Reporte Tarea 2

Baruc Samuel Cabrera García

27 de noviembre de 2023

Buscaremos implementar el siguiente algoritmo:

Algorithm 3 Tangent Bug Algorithm

Input: A point robot with a range sensor.

Output: A path to the q_{goal} or a conclusion no such path exists.

1: **while** True **do**
2: **repeat**
3: Continuously move toward the point $n \in \{T, O_i\}$ which minimizes $d(x, n) + d(n, q_{\text{goal}})$
4: **until**

- the goal is encountered **or**
- The direction that minimizes $d(x, n) + d(n, q_{\text{goal}})$ begins to increase $d(x, q_{\text{goal}})$, i.e., the robot detects a "local minimum" of $d(\cdot, q_{\text{goal}})$.

5: Chose a boundary following direction which continues in the same direction as the most recent motion-to-goal direction.
6: **repeat**
7: Continuously update d_{reach}, d_{followed}, and $\{O_i\}$.
8: Continuously moves toward $n \in \{O_i\}$ that is in the chosen boundary direction.
9: **until**

- The goal is reached.
- The robot completes a cycle around the obstacle in which case the goal cannot be achieved.
- $d_{\text{reach}} < d_{\text{followed}}$

10: **end while**

Figura 1:

Primero, recapitulemos el significado de cierta notación en tal algoritmo:

- T: Punto máximo del sensor en dirección a la meta. Se debe cumplir que $d(x, T) = R$ y no haya otro punto percibido por el sensor entre x y T.
- $\{O_i\}$: Conjunto de end points percibidos por el robot, es decir, puntos donde se detecta discontinuidad en la distancia percibida por el sensor.
- x: Posición actual del robot.
- $\mathcal{W}O_i$: El obstáculo de bloque es el obstáculo mas cercano a x que se percibe entre x y q_{goal}.
- Λ: Todos los puntos dentro de la línea de vista de x con rango R que no están en el obstáculo de bloque $\mathcal{W}O_i$, es decir,
$$\Lambda = \{y \in \partial \mathcal{W}O_i : x(\lambda) + y(1-\lambda) \in Q_{free}, \forall \lambda [0,1]\}.$$
- d_{reach}:
$$d_{reach} = \min_{c \in \Lambda} d(q_{goal}, c).$$

- $d_{followed}$: Distancia mas pequeña entre la frontera detectada y q_{goal}.

Con base en lo anterior, podemos interpretar sin problemas el pseudocódigo ya mostrado.

Primero, hay que dividir el pseudocódigo en dos casos, los cuales están representados en los "repeat" que están en las líneas 2 y 6, los cuales respectivamente representan los casos "Movimiento hacia goal"y "Seguimiento de Frontera". A continuación, explicaremos que realiza cada caso, y como lo implementamos en el código.

Movimiento hacia goal

El objetivo es siempre moverse hacia la dirección n, donde

$$n = \arg \min_{v \in T, O_i} d(x, n) + d(n, q_{goal}).$$

Es decir, siempre nos moveremos en dirección recta hasta encontrarnos con un obstáculo, en tal casó nos moveremos hacia n.

El conjunto $\{T, O_i\}$ se obtiene lanzando rayos de un conjunto finito de grados desde x al espacio, se registran las intersecciones con las aristas, y en caso de que las diferencias en las distancias respecto a x rebasen cierta tolerancia, se considerara como discontinuidad.

Y en caso de que haya vía libre entre x y q_{goal} según el sensor, se agrega el punto T.

Seguimos tal dinámica hasta que se cumpla una de las siguientes condiciones:

1. Encontremos a q_{goal}
2. El valor $d(x, n) + d(n, q_{goal})$ comienza a crecer.

Seguimiento de Frontera

En este caso, no podemos seguir avanzando hacia q_{goal} debido a que chocamos con un obstáculo, al cual denominaremos como $\mathcal{WO}$, y con tal obstáculo, necesitamos tener actualizado el siguiente conjunto según la posición de x:

$$\Lambda = \{y \in \partial\mathcal{WO} : x(\lambda) + y(1 - \lambda) \in Q_{free}, \forall\lambda[0, 1]\}.$$

El conjunto $\mathcal{WO}$ se obtiene lanzando un rayo desde x a q_{goal}, y guardando el obstáculo mas cercano con el que hubo una intersección. Y Λ usa un proceso similar al usado para obtener O_i, solo que no registra discontinuidades, si no que todos los puntos detectados.

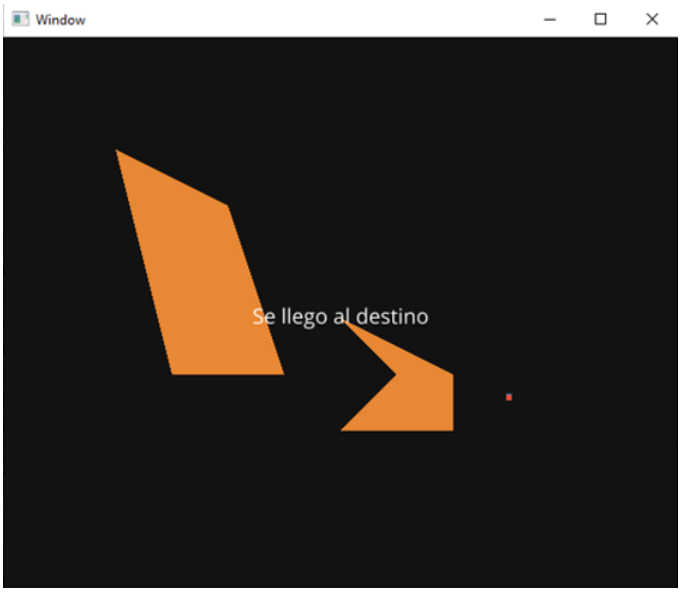
Ahora, lo que haremos es actualizar constantemente $d_{reach}, d_{followed}$ y $\{O_i\}$ a medida que nos movemos hacia $n \in \{O_i\}$ que esta en la dirección de frontera elegida.

Seguimos la dinámica anterior hasta que se cumpla una de las siguientes condiciones:

1. Encontremos a q_{goal}.
2. El robot completa un ciclo.
3. $d_{reach} < d_{followed}$.

Resultado.

A continuación, podemos apreciar una captura del código tras haber sido ejecutado.

Figura 2: 

No hay capturas del proceso debido a que el programa no puede procesar todos los movimientos del robot.