UNIVERSIDADE DA CORUÑA

Escola Politécnica Superior

Trabajo Fin de Grado

CURSO 2018/19

DESARROLLO DE LIBRERÍAS DE CONTROL PARA APLICACIONES DE COORDINACIÓN EN ROBÓTICA COLECTIVA

Grado en Ingeniería en Tecnologías Industriales

ALUMNO

Marcos Parada Pombo

TUTORAS/ES

Abraham Prieto García

FECHA

DICIEMBRE 2018

1 Título y Resumen

Desarrollo de librerías de control para aplicaciones de coordinación en robótica colectiva

En el presente proyecto se desarrollarán librerías de control de coordinación de robótica colectiva para el mapeado de las condiciones ambientales en un determinado espacio de trabajo.

Para el desarrollo de este, se han empleado un grupo de robots de plataforma móvil con smartphone llamada Robobo. Además, se ha realizado una calibración del robot, tanto de sus movimientos como de los sensores.

Se ha desarrollado un sistema de navegación en interior, con sus respectivos algoritmos para su correcto funcionamiento, incluyendo algoritmos de evasión de obstáculos y choques entre robots, mediante etiquetas QR. Estas etiquetas serán visualizadas por los robots a través de la cámara del Smartphone.

Por último, se han sincronizado los datos obtenidos de los sensores con la posición en la que se han tomado, y así de esta manera realizar un mapeado inteligente.

Desenvolvento de librerías de control para aplicacións de coordinación en robótica colectiva

No presente proxecto desenvolveranse librerías de control de coordinación robótica colectiva para o mapeado das condiciones ambientais dun determinado espazo de traballo.

Para o desenvolvemento de este, empregouse un grupo de robots de plataforma móvil con smartphone chamada Robobo. Ademais, realizarase unha calibración do robot, tanto dos seus sensores coma dos seus movementos.

Desenvolveuse un sistema de navegación en espazos interores, cos seus respectivos algoritmos para un funcionamento correcto de este, incluindo algoritmos de evasión de obstáculos e choques entre robots, mediante etiquetas QR. Estas etiquetas seran visualizadas polos robots a traves da cámara do smartphone.

Por último, sincronizaranse os datos obtidos polos sensores ca posición na que se tomou esos valores a un servidor e desta maneira realizar un mapeado intelixente.

Development of control libraries for robotics coordination applications collective

This Project carries out the development of colective robotical coordination control for mapping the enviromental conditions of a concrete workspace.

For this development, a group of robots have been used along with the Smartphone robotic platform called Robobo. Also, movement and sensors from these robots were calibrated.

An indoor navigation system has been developed, with its respective algorithms for correct operation, including algorithms for obstacle avoidance and robot collisions, using QR tags. These tags will be visualized by the robots through the camera of the Smartphone.

Finally, sensor data have been synchronized with the position where they where taken in order to carry out an intelligent mapping.

UNIVERSIDADE DA CORUÑA

Escola Politécnica Superior

TRABAJO FIN DE GRADO

CURSO 2018/19

DESARROLLO DE LIBRERÍAS DE CONTROL PARA APLICACIONES DE COORDINACIÓN EN ROBÓTICA COLECTIVA

Grado en Ingeniería en Tecnologías Industriales

Documento I

MEMORIA

Contenido

1 Título y Resumen 2
2 Introducción 9
3 Objetivos 10
4 Robótica 11
4.1 Plataformas móviles con smartphone 14
4.2 Robótica colectiva 15
5 Robobo 17
5.1 Hardware 17
5.1.1 El smartphone 17
5.1.2 La base 17
5.2 Software 18
5.2.1 Javascript 18
5.2.2 ROS (Robot Operating System) 18
5.2.3 Elección del entorno de programación 19
6 Odometría 20
6.1.1 Calibración del movimiento del Robobo 20
6.1.2 Desplazamiento recto 21
6.1.3 Rotación pura 22
7 Localización 25
7.1 Balizas naturales 25
7.1.1 SLAM (Simultaneous Localization and Mapping) 26
7.2 Balizas artificiales 27
7.2.1 April Tags Fiducial System 27
7.3 Elección del sistema de orientación 28
7.4 Etiquetas QR 28
7.5 Posición relativa respecto al QR 29
7.6 Ángulo de orientación respecto al QR 33
7.7 Factor de correlación coordenadas a centímetros 33
8 Navegación 34
8.1 Esquemas de navegación en robots móviles 34
8.2 Movimiento entre dos puntos 34
8.3 Navegación absoluta con balizas QR 35
8.4 Navegación mixta con balizas QR 36

8.5 Algoritmo de evasión de obstáculos....37
9 Mapeado....41
10 Caso de aplicación: monitorización de niveles de contaminantes....42
10.1 Riesgos laborales en el centro de trabajo....42
10.1.1 La prevención de los riesgos laborales (PRL)....42
10.1.2 Factores de riesgo....42
10.1.3 Iluminación....43
10.1.4 Ruido....47
10.2 Calibración de sensores de Robobo....51
10.2.1 Calibración del sensor de luminosidad....51
10.2.2 Calibración sensor de sonido....53
10.3 Mapeado de contaminantes a tiempo real....54
10.3.1 Resultados....56
11 Presupuesto....58
12 Conclusiones....59
13 Bibliografía....60
14 Anexo....62

Índice de Figuras

Figura 1 Primer robot móvil 11
Figura 2 Robot moderno 12
Figura 3 Tipos de robots según su arquitectura 13
Figura 4 Plataforma móvil con smartphone móvil 14
Figura 5 Sistema multi-robot 15
Figura 6 Robobo 17
Figura 7 Base Robobo 18
Figura 8 medición distancia calibración movimiento recto 21
Figura 9 Robot guiado por April TagsGráfica 2 Calibración movimiento recto 22
Figura 10 Medición giro calibración 23
Figura 11 Robot guiado por April Tags 27
Figura 12 Etiqueta QR 28
Figura 13 Desplazamiento entre dos puntos 35
Figura 14 Pseudocódigo navegación absoluta con balizas QR 36
Figura 15 Pseudocódigo navegación mixta 37
Figura 16 Algoritmo de evasión de obstáculos 38
Figura 17 Pseudocódigo salvar obstáculos 39
Figura 18 Refracción de luminancia sobre una superficie 44
Figura 19 Incidencia de luz sobre una superficie 45
Figura 20 Sonómetro 50
Figura 21 Calibración luxómetro 52
Figura 22 Calibración luxómetro 52
Figura 23 Calibración sonómetro 53
Figura 24 Mapa de espacio de trabajo 54
Figura 25 Medición de niveles contaminantes con sistema multirobot 55

2 INTRODUCCIÓN

En la actualidad, la sociedad presenta un proceso de avance continuo, este hecho es gracias a la capacidad del ser humano para progresar y crear conocimiento. Este avance es palpable en todos los aspectos de esta, tanto en el ámbito personal como en el profesional.

En los últimos años la robótica ha presentado un avance de gran índole en nuestra sociedad, insertándose en nuestra vida de tal forma que la consideramos indispensable en determinados ámbitos. La robótica en la actualidad está tendiendo a sustituir al ser humano en procesos peligrosos, procesos de repetición o de precisión o simplemente como una ayuda al individuo. En los últimos tiempos ha empezado a dar sus primeros pasos la robótica colectiva, esto es cuando un conjunto de robots colabora en la realización de una misma tarea.

Hoy en día una gran parte de las personas tienen como espacio de trabajo una oficina. De esta manera está estipulado por ley una serie de condiciones que se tienen que tener los espacios de trabajo. Está estudiado que los trabajadores son más productivos cuando las condiciones de trabajo son las óptimas, estos son los principales motivos por los cuales las empresas buscan adecuar las condiciones de trabajo a unos estándares estipulados.

Dicho esto, en el presente trabajo de fin de grado se ha desarrollado un sistema de navegación autónoma con una plataforma de robótica móvil con smartphone, llamada Robobo, desarrollada en el Grupo Integrado de Ingeniería (GII) de la Universidade Da Coruña (UDC), capaz de detectar los diferentes obstáculos e identificar los otros robots que están desempeñando la misma tarea. Este robot recogerá los datos de luminosidad y ruido obtenidos con los sensores del smartphone.

Una vez obtenidos los datos anteriores se realizará un mapeado de los mismos para determinar las condiciones más desfavorables para el desempeño del trabajo requerido en óptimas condiciones.

3 OBJETIVOS

El objetivo principal del presente proyecto es desarrollar una serie de herramientas que permita la navegación y localización en interiores de uno o más robots. Este objetivo se divide en tres subobjetivos:

- Calibración de movimientos y sensores
- Localización
- Navegación y mapeado

Donde en el primero se realizará un estudio de la respuesta de los diferentes sensores y movimientos del robot que se utilizará para la ejecución del presente trabajo de fin de grado.

