MODELADO DE ENTORNO Y PLANIFICACIÓN DE TRAYECTORIA

Práctica nº3 de Robótica

22/12/2019

4.º Curso, grado de ingeniería de computadores

Álvarez Gil, Daniel

Krasimirov Ivanov, Todor

Resumen

En la práctica se recogen las ideas y métodos utilizados para llevar a cabo un análisis del entorno por el que se moverá un robot, el cual buscará obstáculos mediante sensores ultrasonido y hará un mapa de la planta posicionando todos los obstáculos en el lugar donde fueron detectados.

Para conseguir esto se utiliza la técnica de muestreo del entorno del algoritmo Vector Field Histogram. Se basa en recorrer el espacio e ir rellenando una parrilla o grid que representará un modelo de la planta con sus respectivos obstáculos.

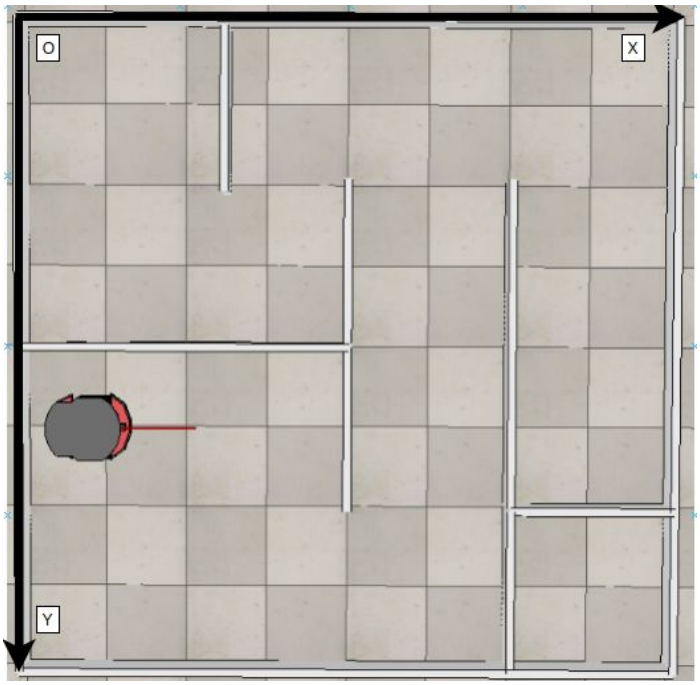
Posteriormente, se hará uso del grid previamente mencionado para establecer una trayectoria de un punto origen a un punto destino del entorno sin chocar con los objetos existentes en él. Para ello, se tratan los datos del mapa de la planta de forma que sea posible determinar un camino a través de los obstáculos para realizar la trayectoria con éxito.

Todo ello se visualizará en V-REP, un entorno de simulación virtual para robots.

Modelado del entorno

El modelo del entorno se construirá mediante un grid, por lo que se estudia previamente la escena para poder valorar cual sería una dimensión aceptable y así tener el mapeo realizado correctamente.

En primer lugar, se decide trasladar el origen de coordenadas a la esquina superior izquierda para tener un sistema de referencia con el que poder trabajar.



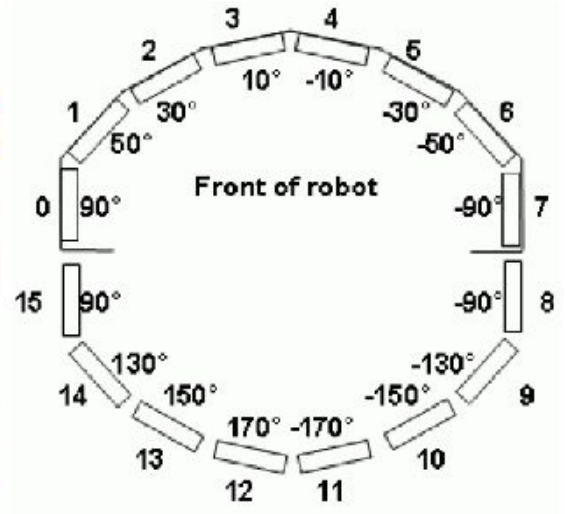
Para saber la dimensión del grid se emplea la siguiente lógica. Se conoce el tamaño de los catetos del cuadrado que conforman la escena, teniendo un valor de 4 unidades. Después se decide cual va a ser la resolución del grid, en este caso, se utiliza una resolución de 0,2 unidades con la que se discretizará la planta. Entonces, la dimensión de la parrilla o matriz se calculará del siguiente modo:

$$dimX = catetoX / resolución$$

$$dimY = catetoY / resolución$$

En este caso, ambas dimensiones tienen el mismo valor debido a que la escena que se está empleando es un cuadrado.

