, se procede a la conexión de los distintos nodos para la generación del grafo de la hoja de ruta. El criterio seguido para conectar dos nodos es su vecindad. Se consideran vecinos dos nodos que estén a una distancia entre ellos menor que un valor umbral. Para cada pareja de nodos se calcula la distancia y se coloca en una matriz que representa la conectividad del grafo, y que es usada como entrada de un algoritmo de búsqueda en grafos como Dijkstra o  $A^*$ . Al ser la distancia del nodo A al nodo B la misma que la del nodo B al nodo A, sólo se calculan las distancias una vez.

Por último, y antes de añadir cada conexión a la hoja de ruta, se comprueba si esa arista del grafo es una trayectoria válida para el robot o colisiona con algún obstáculo, y sólo se añade en caso de que sea válida. Para la comprobación de la arista, se discretiza en diversos puntos, y se comprueba cada uno de ellos con alguno de los métodos presentados anteriormente.

Una vez se ha muestreado el espacio libre y se han conectado los nodos entre sí para generar un grafo, el siguiente paso es buscar el camino más corto. Para ello nuestra librería **PLATANO** se dispone de dos algoritmos diferentes, el  $A^*$  y el Dijkstra.

El algoritmo de Dijkstra ... es un método para encontrar el camino mas corto desde un nodo hasta todos los demás nodos de un grafo, aunque se suele usar para encontrar el camino más corto entre dos nodos, que para el algoritmo cuando se ha encontrado ese camino. En este algoritmo, el coste de cada nodo (también llamado prioridad) se calcula como la distancia entre éste y el nodo inicial, manteniendo una lista de nodos visitados y una de nodos por visitar. De esta forma se recorre el grafo viajando siempre por los nodos de menor distancia hasta que se encuentra un nodo anterior con menor distancia o se llega hasta el nodo objetivo. y revisar

El algoritmo  $A^*$ , una evolución del algoritmo de Dijkstra, busca devolver la trayectoria que vaya del nodo inicial al final y que presente un coste más bajo. Mientras va explorando un camino mantiene otras posibles soluciones para cambiar de trayectoria, en caso de que la que se está recorriendo deje de ser óptima. Este algoritmo garantiza que siempre se encontrará una solución, si existe alguna.

El coste que se emplea para comparar las distintas posibles trayectorias entre sí viene dado por la suma de dos funciones. La primera es la distancia recorrida entre el nodo inicial y el nodo para el cual se está calculando el coste, mientras que la segunda consiste en una estimación de la distancia que es necesario recorrer para llegar al punto de destino. Aunque esto último se puede representar de diversas maneras, lo mas común es calcular la distancia en línea recta desde el nodo actual hasta el final.

Una vez obtenida la trayectoria como una secuencia de puntos por los que el robot debe pasar hasta llegar a la meta, estos puntos son pasados al controlador del robot para que realice el movimiento hasta la meta.

#### 3.3.4. Control del robot y seguimiento de la trayectoria

## Capítulo 4

# Resultados obtenidos

## Capítulo 5

## Conclusion