En segundo lugar, realizaremos un sistema de localización del robot en cuestión mediante la visualización de balizas QR y de esta manera conocer nuestra posición en un sistema de coordenadas absolutas.

Por último, se desarrollará un sistema de navegación multirobot en interiores, monitorizando los resultados de los sensores de luminosidad y ruido y de esta manera realizaremos un mapeado de las condiciones ambientales en un determinado espacio de trabajo.

4 ROBÓTICA

En este capítulo repasaremos la historia de la robótica y realizaremos una visión global de la robótica en la actualidad. Además, se presentarán diferentes plataformas robóticas con smartphone y de su utilidad. Para concluir hablaremos de la robótica colectiva y sus diversos enfoques.

La robótica es la ciencia que está involucrada en el diseño, fabricación y utilización de robots. Por otra parte, un robot es una máquina que puede ser programada para que interactúe con objetos y que realice un comportamiento deseado.

En la robótica se combinan la electrónica, la informática, la mecánica, y la ingeniería entre otras muchas disciplinas. El objetivo principal de la misma es realizar dispositivos que funcionen de forma automática y que realicen una o varias tareas deseadas, estas tareas pueden ser de diversa índole, como tareas simples en las que se sustituye al ser humano, automatizándolas, o de manera que ayude o complemente a este, tareas de elevada precisión o tareas peligrosas.

El inicio de la robótica puede establecerse en el siglo XVIII en diferentes industrias, desde la textil hasta la de facturación de piezas, con la automatización de ciertos procesos. También durante el siglo XVII y XVIII se construyeron en Europa ciertas máquinas que tenían ciertas cualidades de robots.

A mediados del siglo XVIII Jacques de Vauncansos varios robots mecánicos capaces de hacer música. En 1805, Henri Maillardert, mecánico suizo, fue capaz de construir una muñeca mecánica que realizaba dibujos, esta muñeca utilizaba una serie de levas para escribir y dibujar. Cabe decir, que estos casos se deben considerar aislados y adelantados a su época.

El término “robot” surgió por primera vez en 1921, en una obra de teatro llamada “R.U.R.” o “Los Robots Universales de Rossum” del escritor checo Karel Capek, este término surgió de la unión de los términos “robota” que puede definirse como “trabajo forzado” y en “rabota” cuyo significado es el de “servidumbre”. El escritor Isaac Asimov es considerado como el precursor del término “robótica”, es también el padre de las “tres leyes de la robótica”, una norma para el accionamiento de los robots, ya que en sus novelas de ficción se relataban como los robots invadían mundos y sometían a los seres humanos.

Las primeras patentes de robots aparecieron en 1946 con los robots de Devon, cuya función era el traslado de maquinaria. Ese año también estaría marcado por la aparición de las primeras computadoras, el ENAC construido en la Universidad de Pensinvalnia por J. Presper Eckert y John Maulchy, y la primera máquina digital de propósito general desarrollada en el MIT.

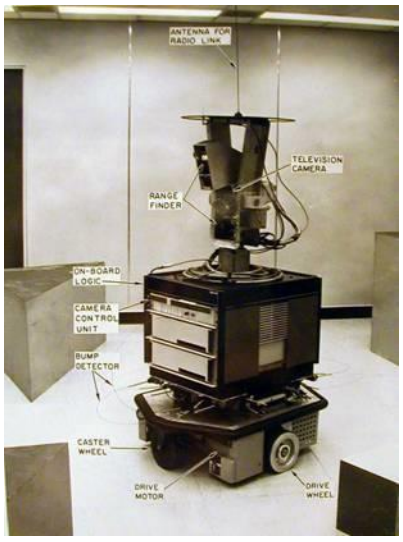


Figura 1 Primer robot móvil

Gracias a los diferentes avances de la tecnología en lo que restaba de siglo, tanto en capacidad de computación, sensorización y sistemas de actuación a la par de la investigación sobre las diferentes teorías de inteligencia artificial, la robótica seguiría avanzando. De aquí nace el robot como conocemos hoy en día.

A parte de los hechos citados anteriormente se le tiene que unir una gran demanda de los robots, debido a la revolución industrial en las diferentes industrias para explicar el increíble auge de estos. La aparición de la producción en cadena en la industria fue uno de los hechos vitales para el gran aumento de la demanda de estos, ya que podían sustituir al trabajador en ciertas tareas por robots y así abaratar costes o mejorar el proceso productivo.

En los últimos tiempos la robótica ha contemplado un inmenso desarrollo, de tal forma que los robots llegaron a ser considerados como el paradigma de la automatización, se han convertido en nuestros días en un elemento más de dicha automatización.

Hoy en día, gracias a los avances en los campos como la informática y la electrónica entre otros muchos, los robots han sido dotados de una gran flexibilidad y capacidad de adaptación al entorno. En la actualidad los robots inteligentes además de detectar las modificaciones del entorno, estos tienen la capacidad de memorizar sus acciones. De este modo los robots tienen un periodo de aprendizaje y una vez se le presente una tarea similar puede inspeccionar su memoria para realizarla de manera más eficiente.

Figura 2 Robot moderno

Estas cualidades son las que hacen a los robots como una herramienta de gran utilidad para la sociedad, pudiendo reemplazar en ciertas tareas al ser humano. Principalmente en las cuales sean peligrosas, monótonas o simplemente tengan una capacidad mayor a la del ser humano.

Los robots se pueden clasificar de diversas maneras, según su cronología (primera generación, segunda…), de esta manera se clasifican en función de las capacidades que podría tener el mismo según su época de lanzamiento. Según su arquitectura, así se clasifican los robots en función de su tipo de configuración general del robot. Y por último según su aplicación. A continuación, incidiremos más detalladamente en las dos últimas.

La clasificación según su estructura establece una diferenciación entre los robots según su diseño, el cual enfocara al robot en una actividad más específica. La clasificación de los robots en base a su arquitectura es la siguiente:

- Poliarcitulados

  Robots inmóviles prácticamente que están diseñados para la realización de una tarea en un determinado espacio de trabajo, como se puede observar en la figura 1.

- Móviles

  Se caracterizan por tener una gran capacidad de desplazamiento. Realizan un recorrido por telemando o guiándose por la información recibida del entorno a través de sus sensores.

- Androides

  Estos robots intentan reproducir parcialmente o en su totalidad el comportamiento cinemático del ser humano. Su principal utilidad es la investigación debido a su escaso desarrollo.

- Zoomórficos

  Dentro de este grupo también se podrían considerar los androides, ya que los robots zoomórficos intentan reproducir el comportamiento de los diversos seres vivos.

- Híbridos

  Son aquellos de difícil clasificación cuya estructura se sitúa en combinación con alguna de las anteriores.

Figura 3 Tipos de robots según su arquitectura

Según su aplicación podemos distinguir entre los robots industriales y los de servicios. Los robots industriales es una máquina de manipulación automática, reprogramable y multifuncional con varios grados de libertad que le hacen capaz de posicionar y orientar materiales, piezas, herramientas o dispositivos para la realización de tareas. Este tipo de robots suelen ser poliarticulados, ya que su movilidad es limitada y el entorno en el que se desenvuelve, el centro de trabajo está bajo control. Los robots de servicios son dispositivos electromecánicos que actúan de forma parcial o totalmente autónoma que realizan servicios que mejoran el bienestar de las personas. Dentro de este grupo se encuentra robots de diversa índole, de investigación, militares, médicos, nano robots, educacionales y de entretenimiento.

Robobo, el robot empleado en el presente trabajo de fin de grado es un robot móvil de servicios que sus principales cualidades son su precio reducido en combinación con sus capacidades computacionales. Estas hacen a Robobo perfecto para un sistema de multirobots. Se puede programar tanto a nivel educacional con herramientas como Scratch como para investigación con herramientas de alto nivel.

4.1 Plataformas móviles con smartphone

Unos de los principales problemas de los robots son su alto precio y que se quedan obsoletos rápidamente en temas como investigación. Una solución a estos problemas es el aprovechamiento de los smartphones como “cerebro” gracias a su gran capacidad de actualización y su accesibilidad. A parte de este hecho, el smartphone nos permite una enorme conectividad, cualidad que lo hace aún más interesante ya que nos permite conectarnos a otros dispositivos. En cuanto a la actualización de hardware, consiste en cambiar de móvil por uno más actual, este hecho hace que la plataforma tenga una vida útil más larga.

El primer robot plataforma con smartphone fue el ROMO, esta plataforma robótica compatible con sistema operativo IOS permitía una programación básica, aunque carecía de software de programación de bajo nivel, el cual no lo hace apto para la investigación. A este inconveniente le hay que sumar la carencia de sensores en la plataforma.

