UNIVERSIDAD CALOS III DE MADRID

Trabajo Final

Robótica

Departamento de Ingeniería de Sistemas y Automática

04/12/2014

Sergio Sánchez Martin 100292704
María Elvira Tomeo Martin 100292974

ÍNDICE:

Controlador para 'Rescate y Salvamento'

1. Descripción del problema

El proyecto se basa en la implementación de un controlador de un robot para realizar un rescate y salvamento.

Durante la simulación el robot tendrá que atravesar el escenario sorteando los diferentes obstáculos que encuentre a su paso hasta llegar al final del recorrido donde deberá reconocer a dos víctimas y regresar al punto de partida.

Cada víctima será representada como un cilindro verde y para simular su reconocimiento y posterior rescate, el robot deberá pararse a un metro de distancia, esperar dos segundos y dar una vuelta completa sobre sí mismo.

Para la simulación del proyecto utilizamos el simulador de robótica comercial *CyberboticsWebots (v7.2.4)* como robot utilizaremos el robot móvil Pioneer II

La simulación de la eficacia del controlador tendrá diez posibles escenarios, dichos entornos simulan una serie de daños provocados por diferentes catástrofes, condiciones de oscuridad, de humo…Todos los escenarios cuentan con dos líneas amarillas situadas al inicio y final del entono.

2. Proceso de diseño

El primer problema que debíamos resolver para realizar el diseño de la solución fue la estructuración del problema, subdividiéndolo en bloques e implementando funciones para la realización de éstos.
Por lo que nuestro controlador se basa principalmente en dos bloques diferenciados:

- Navegación
- Búsqueda

Para la realización del controlador contábamos con diferentes sensores del robot Pioneer II, entre ellos se encuentran:

- Brújula ("Compass").
- GPS ("gps").
- 16 sensores de proximidad ("ds0- ds15").
- Cámara frontal ("Forward camera").
- Cámara omnidireccional ("Spherical camera").

De los cinco sensores citados, descartaremos la utilización del GPS y la cámara esférica por los siguientes motivos:

- Por un lado pensamos en un inicio que el GPS nos podría servir para devolver la posición del robot por lo que nos ayudaría en la navegación hacia las personas, tras consultar las características del GPS nos dimos cuenta de que el error de precisión del GPS es muy elevado y el entorno de simulación nos resultaba muy pequeño con respecto al error, por lo que pensamos que el GPS nos daría en numerosas ocasiones una posición errónea del robot lo que ocasionaría errores al controlador al atravesar el entorno y localizar a las personas.
- Tampoco utilizaremos la cámara esférica en la implementación de nuestro controlador, dado que nos muestra una imagen de 360º mostrando todo lo que se encuentra alrededor del robot por lo que consideramos que la lógica de control para la correcta interpretación de la imagen aumentaría en dificultad.

Por lo que finalmente utilizaremos:

- **Brújula:** Con la ayuda de la brújula conseguimos orientar el robot a la posición deseada para que pueda realizar la navegación por el entorno correctamente.
- **Sensores de proximidad:** De los dieciséis sensores disponibles solo utilizaremos cuatro, dos frontales y dos laterales, esta decisión la tomamos con el fin de simplificar la lógica de control lo máximo posible.
- **Cámara frontal:** Utilizaremos esta cámara para poder realizar la localización de las víctimas en cada entorno.