Para actualizar el grid según se vayan detectando obstáculos, hacemos uso de los 16 sensores, que están orientados de la forma que refleja la figura de la derecha.



Al utilizar los 16 sensores de los que dispone el robot, cuando uno de ellos detecta un obstáculo, se calculará qué celda de la parrilla es la que se ha de actualizar para obtener un mapa correcto. La celda a modificar va en función del sensor que detecta el obstáculo debido a que cada uno de ellos poseen una orientación distinta. Además, se ha de tener en cuenta la separación entre los sensores y el centro de gravedad (G) del robot para reducir los márgenes de error en la actualización del grid.

Dado que el robot no es circular, existe un desplazamiento entre el centro de gravedad del robot y el centro desde el que parten las orientaciones de los sensores. Desde la referencia relativa del robot, este offset es positivo para los sensores frontales y negativo para los traseros.

Como se pretende construir un mapa general del entorno, el eje de referencia será el mismo entorno, por lo que también hay que tener en cuenta la posición y orientación del propio robot respecto del eje de referencia absoluto.

Con todas las variables anteriores, las ecuaciones para el cálculo de la celda a modificar da como resultado:

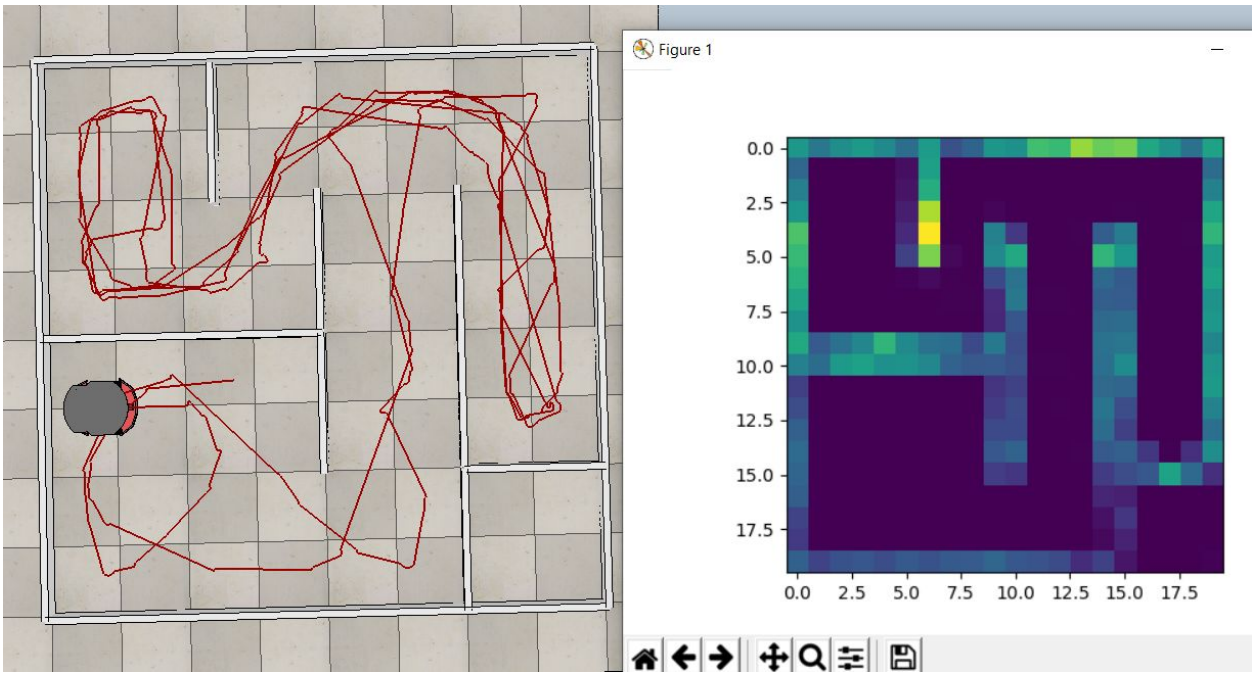
$$X(grid) = X + (ds + d) * cos(\Phi + \theta) + offset * cos(\Phi)$$

$$Y(grid) = Y + (ds + d) * sin(\Phi + \theta) + offset * sin(\Phi)$$

- ***X(grid), Y(grid):*** *Coordenadas a modificar del grid*
- ***X, Y:*** *Posición del robot en la escena*
- ***ds:*** *Distancia entre el sensor y la posición(centro) del robot*
- ***d:*** *Distancia detectada por el sensor*
- ***Φ:*** *Orientación del robot en la escena*
- ***θ:*** *Orientación del sensor respecto del robot*
- ***offset:*** *Separación entre el punto de referencia de los sensores y el del robot*