Figura 4 Plataforma móvil con smartphone móvil

Seguidamente se fabricaron otros robots de este tipo, pero siempre con alguna característica que no lo hacía idóneo para la investigación. Por ejemplo, el ODDWERX, el cual, si disponía de un software de programación de bajo nivel, pero carecía de sensores en la plataforma que le aportasen autonomía. Esta carencia la cubría el WHEELPHONE, pero carecía de elementos para comunicarse con el exterior y la conexión con el smartphone se realizaba mediante un cable.

Las diferentes carencias anteriormente de los diferentes robots llevaron al Grupo Integrado de Ingeniería (GII) de la Universidad de La Coruña (UDC) a desarrollar su propia plataforma móvil con smartphone. Este será el robot empleado para la realización del presente trabajo de fin de grado.

4.2 Robótica colectiva

Un sistema multi-robot se puede definir como un conjunto de robots que operan sobre la misma tarea. Suele ser aplicado para resolver problemas colectivos (aquellos que requieren m´as de un robot para ser resueltos) y distribuidos, pero también se puede utilizar en problemas que no requieren una solución colectiva, pero en los que resolver la tarea colectivamente produce algún beneficio. De manera general, las principales ventajas de los sistemas multi-robot son las siguientes:

- son capaces de resolver tareas complejas al dividir la tarea global en otras más sencillas.
- aumentan la eficiencia en la resolución de la tarea sí esta se puede realizar en paralelo
- son robustos y tolerantes a fallos debido a su redundancia
- son flexibles y escalables. Pueden estar formados con robots de diferentes capacidades (sensores y actuadores) o con las mismas, y pueden ser tolerantes a cambios en el número de integrantes del equipo.

Y sus principales desventajas son la de aumentar la complejidad de los problemas y la comunicación entre los diferentes robots y o un servidor.

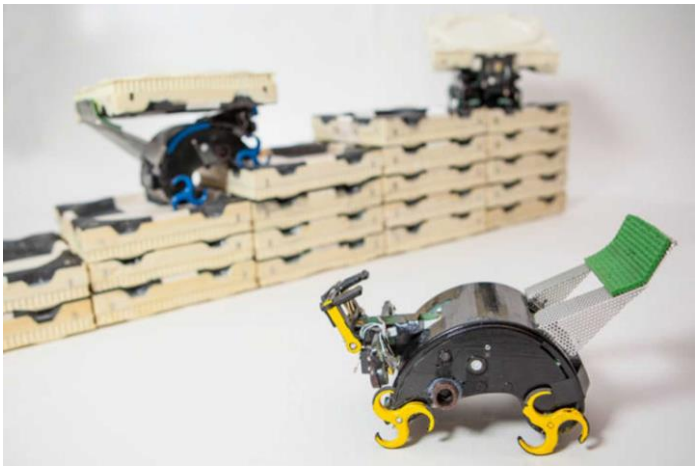


Figura 5 Sistema multi-robot

Los enfoques de la robótica colectiva son muy diversos debido a la gran variedad de problemas a solucionar mediante esta tecnología. De esta manera, se estudia desde los principales comportamientos del ser humano hasta los diversos modelos computacionales para aplicarlos a este campo y así buscar la mejor opción para la resolución de cada problema. Por consiguiente, los diferentes enfoques son:

- Etología
  Estudiando como cooperan y se comunican los animales y humanos.
- Organizacional
  Observando cómo funcionan las estructuras sociales y humanas.
- Modelos computacionales
  Aplicando conceptos de multiprocesamiento y diseño de sistemas paralelos.
- Inteligencia artificial distribuida
  Imitando soluciones de problemas de agentes distribuidos como la negociación, comunicación y evaluación.
- Análisis de movilidad
  Estudiando los modelos cinéticos y dinámicos de los problemas multiagente.

- Vida artificial

  Estudio de los ecosistemas artificiales, principalmente en la relación del conjunto con el entorno.

La comunicación es un recurso indispensable para cualquier trabajo en conjunto, la cual sirve para la sincronización, el intercambio de información y la negociación entre otras muchas aplicaciones. De este recurso se deben tener en cuenta los diferentes aspectos de esta:

- Necesidad

  A veces, la comunicación no es imprescindible, también cabe destacar que puede haber cooperación sin comunicación debido a que esta puede ser implícita a través del medio.
- Rango

  No siempre a mayor rango mejor, ya que existen diversos mensajes que no son importantes para individuos alejados.
- Contenido

  No siempre cuanto mayor detalle mejor, sino que debe de ser la información justa y necesaria.
- Robustez y rendimiento

  Debe transmitirse el mensaje de forma sólida y eficiente.

5 ROBOBO

Robobo es una combinación de una base robótica móvil, que será el cuerpo del robot, y un smartphone, que será el cerebro de este. Debido a la gran cantidad de sensores de los smartphones actuales y a la tecnología de estés, añadiéndole las capacidades de movilidad, la detección a bajo nivel y las capacidades gestuales proporcionadas por la base, Robobo se convierte en una buena herramienta para realizar determinadas tareas.

Figura 6 Robobo

Para programar el Robobo necesitamos 3 componentes, la base robótica y el smartphone citados anteriormente, y un ordenador. Estos 3 elementos son los que hacen del Robobo una herramienta tan atractiva, debido a que los elementos del mismo son de una gran accesibilidad. El funcionamiento de los distintos elementos es la siguiente, la base y el smartphone se sincronizan mediante bluetooth, mientras que a la vez el smartphone se conecta al ordenador a través de WIFI.

5.1 Hardware

El hardware del Robobo está compuesto por el smartphone y la base, desarrolladas a continuación.

5.1.1 El smartphone

Se encuentra sobre la base mediante un soporte. El smartphone es el principal elemento de Robobo. Este proporciona al robot una gran variedad de sensores de una gran calidad como la cámara, micrófono, giroscopio o acelerómetro entre muchos otros. Este también proporciona un gran abanico de comunicaciones y la capacidad de procesamiento necesaria para gestionar todos los elementos y ejecutar programas complejos.

5.1.2 La base

Esta contiene dos ruedas motorizadas, en la parte delantera de la base, con un motor cada una, que permite que el Robobo se mueva por la superficie con libertad ayudado por una deslizadera en la parte trasera del Robobo. Un trípode, el cual es el soporte del smartphone, este también esta motorizado con dos motores que le dan libertad de movimientos en el eje “z” e “y”, con los movimientos tilt y pan. También contiene 8 infrarrojos, 5 delante y 3 detrás, que le sirven para detectar la proximidad o no de objetos o superficies, cabe destacar que tiene otros dos en la parte inferior, uno en la parte delantera y otro en la trasera, para evitar los desniveles que no puede superar

y evitar caídas. En la base también se encuentran 7 luces LED que le permiten a Robobo comunicarse con el entorno. A parte de todos estos elementos de movimiento y sensorización, en el cuerpo están contenidos tanto la gran batería de 5000 mAh y la placa base de esta.

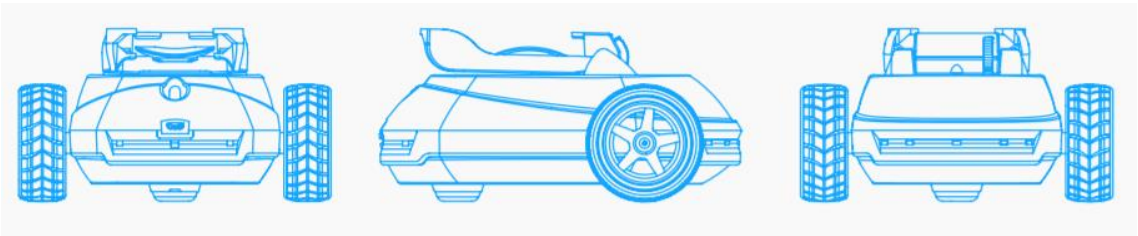


Figura 7 Base Robobo

5.2 Software

En este punto tenemos que describir de los dos tipos de software diferentes que tenemos, la aplicación y el entorno de programación. En primer lugar, vamos a hablar de la aplicación para móvil, en concreto para el sistema operativo Android, es la aplicación que debemos instalar en el smartphone para programar el Robobo. Esta es la que interactúa con todos los elementos tanto del smartphone como de la base, y tiene la capacidad a la vez de comunicarse con el entorno de programación. El entorno de programación, que será utilizado desde un ordenador, de los cuales existen tres niveles, Scratch para programadores principiantes, este software se utiliza principalmente en la enseñanza, javascript, para programadores intermedios y ROS para programadores avanzados. Combinando los dos softwares citados anteriormente tendremos acceso a todos los sensores y actuadores del Robobo.

Ahora describiremos un poco más detalladamente los entornos de programación que se han contemplado para realizar este proyecto, argumentando por cual nos hemos decantado para la ejecución del mismo.