3. Descripción de solución

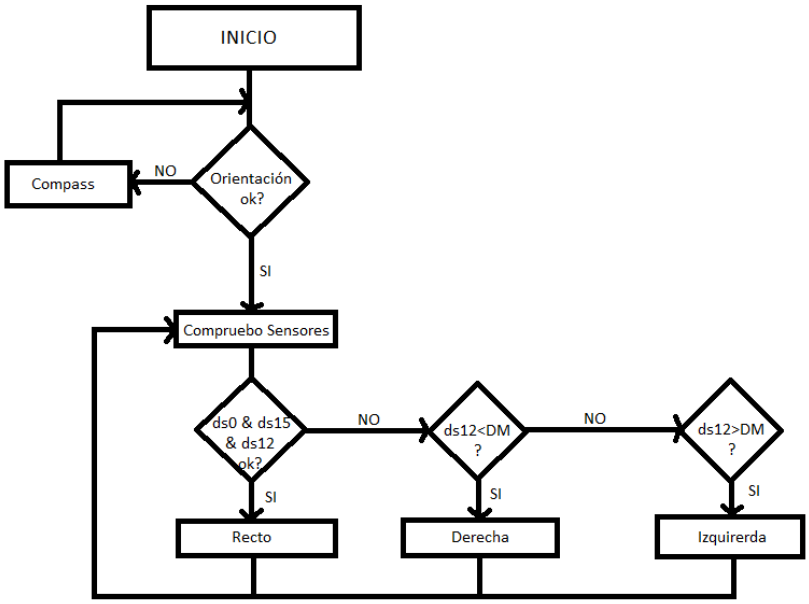
Como hemos indicado anteriormente hemos subdividido el problema en dos grandes grupos, que eran *navegación* y *búsqueda*:

Navegación:

Para la realización de la navegación utilizaremos cuatro de los dieciséis sensores de proximidad disponibles, que son dos frontales (ds0 y ds15) y dos laterales (ds3 y ds12). Además utilizaremos la brújula para orientar al robot en la posición necesaria al inicio de la navegación.

La lógica usada para la navegación consiste en el seguimiento de una pared del escenario. Aunque no resulta la solución más rápida si que consideramos que resulta algo más óptima dado que conseguimos evitar los obstáculos llegando hasta el final del escenario sin colisiones.

Una simplificación de la lógica para esta función es la siguiente:



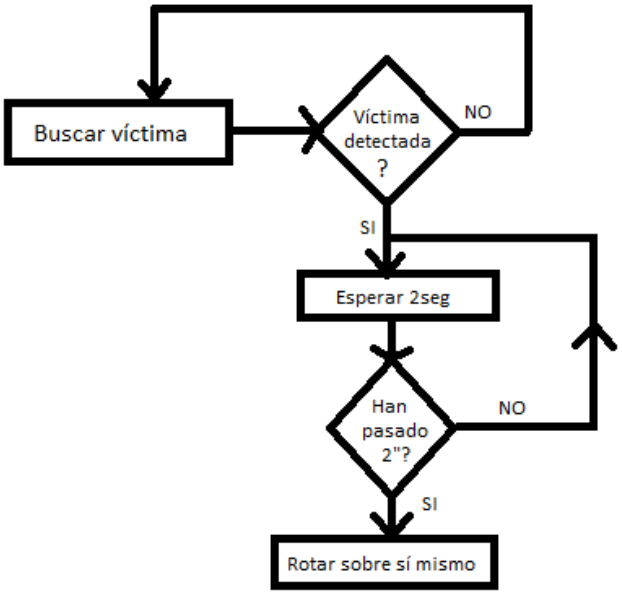
Siendo DM la distancia máxima permitida.

Búsqueda:

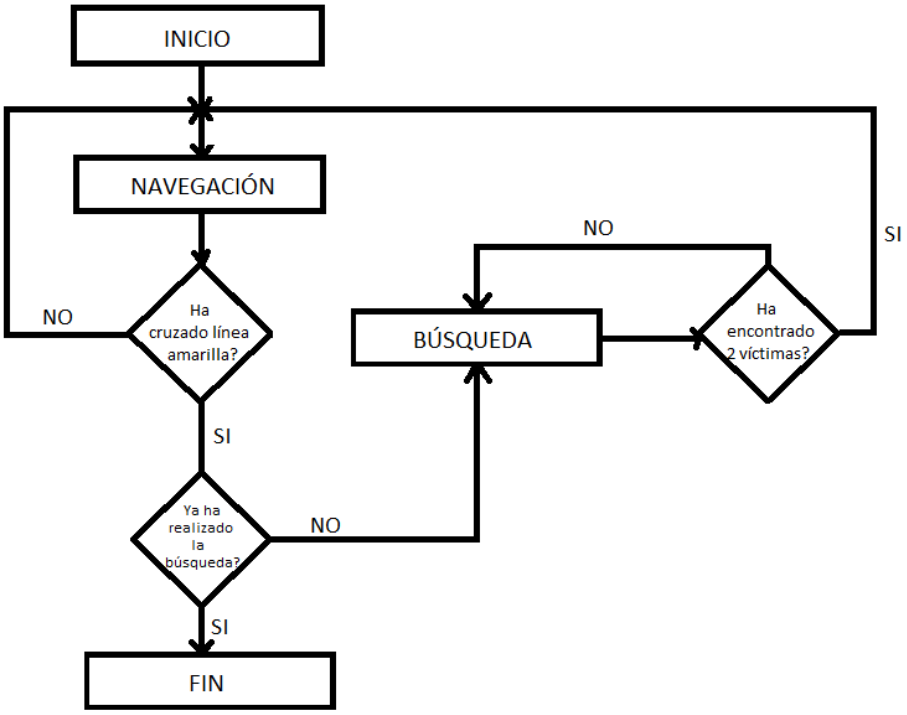
Para la realización de la búsqueda utilizaremos la cámara frontal y el robot se moverá según el porcentaje de verde buscando la mayor cantidad, es decir buscando a la víctima.
Una vez que se encuentre a un metro de la víctima parará y girará una vuelta completa sobre sí mismo para así continuar la búsqueda de la segunda víctima.
La cantidad de verde se obtiene en código RGB y se calcula el porcentaje del total de la imagen.

Para la realización de la espera y la rotación se han implementado dos funciones que realizan estas acciones.

Una simplificación de la lógica de control para la función búsqueda es la siguiente:



El esquema principal de la lógica del controlador sería:



Simplificaciones de código:

- **Variable emun:** Para las sentencias de movimiento del robot hemos creado una variable global de tipo enum llamada _mode, esta variable tiene distintos modos (TURN_LEFT, TURN_RIGHT…) implementado cada modo con sentencias diferentes, diferentes velocidades para cada rueda del robot, para conseguir que el robot realice la sentencia deseada. Creamos esta variable ya que reduce mucho código, tu creas una vez que el modo TURN_LEFT, y siempre que necesitas torcer a la izquierda basta con asignar este modo.

- **Funciones:** Las funciones también ayudan a depurar el código, hacen que el código pueda ser entendido por más personas. En lugar de crear una única función y meter todo el código en esa función, hemos dividido esta función en varias (por ejemplo una función para navegar, otra para detectar la línea amarilla, otra para esperar…). Cuando queremos utilizar alguna de las funciones nos basta con llamar a la función.

4. Descripción de los resultados obtenidos

- Escenario 1

	Tiempo(min)	**Personas encontradas**	**Líneas negras a la vuelta**	**Colisiones**
Prueba 1	2.59	1	0	Una grave
Prueba 2	2.03	1	0	Una grave
Prueba 3	2.52	1	0	Una grave

En las tres pruebas realizadas el robot colisionaba fuertemente con alguna de las paredes, para poder continuar la simulación hemos movido el controlador manualmente para ver si conseguíamos obtener un resultado satisfactorio.

- Escenario 2

	Tiempo	**Personas encontradas**	**Líneas negras a la vuelta**	**Colisiones**
Prueba 1	2.25	0	3	Una grave
Prueba 2	2.23	0	0	Una grave
Prueba 3	2.14	0	3	Una grave

En este entorno el robot no detecta ni el color amarillo ni el verde, por lo que toma las personas como paredes, colisiona con una de las personas por lo que necesita colocación manual para intentar terminar el recorrido de vuelta.