Las coordenadas obtenidas se traducen a filas y columnas para obtener la celda a actualizar. Esto se consigue, al igual que en la ecuación expuesta en la página anterior, con el cociente entero del valor de *X(grid)* e *Y(grid)* entre la resolución, obteniendo la fila y columna respectivamente.

La actualización se basa sencillamente en incrementar en 1 el valor de la celda detectada, de esta forma, cuando el robot esté explorando la planta con su debido algoritmo de evasión de obstáculos, irá creando un mapa con valores altos en lugares donde existan obstáculos y valores bajos (o nulos) en lugares espaciados, es decir, regiones por los que el robot puede transitar sin problemas.



Resultado obtenido del modelado del entorno

Planificación de trayectorias

En esta sección, se hará un tratamiento del mapa obtenido en el apartado anterior para establecer una trayectoria factible, libre de obstáculos, entre un punto A y un punto B de la escena que representa dicho mapa.

La planificación de trayectorias está dividida en dos partes:

- Construcción, basada en el grid, de los posibles caminos entre obstáculos a nivel genérico para realizar cualquier trayectoria en la escena que representa, lo cual es análogo a la creación de un grafo.
  Dado que el mapa de caminos es general, este paso solo se ha de realizar una vez por cada entorno distinto.

- Búsqueda de la trayectoria a realizar para llegar desde un punto origen a otro destino lo cual se servirá del modelo de caminos construido en la primera parte. Este paso es necesario cada vez que se desee cambiar el origen o el destino.

Construcción de caminos

La construcción de caminos se realiza mediante una máscara 2x2 que analiza el grid buscando puertas, lo cual implica que el espacio libre se ve estrechado o ensanchado por el comienzo o final de un obstáculo. Esto se traduce en la búsqueda de esquinas de obstáculos, es decir, cuándo se cierra o se abre un camino.

Cuando se detecten esos casos, se establecerá un vértice del grafo en la diagonal hacia la que esté apuntando la esquina, creando así un punto en el que habrá una puerta, un cruce o un cambio de dirección junto con sus coordenadas en el mapa.

Después, para determinar qué vértices están interconectados la idea se basará en ver, para todos los pares de vértices, si existe algún obstáculo entre ellos lo cual se puede hallar sumando los

coeficientes de obstáculo del área cuadrática que forman en el grid siendo dicha suma un valor por debajo de un umbral. Esta forma, permite establecer los caminos factibles entre los distintos puntos críticos del entorno.

El valor de las aristas entre punto y punto será la orientación absoluta(relativa al entorno) que se ha de disponer para ir del vértice *i* al vértice *j*, siendo la del vértice *j* al *i* la misma sumando 180º o π radianes.

Con esto, ya tendríamos un grafo generalista con los posibles caminos que existen entre obstáculos, lo cual permitirá el tránsito entre zonas en las que existan paredes y cambios de dirección. Esta solución, reduce mucho el espacio de exploración a la hora de definir una trayectoria concreta.

La solución desarrollada en programación es mejorable para incrementar la eficiencia del algoritmo, por ejemplo, evitando la creación de algunos vértices redundantes que ofrezcan los mismos caminos que otros ya existentes.

Búsqueda de una trayectoria

La trayectoria comienza por determinar un punto crítico accesible desde el punto origen de forma directa, es decir, sin necesidad de esquivar obstáculos. Análogamente con el punto destino.