5.2.1 Javascript

Javascript es un lenguaje ligero de programación interpretado, orientado a objetos. Para la programación de Robobo con JavaScript se utiliza el entorno de ejecución Node.js. Es un entorno de ejecución multiplataforma, de código abierto, creado principalmente para el desarrollo de programas de red.

5.2.2 ROS (Robot Operating System)

ROS es un framework con base de sistema operativo Ubuntu, para la escritura de software robot. Se puede decir que es un sistema operativo de código abierto que surgió para proporcionar a los desarrolladores y programadores de robots un conjunto de normas y herramientas estándar que permitan trabajar a los profesionales de manera más rápida y sencilla.

ROS lo que nos aporta son una serie de herramientas y atajos que nos permite una sencilla comunicación entre los diferentes nodos que intervienen en la programación de un robot. Por lo tanto, la principal ventaja de esto es que tenemos un lenguaje universal del cual no nos tenemos que preocupar por la comunicación de los diferentes nodos y además nos da mucha versatilidad en la programación.

5.2.3 Elección del entorno de programación

Cabe decir que uno de los principales problemas de la robótica es la implementación del lenguaje y la optimización de la comunicación de los diferentes servidores. Tras haber probado los dos entornos de programación citados anteriormente los retardos de ROS son mucho mayores de los de javascript, por lo tanto, se ha optado por este último para la elaboración del montante del siguiente trabajo de fin de grado.

6 ODOMETRÍA

La odometría es el estudio de la estimación de la posición de vehículos con ruedas durante su navegación. En los robots se utiliza para estimar su posición relativa frente a su localización inicial. El principal factor para que la odometría es tener un modelo preciso de las acciones del robot. De esta forma si conocemos las ordenes de movimiento podremos estimar la posición en la que nos encontramos después de dicho movimiento. De todos modos, es inevitable que existan ciertos errores y que por tanto a medida que se acumulan acciones se va acumulando el error en la estimación de la posición.

6.1.1 Calibración del movimiento del Robobo

La calibración es un proceso para ajustar los parámetros que definen el modelo de comportamiento de un robot. El comportamiento tiene ruido y eso producirá un error inherente al modelo independiente de que esté calibrado.

Los comandos no tienen asociado una respuesta exacta a su aplicación pues no es posible calcular la respuesta mecánica del robot ni la del motor en función de las ordenes enviadas porque influyen un gran número de factores que la modifican y es por eso por lo que hay que basarse en datos obtenidos en pruebas reales para generar el modelo. En la calibración se pueden combinar varios sensores para disminuir errores de medida. La calibración de Robobo la dividiremos en dos partes, movimientos y sensores. En este apartado nos dedicaremos a la calibración de los diferentes desplazamientos que puede realizar.

La calibración de los movimientos es fundamental para el desplazamiento del robot, de esta manera a parte de determinar los desplazamientos reales respecto a los movimientos programados, determinaremos a qué velocidad se deben de hacer los mismos debido a los deslizamientos con la superficie para evitar grandes errores.

6.1.2 Desplazamiento recto

Para la calibración de este movimiento se ha utilizado el comando de Robobo:

moveWheelsByTimeBLK(Vr, Vi, T)

Donde Vr es la velocidad de rotación de la rueda derecha, Vi la velocidad correspondiente a rueda izquierda y T el tiempo durante el cual se ejecutará el comando. De esta manera se ordenará a el robot que se mueva a diferentes velocidades durante diferentes periodos de tiempo para ver su respuesta. Para realizar la medición de manera correcta se ha de coger la medida de desplazamiento de la forma más precisa posible, de esta manera se ha optado por realizar la medida en el eje de las ruedas debido a su escasa distancia respecto al suelo y evitar errores ópticos.

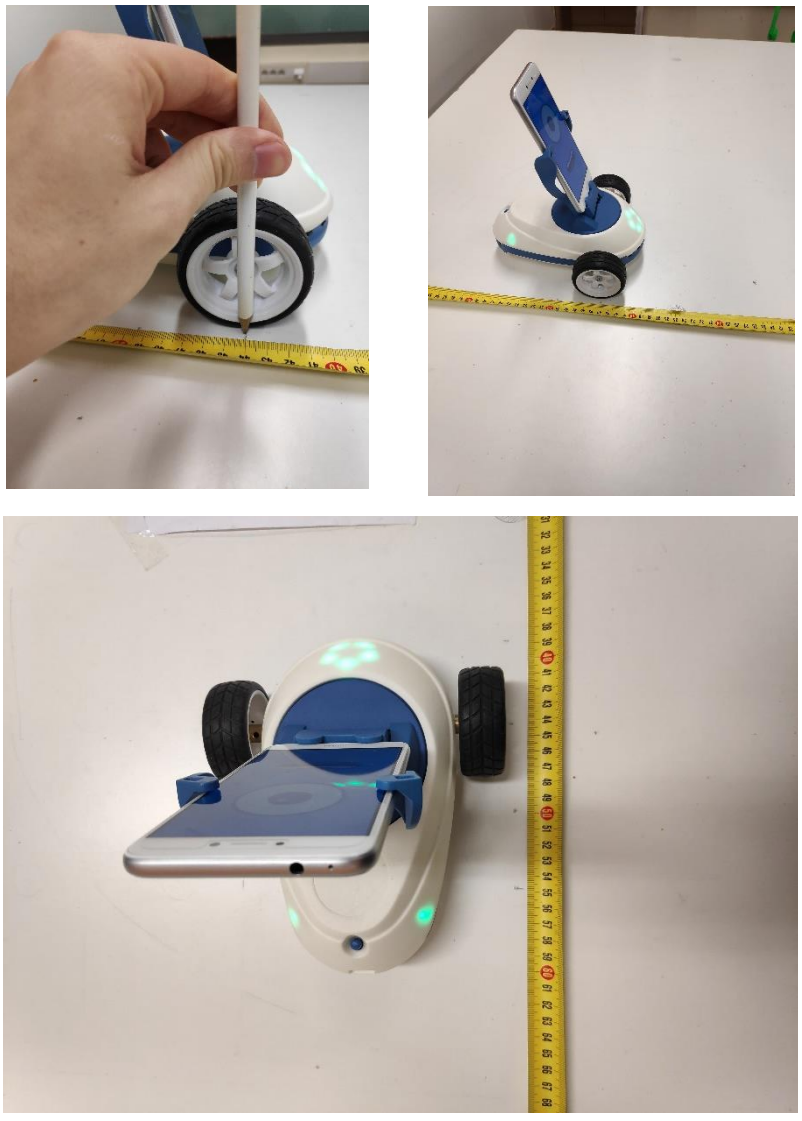


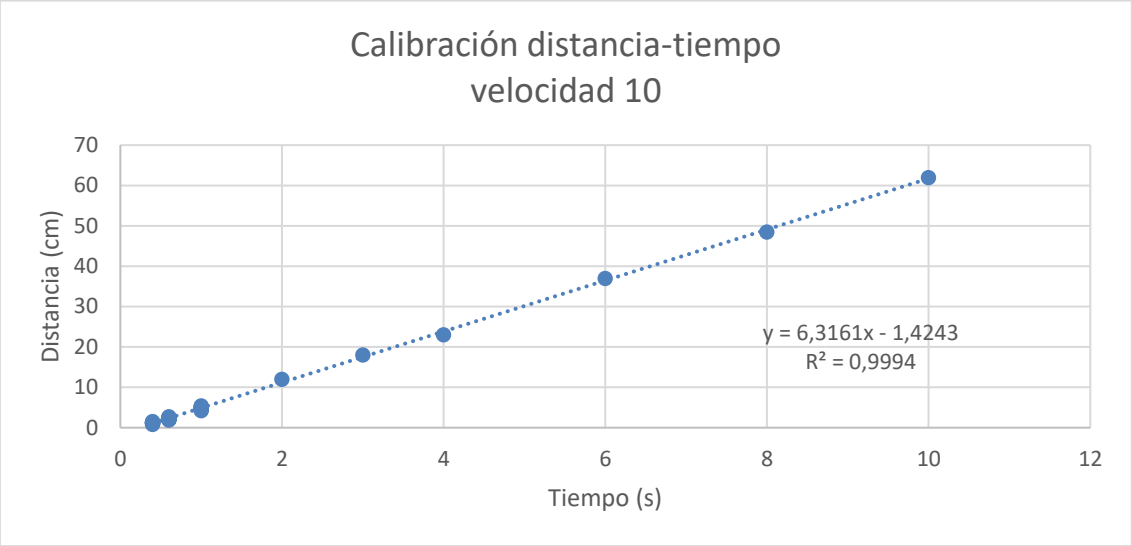
Figura 8 medición distancia calibración movimiento recto

Los datos de distancia en centímetros respecto a el tiempo obtenidos en la calibración son los observados en la taba 1. En la gráfica Y podemos observar que el Robobo se comporta de manera lineal en los desplazamientos rectos, con una regresión de 0,9994. El desplazamiento corresponde a la ecuación que se encuentra en el gráfico donde “x” es el tiempo en segundos e “y” a la distancia en centímetros. De las

ecuaciones obtenidas podemos concluir que el desplazamiento es lineal pero que este tiene no arranca en el instante cero debido a un pequeño retraso en la respuesta del robot.

tiempo (s)	0,4	0,4	0,4	0,4	0,4	0,4	0,4	0,4	0,4	1	1	1	1	1	1	0
distancia (cm)	1,2	1,2	1,3	1,5	1,1	1,2	1,2	1,2	0,8	5,4	4,2	4,5	4,8	5	5,1	0

Tabla 1 Datos calibración movimiento recto



Gráfica 1 Calibración movimiento recto

La calibración se ha hecho a diferentes velocidades, pero pudimos observar que a cuanto mayor velocidad más variación en la respuesta debido a deslizamientos por causa de la baja adherencia de las ruedas en la superficie.