- Escenario 3

	Tiempo	Personas encontradas	Líneas negras a la vuelta	Colisiones
Prueba 1	2.11	0	3	Una grave
Prueba 2	0	0	0	Una grave
Prueba 3	2.16	0	3	Una grave

En el tercer escenario ocurre lo mismo que en el segundo, con la luz con menos potencia el robot no detecta ni el color verde ni el amarillo, por lo que vuelve a tomar las personas como si fuesen pared, en la segunda prueba el robot entra en un bucle en la mitad del escenario sin lograr salir de los obstáculos lo que hace imposible que llegue al final del escenario.

- Escenario 4

	Tiempo	Personas encontradas	Líneas negras a la vuelta	Colisiones
Prueba 1	1.45	0	5	Una grave
Prueba 2	1.30-2.35	0	0	Una leve

En este escenario las cámaras tampoco detectan el color, en la segunda simulación pasa la línea amarilla pero vuelve hacia atrás por lo que pasa la línea amarilla dos veces. En este mundo una de las personas se encuentra antes de la línea amarilla por lo que el número máximo de personas encontradas seria una persona.

- Escenario 5

	Tiempo	Personas encontradas	Líneas negras a la vuelta	Colisiones
Prueba 1	2.23	1	3	Una grave
Prueba 2	0	0	0	0
Prueba 3	2.21	1	3	Una grave

En el quinto escenario las cámaras si detectan color, pero el robot no llega recto a la persona por lo que finalmente acaba chocando contra ella, tiene una colisión grave con la persona que necesita de ayuda manual para poder continuar la simulación. En la segunda simulación el robot queda atrapado entre los obstáculos del entorno.

- Escenario 6

	Tiempo	Personas encontradas	Líneas negras a la vuelta	Colisiones
Prueba 1	1.51	0	5	0
Prueba 2	1.31	0	Todas	0

En este escenario toma las personas también como objetos pero al no estar estas situadas cerca de la pared el robot recorre el mundo en la primera ocasión pasando cinco líneas negras a la vuelta y en la segunda simulación consiguiendo llegar al final del escenario.

- Escenario 7

	Tiempo	Personas encontradas	Líneas negras a la vuelta	Colisiones
Prueba 1	1.56	0	3	Una grave
Prueba 2	0	0	3	Una grave

En este escenario las personas siguen siendo objetos por lo que no son detectadas por las cámaras y las esquiva.

- Escenario 8, 9, 10.

  En estos escenarios el robot no es capaz de realizar la navegación correctamente ya que en el inicio no consigue orientarse correctamente.

5. Conclusiones y futuras mejoras

La primera dificultad que hemos encontrado a la hora de realizar el controlador ha sido la falta de conocimientos en programación, dado que la base que teníamos no era muy buena.
Por lo que a pesar de haber intentado realizar un código claro y bien estructurado, no lo hemos conseguido tanto como pretendíamos, y a pesar de tener los conceptos claros de que es lo que queríamos realizar no disponíamos de las técnicas necesarias para implementarlo.
A pesar de estos problemas técnicos hemos conseguido realizar una función para la navegación bastante buena y óptima ya que en casi todos los escenarios funciona correctamente, en cambio la función para la búsqueda no ha sido tan satisfactoria como esperábamos pudiendo decir que solo nos consigue localizar a una víctima y a veces chocando contra la misma.

Otro problema con el que hemos tenido que lidiar ha sido la poca repetitividad de la solución del controlador, es decir, el controlador no siempre funciona de igual forma, ya sea trabajando en distintos entornos o en el mismo.

El inconveniente de no poder trabajar desde casa ha sido otro problema añadido para no poder obtener el resultado esperado, a pesar de las numerosas horas empleadas para esta tarea.

La principal mejora que necesitaría el controlador implementado sería la optimización de la navegación para reducir el tiempo que tarda en la misma, para ello se debería mejor la lógica diseñada.
Además de conseguir utilizar las cámaras disponibles correctamente para la localización de las víctimas.