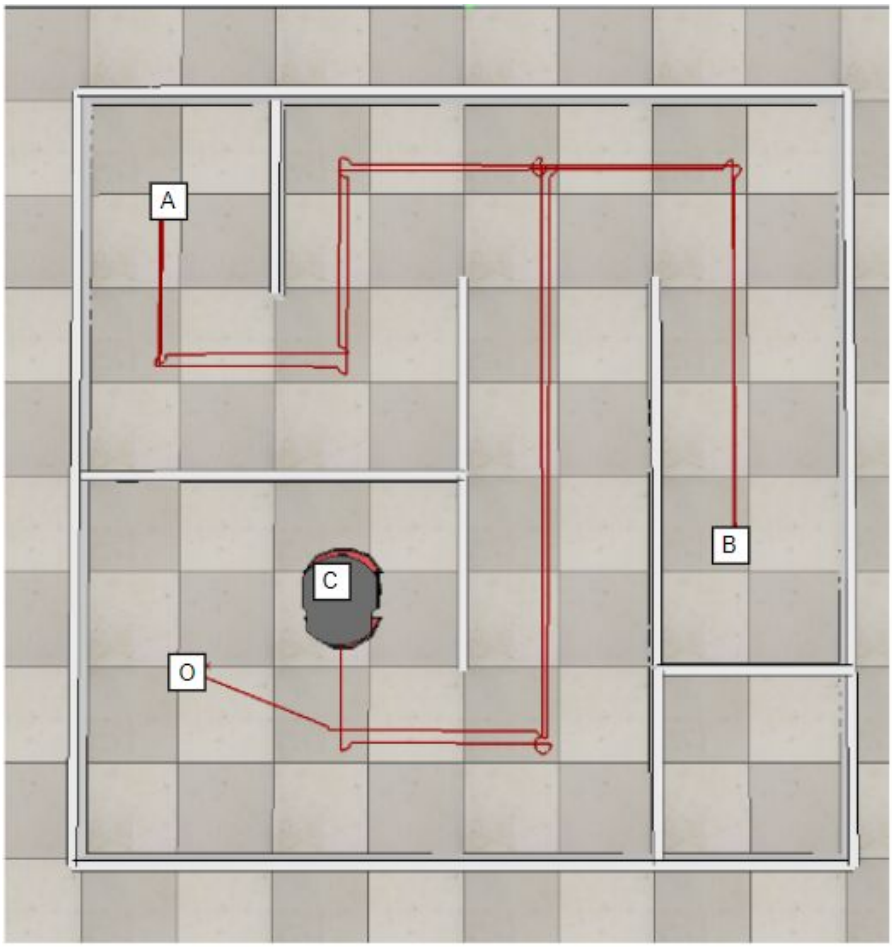
Este proceso es paralelo al de la creación de aristas del apartado anterior de construcción de caminos, pues sería como disponer de dos vértices más (origen y destino) en el mapa de caminos y determinar con cuál del resto de vértices existe arista: suma de coeficientes de obstáculo del área cuadrática que forman ambos vértices. El primer vértice encontrado que sea accesible desde el punto origen(posición actual del robot) será al que se dirija de forma directa y el primero encontrado para el destino será desde el cual se llegará al punto destino directamente. Estos vértices se denominan vértices destino y origen.

Una vez hecho esto, el siguiente paso es buscar en el grafo de caminos un conjunto de vértices que lleven desde el vértice origen hasta el vértice destino. Existen muchos algoritmos para la

búsqueda de caminos en grafos. Aquí, al ser el grafo de caminos tan pequeño, se ha optado por una búsqueda tradicional del camino más corto.

Al obtener la secuencia de vértices que se han de transitar para llegar desde el punto origen al punto destino, se obtiene la secuencia de coordenadas con sus respectivas orientaciones para ir de un vértice a otro.

Con esta información, el robot podrá realizar la trayectoria siguiendo punto a punto, con el ángulo adecuado, el camino calculado.



Resultado obtenido

Guía para el uso del programa

Se dispone de dos clientes:

- ***client-pr3.py***

  Modela el entorno. Es el cliente que realiza la creación y actualización del grid cuya dimensión se puede definir estableciendo la longitud del cuadrado de la escena y la resolución deseada (variables situadas al inicio del programa). Estos parámetros ya están definidos acorde a la escena que se aporta.

  Después de unos 3 minutos, se da por finalizado el muestreo de obstáculos, se almacena el grid en un fichero llamado *grid.txt* y acaba el cliente.

- ***trajectory.py***

  Es el cliente que contiene los procedimientos para la planificación de trayectorias. Se deberá definir la resolución del grid y, de forma explícita, las coordenadas iniciales del robot, puesto que inicialmente no se leen de forma adecuada. Se podrán definir una sucesión de puntos destino que el robot visitará por orden. Todo definido en las primeras líneas de código. Para un funcionamiento adecuado, dichos puntos no deberán estar demasiado cerca de un obstáculo ni salirse de las coordenadas que encierran la escena. En caso de definir un punto inalcanzable, el programa cesará puesto que no encuentra un camino.

Primero, ejecutar el primer cliente para que genere el fichero con el grid y, posteriormente, ejecutar el segundo cliente para que el robot realice la trayectoria.