6.1.3 Rotación pura

En la siguiente calibración se estudiará el comportamiento del robot en un movimiento de rotación en estático. Este movimiento podría estimarse a partir de la respuesta en línea recta, pero hemos comprobado que este hecho se corrobora debido a la variación de la fuerza de oposición al movimiento y los pequeños deslizamientos que siempre aparecen en un movimiento circular.

En esta calibración se realizará la medición de la misma manera que en la calibración anterior. Midiendo el movimiento sobre un transportador de ángulos impreso sobre una hoja de papel poniendo sobre esta el Robobo con el eje de giro de este en el centro de nuestro transportador impreso, haciéndolo girar con el comando anterior con velocidades con signos opuestos.

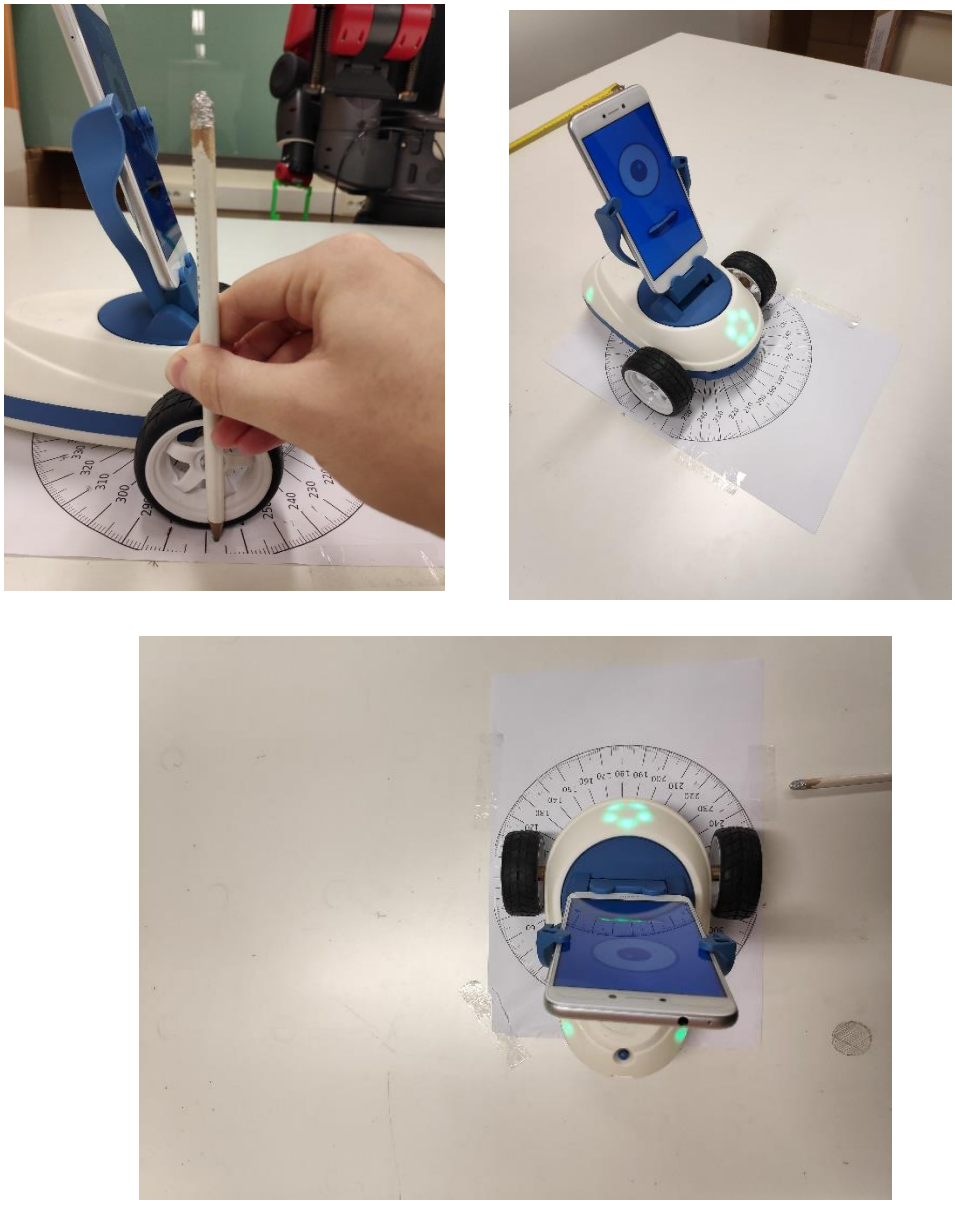


Figura 10 Medición giro calibración

Una vez obtenidos los datos de la tabla 2 vemos que no se comporta de forma tan regular como el movimiento frontal, este hecho tiene como motivo la medida de este ya que con un desplazamiento pequeño que puede ser causado por diversos motivos como los deslizamientos de las ruedas o el tiempo de respuesta del robot entre otros, puede hacer variar varios grados la medida. Aun así, la ecuación a pesar de usar un polinomio más complejo para ajustar el movimiento podemos obtener una aproximación completamente satisfactoria con un coeficiente de regresión de 0,99.

tiempo	0,25	0,25	0,25	0,25	0,25	0,25	0,25	0,25	0,25	0,25	0,3
grados por secuencia	1	0,5	1	0,5	0,5	0,5	1	1	1	1,5	1,5

tiempo	0,3	0,3	0,3	0,3	0,3	0,3	0,3	0,3	0,3	0,4	0,4
grados por secuencia	1,5	2	1,5	1,5	2	2,5	1,5	1,5	1,5	3	7

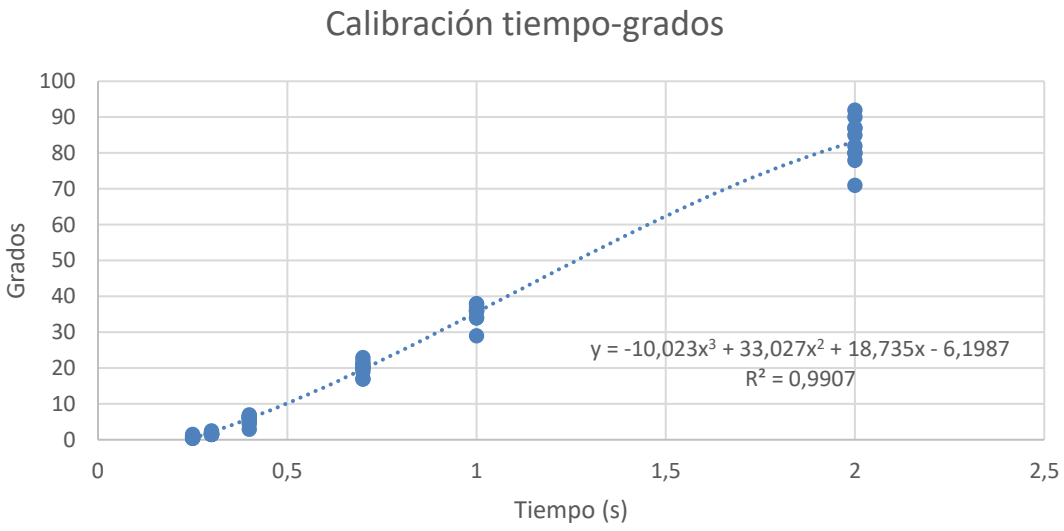
tiempo	0,4	0,4	0,4	0,4	0,4	0,4	0,4	0,7	0,7	0,7	0,7
grados por secuencia	6,5	6,5	6	5	6	6,5	4,5	17	20	19	20

tiempo	0,7	0,7	0,7	0,7	0,7	1	1	1	1	1	1
grados por secuencia	17	21	22	21	23	29	38	38	36	36	36

tiempo	1	1	1	2	2	2	2	2	2	2	2
grados por secuencia	36	34	34	80	85	78	87	80	92	90	71

tiempo	2	0,4	0,7	1	2
grados por secuencia	82	6	20	37	87

Tabla 2 Mediciones de ángulos de giro respecto al tiempo



Gráfica 2 Calibración rotación pura

Hemos tenido que recurrir a un polinomio de mayor grado para realizar el ajuste, pero como se puede observar también se puede observar que son correctos debido a su regresión.

7 LOCALIZACIÓN

En el presente proyecto se busca hacer un mapeado de un espacio interior, el principal problema de este es la localización en el área que se quiere estudiar y de esta manera poder conocer la posición y la orientación en cada instante de tiempo. Como ya se ha dicho anteriormente el método más común y más sencillo de aplicar es el odométrico, donde se integra la trayectoria para estimar la posición y orientación del robot por el movimiento de las ruedas motrices. Su principal inconveniente es el error acumulativo, también llamado deriva, debido a errores intrínsecos del modelo que hacen que tras una serie de acciones la posición sea totalmente impredecible. Por este motivo es por el cual es necesario complementar este método con diferentes procesos que actualicen la posición del robot para mantener el error acotado dentro de un rango a lo largo de toda la secuencia de acciones. Al proceso que se encarga de determinar la posición y orientación del robot móvil, utilizando la información procedente de los sensores externos, se le denomina relocalización.

Los métodos desarrollados para la resolución de esta problemática se pueden agrupar en dos grupos, la detección de marcas, tanto naturales como artificiales, presentes en el entorno, y la correspondencia entre la información suministrada por los sensores y un mapa a priori del entorno.

Las características que se deberán tener en cuenta a la hora de diseñar el sistema sensorial vendrán dadas por: tamaño, consumo, simplicidad, redundancia, capacidad de operar en tiempo real, capacidad para detectar todo tipo de objetos en el entorno, resolución, exactitud, máxima y mínima distancia efectiva, y campo de visión. Los sensores más utilizados son los siguientes:

- Sensores de proximidad: ultrasonidos, infrarrojos y láseres.
- Sistemas basados en visión artificial: monoculares, estéreos o luz estructurada.
- Sistemas de posicionamiento global.
- Sistemas de navegación inercial.

En este trabajo de fin de grado utilizaremos el reconocimiento de marcas para resolver la problemática de conocer nuestra posición. Como ya se ha dicho anteriormente se pueden clasificar en dos grupos, marcas naturales como artificiales.

7.1 Balizas naturales

El principal problema del uso de las marcas naturales o del entorno, es poder detectar y extraer información de objetos que se encuentran en el entorno a través del sistema sensorial del vehículo. Este es el motivo principal por la cual en la mayoría de los casos se emplean sistemas de visión artificial, que permiten por ejemplo la detección de los bordes verticales del entorno, asociados a puertas, ventanas, esquinas, así como las luces del techo, etc. Presentan la ventaja de no tener que introducir modificaciones en el entorno de trabajo. El estudio de este problema es el problema de SLAM (Simultaneous Localization and Mapping), este es un problema de gran relevancia en la robótica, en el cual se investiga actualmente y existen diferentes aproximaciones para resolverlo.

7.1.1 SLAM (Simultaneous Localization and Mapping)

El SLAM o Simultaneous Localization and Mapping ha sido uno de los éxitos más importantes de la comunidad robótica para la solución a la problemática dada por la navegación. Esta técnica consiste en establecer bases estadísticas para describir las relaciones entre los objetos de referencia (conocidos como landmarks) y la manipulación de la incertidumbre geométrica. Un elemento clave de estos trabajos fue mostrar que debía existir una gran correlación entre las estimaciones de la localización de diferentes landmarks, y que esta estimación debía crecer con las sucesivas observaciones.

Con los primeros avances de la navegación visual y en la navegación por sónar, usando algoritmos basados en filtros de Kalman. Estas dos ramas separadas de investigación combinadas ayudaron en la investigación de la navegación de robots móviles basada en landmarks.

Estas investigaciones demostraron que para un robot móvil moviéndose a través de un entorno desconocido y capturando observaciones relativas de puntos de referencia, las estimaciones de éstas están necesariamente correlacionadas entre sí a causa del error común en la estimación de la posición del vehículo. La implicación de esto era profunda: una solución consistente al problema combinado de la localización y el mapeado requeriría un estado conjunto compuesto por la posición de cada uno de los objetos de referencia, que debería ser actualizado con cada observación de unos de estos objetos. Esto requeriría que el estimador de posición almacenara un gran vector de estados (de orden igual al número de landmarks que se conserven del mapa), con un coste computacional escalado al doble del cuadrado de landmarks.

En primera instancia los estudios no fueron fructíferos los estudios sobre este tipo de práctica por errores de convergencia en las estimaciones. Así, dada la complejidad computacional del problema del mapeado y con un completo desconocimiento del comportamiento convergente del mapa, los investigadores se centraron en una serie de aproximaciones a la solución al problema del mapeado consistente que asumía, e incluso forzaba, la minimización de las correlaciones entre los objetos de referencia, reduciendo así el filtro completo en una serie de filtros vehículo-referencia sin relación entre sí.

El descubrimiento más importante fue que el problema del mapeado y la localización, una vez formulado como un único problema de estimación, era de naturaleza convergente. Aún más, se reconoció que las correlaciones entre los objetos de referencia, que otros científicos intentaban minimizar o eliminar, eran la pieza clave del problema y que, de hecho, cuanto más ricas fueran estas correlaciones, mejor sería la solución.

El proceso del SLAM consiste en un cierto número de pasos: extracción de características, asociación de datos, estimación del estado y actualización de las características. La finalidad del proceso es usar el entorno para actualizar la posición del robot. Dado que la odometría del robot no es enteramente fiable, no podemos depender directamente de ella. Debemos usar sensores de distancia para corregir esta posición.

Esto se consigue extrayendo características del entorno y observándolas mientras el robot se encuentre en movimiento, de esta manera captamos la variación del punto de referencia respecto al movimiento. Un filtro extendido de Kalman es responsable de actualizar la estimación de la posición del robot basándose en estas características, a las que hemos llamado puntos de referencia o landmarks.

Este método hoy en día tiene muchas limitaciones y es necesaria un gran poder computacional para poder llevarlo a cabo. Por estos motivos no es posible la implantación de este sistema en nuestro proyecto.

7.2 Balizas artificiales

La detección de las balizas artificiales simplifica el problema, pero requieren de una actuación sobre el entorno, como pegar los marcadores y localizarlos en el entorno. Al ser conocido su tamaño y forma, permitiendo poder determinar la posición del robot a través de la relación posicional existente entre la cámara y la marca, por la proyección bidimensional de ésta última sobre el plano de imagen. Los marcadores artificiales más importantes en este campo son los April Tags debido a su sencillez y a la información proporciona por estés.

7.2.1 April Tags Fiducial System

Este sistema utiliza una serie de marcadores artificiales desarrollados en la Universidad de Michigan que son utilizados en una gran cantidad de tareas, realidad aumentada, robótica o calibración de cámaras, entre muchas otras. Estos Tags son unos patrones de cuadrados blancos y negros que pueden estar impresos en un folio. Este sistema es tan extendido ya que las librerías de detección están puestas a disposición pública (implementables en los leguajes Java y C). Estos marcadores nos permiten obtener la posición 3D, la orientación y el identificador del Tag respecto a la cámara.

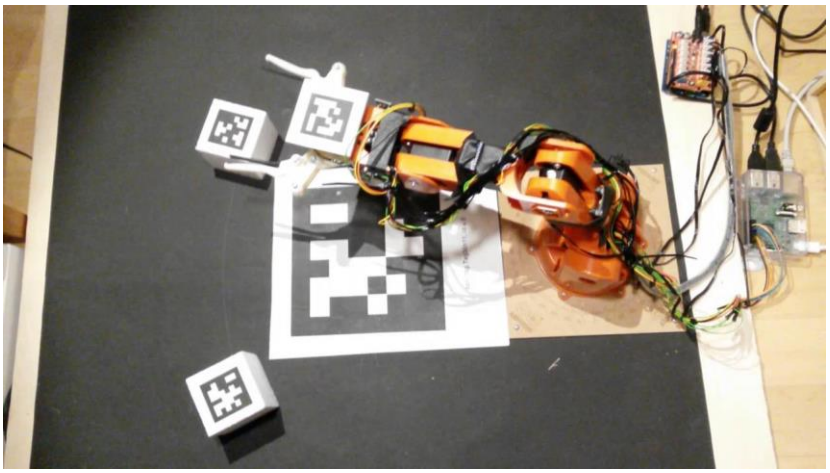


Figura 11 Robot guiado por April Tags

Los sistemas basados en la localización de objetos referencia son los más interesantes en el ámbito de la navegación ya que se basan en detectar características del entorno para la orientación y la localización del robot, pero este sistema tiene una serie de necesidades para su correcto funcionamiento, como una enorme capacidad de computación y una gran precisión de sus sensores. Por este motivo la aplicación de marcadores gana interés, ya que su disponibilidad es muy amplia y con una sencilla aplicación.

Existen también otro tipo de Tags, como por ejemplo los QR, este tipo de tags son más utilizados en el mundo de los smartphones debido a que pueden contener información de forma de cadena de texto, como por ejemplo una página web, a pesar de que en nuestro caso no sería preciso. Los April Tags tienen la ventaja en el objetivo de la navegación de que no es necesaria una gran resolución para su detección ya que están compuestos por una menos cantidad de pixeles, entre 49 y 100, frente a los 268 de los QR. También un hecho diferencial entre los April Tags y los QR es el hecho de que los April Tags pueden ser detectados de manera múltiple en una sola imagen, al contrario de los QR.

7.3 Elección del sistema de orientación

En este proyecto se opta por un sistema de navegación basado en la localización de QR por los motivos que se van a exponer a continuación. Se ha descartado el sistema de navegación SLAM (Simultaneous Localization and Mapping) debido a la complejidad del mismo frente a las ventajas que nos ofrece en nuestro caso, aparte de los requerimientos del sistema para que este funcione de forma correcta. Por estos motivos se ha optado por la detección de balizas artificiales para la elaboración de nuestro sistema de navegación. Dentro de estos se han optado por las etiquetas QR ya que estas librerías de detección ya estaban implementadas en Robobo, a pesar de que los ideales en nuestro caso serían los April Tags. De todas formas, se ha implementado un algoritmo en el que nos da la información que nos resulta de utilidad para dicha navegación como la posición respecto al QR y el ángulo de orientación del Robobo frente al QR.

7.4 Etiquetas QR

Las etiquetas QR se caracterizan por tener cuadrados concéntricos en 3 de sus esquinas como parámetro de identificación de la etiqueta, estos cuadrados concéntricos siempre son del mismo tamaño, y como ya se ha dicho, sirven para reconocer a este tipo de etiquetas. En el contorno que forman estos 3 cuadrados se encuentran 268 pixeles de color blanco o negro, que contienen algún tipo de información.

En nuestro caso nos centraremos en la visualización de los 3 cuadrados concéntricos de las esquinas para obtener nuestra posición relativa.

Figura 12 Etiqueta QR

7.5 Posición relativa respecto al QR

En cuanto a la colocación de los QR se ha optado por situarlos en el techo de la sala que se esté monitorizando ya que esto nos permitirá por un lado hacer menos invasiva la modificación del entorno y por otro facilitar la visualización del mismo mediante la cámara de Robobo. En el setup experimental utilizado, el Robobo se desplazará en el suelo de la oficina y por tanto los QRs se encontrarán a 2.5 metros de distancia de la cámara. Para mejorar la visualización de los mismos los hemos imprimido en tamaño A3 lo cual junto con la calidad de las cámaras de los smartphones actuales nos han permitido detectarlo adecuadamente.

Una vez detectado el QR toca establecer nuestra posición relativa respecto a este, para ello se han de utilizar los parámetros dados por el comando de Robobo que lee el QR, *readQR().* Este comando nos devuelve la posición de los tres cuadrados concéntricos de las esquinas en nuestra imagen y la distancia de la diagonal del QR.

Según el esquema seguido podemos diferenciar cuatro sistemas de referencia. En primer lugar, el sistema de referencia global, S_g, de la sala que permanece fijo y es el que nos interesa utilizar. En segundo lugar, el sistema de referencia de cada QR, S_{QR_i} cuya orientación y posición son fijas respecto al sistema de referencia global.

A continuación, tenemos el sistema de referencia de la cámara del Robobo, S_{cam}, y que es el que va a definir las coordenadas que obtendremos del comando *readQR* y por último el sistema de referencia del Robobo, S_{ROB}, que está centrado en el punto medio entre sus dos ruedas y alineado con el eje principal del Robobo.

Nuestro objetivo es transformar obtener la posición y orientación de S_{ROB} en el sistema de referencia global. Para ello utilizamos el siguiente modelo.

En el sistema de referencia S_{cam}:

Vector distancia entre los puntos 2 y 1.

$$x_1 = qr.p1.x - qr.p2.x$$

$$y_1 = qr.p1.y - qr.p2.y$$

$$v_1 = (x_1, y_1)$$

Vector distancia entre los puntos 2 y 3.

$$x_2 = qr.p3.x - qr.p2.x$$

$$y_2 = qr.p3.y - qr.p2.y$$

$$v_2 = (x_2, y_2)$$

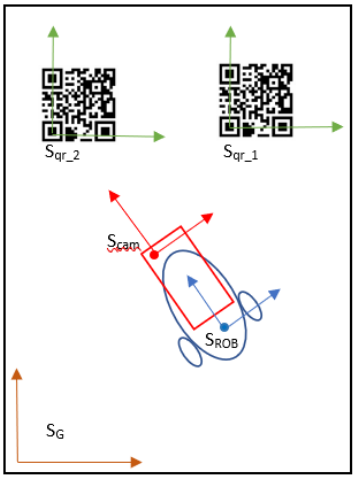


Figura 13 Visualización sistemas de referencia

Haremos una matriz cuya composición sea cada vector anterior una columna de esta.

$$A = [v_1 : v_2]$$

Esta matriz A va a representar la base del sistema de referencia de un QR, S_{qr_i} en el sistema de referencia de la cámara. Y la inversa de la matriz A nos permitiría expresar cualquier vector en S_{cam} en función del sistema de referencia del QR. Para poder expresar la ubicación del Robobo en S_{QR} sólo nos faltaría conocer la posición del centro del Robobo en S_{cam} y formar un vector entre esta y el origen del QR. Para hacer esto se

ha colocado el Robobo en diferentes posiciones y se han obtenido datos haciendo girar al Robobo sobre su centro. Todas estas posiciones deben transformarse las mismas coordenadas en el sistema de referencia del QR, a las cuales hemos denominado [C_1,C_2].

Vector distancia entre el centro de la imagen y el punto 2.

$$x_3 = x_{centro} - qr.p2.x$$

$$y_3 = y_{centro} - qr.p2.y$$

$$V_{origen} = \begin{bmatrix} x_3 \\ y_3 \end{bmatrix}$$

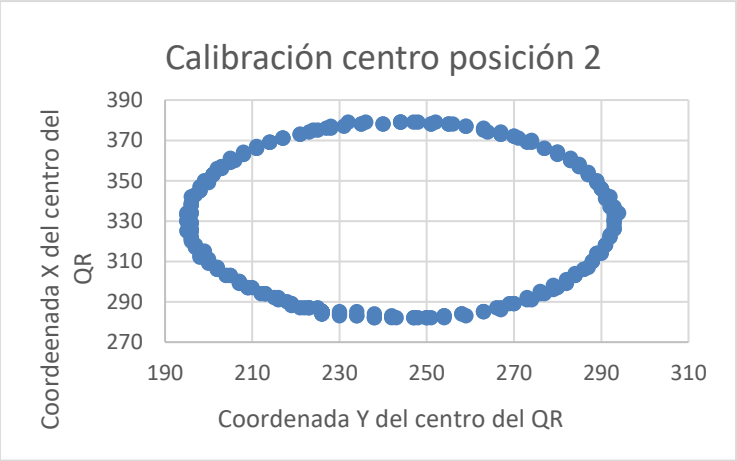
$$A^{-1} \cdot V_{origen} = \begin{pmatrix} C_1 \\ C_2 \end{pmatrix}$$

Esta calibración se ha realizado para distintas posiciones del Robobo para tener en cuenta las distorsiones ópticas de la lente de la cámara. Una vez obtenidos los datos y mediante el uso de un solver, se han obtenidos los valores de las coordenadas del centro (x_{centro}, y_{centro}) que minimizan la varianza de las coordenadas C1 y C2, pues deberían mantenerse fijas en cada maniobra de giro. Los datos obtenidos nos dan las siguientes gráficas con sus respectivas varianzas:

Gráfica 3 Calibración centro posición 1

Varianza C1	0,52097336
Varianza C2	1,04432854
Varianza cuadrática media	0,82523795

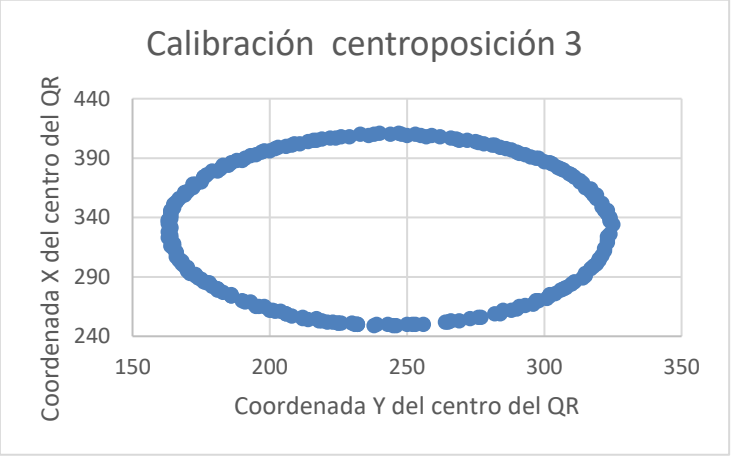
Tabla 3 Varianzas posición 1



Gráfica 4 Calibración centro posición 2

Varianza C1	1,00433743
Varianza C2	0,29439957
Varianza cuadrática media	0,74005566

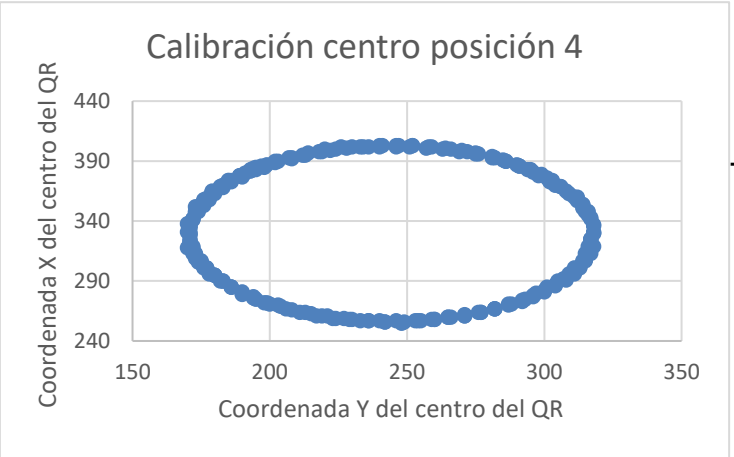
Tabla 4 Varianzas posición 2



Gráfica 5 Calibración centro posición 3

Varianza C1	3,18733534
Varianza C2	2,42893063
Varianza cuadrática media	2,83362053

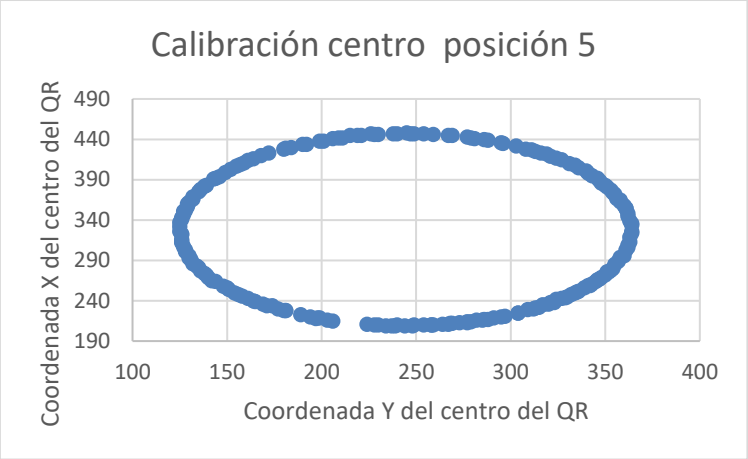
Tabla 5 Varianzas posición 3



Gráfica 6 Calibración centro posición 4

Varianza C1	1,39077257
Varianza C2	0,69099376
Varianza cuadrática media	1,09811673

Tabla 6 Varianzas posición 4



Gráfica 7 Calibración centro posición 5

Varianza C1	1,39077257
Varianza C2	0,69099376
Varianza cuadrática media	1,09811673

Tabla 7 Varianzas posición 5

Podemos observar que los datos obtenidos son de gran concordancia gracia a la varianza, ya que es baja, con una varianza total de los datos obtenidos de 3,4237. Gráficamente se puede observar que los puntos obtenidos del experimento forman un óvalo casi perfecto, esto nos quiere decir que no hubo deslizamientos y que el centro de giro se ha mantenido siempre en el mismo sitio. De los datos obtenidos se establece que nuestro centro de la imagen es 244,4 en el eje “x” y 330,3 en el eje “y”.

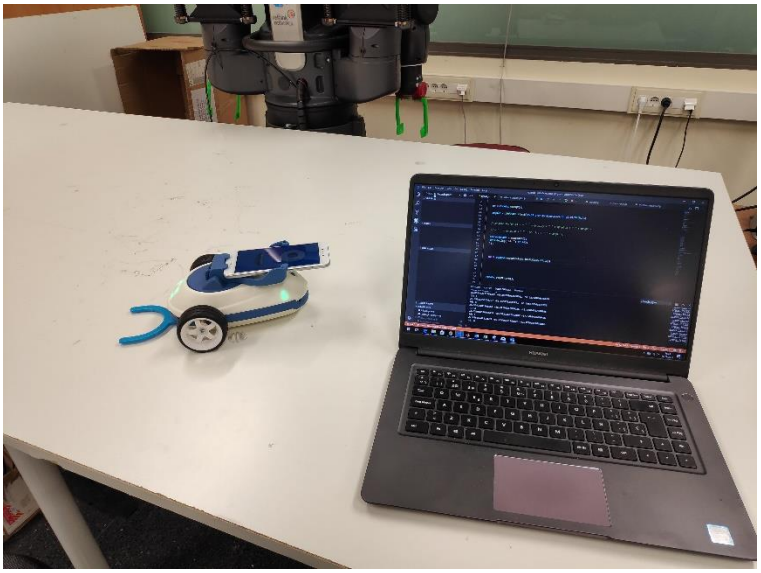


Figura 14 Calibración centro eje

7.6 Ángulo de orientación respecto al QR

El ángulo de orientación respecto al QR conociendo los datos anteriores se puede calcular con los diferentes componentes de cualquier vector, v_1 o v_2, de esta forma haciendo la arcotangente de alguno de los dos vectores con la componente "x" y la componente "y". De esta manera aplicando las siguientes ecuaciones obtendremos el ángulo de orientación respecto al QR.

$$arctg\left(\frac{y_1}{x_1}\right) \quad \text{o} \quad arctg\left(\frac{y_2}{x_2}\right)$$

7.7 Factor de correlación coordenadas a centímetros

Estas coordenadas obtenidas por los cálculos anteriores no están en centímetros, factor que nos causará un problema ya que las calibraciones de los distintos movimientos del Robobo se han hecho utilizando esta unidad de medida. Por lo tanto, tendremos que buscar un factor de correlación para pasar las unidades de las coordenadas que nos dan las ecuaciones anteriores a centímetros.

Este factor de correlación se encontrará simplemente haciendo las mediciones de ciertas distancias en diferentes ejes y las compararemos con los valores de las coordenadas dadas. Los resultados de esta prueba son los siguientes:

Coordenadas	Distancia real	Coeficiente
21,165	10	2,1165
44,8	20	2,24
65,82	30	2,194
11,8	5	2,36

Tabla 8 Relación de distancias y coordenadas

Seguidamente haremos una media de los distintos coeficientes y esta será de 2,256. Por lo tanto, los valores de nuestra posición obtenidos por los QR los tendremos que dividir por este coeficiente para hallar nuestra posición en centímetros. De esta manera ya podremos utilizar los datos de posición para el cálculo de los diferentes movimientos que queramos realizar.

8 NAVEGACIÓN

La navegación es la metodología que permite guiar a un determinado vehículo a través de un entorno. Las tareas involucradas en la navegación de un robot móvil son principalmente la percepción del entorno a través de sus sensores, de este modo el robot debe de tener una visión lo suficientemente clara para la interacción del mismo para así realizar la planificación de una trayectoria libre de obstáculos, para alcanzar el punto destino seleccionado. El guiado del vehículo a través de la referencia construida. De forma simultánea, el vehículo puede interaccionar con ciertos elementos del entorno. Esta es la definición del concepto de operación como la programación de las herramientas de a bordo que le permiten realizar la tarea especificada.

8.1 Esquemas de navegación en robots móviles

Realizar una tarea de navegación para un robot móvil significa recorrer un camino que lo conduzca desde una posición inicial hasta otra final, pasando por ciertas posiciones intermedias o submetas. El problema de la navegación se divide en las siguientes cuatro etapas:

- Percepción del entorno
  Observar el entorno de navegación para de esta manera observar los obstáculos y sortearlos.
- Planificación de la ruta
  Crea una secuencia de posiciones a las que debemos ir para así sortear los diferentes obstáculos del área en el que se mueve.
- Seguimiento del camino:
  Efectúa el desplazamiento del vehículo, según el camino generado mediante el adecuado control de los actuadores del vehículo.

8.2 Movimiento entre dos puntos

El movimiento entre dos puntos, una posición final y otra final, es la función principal de la navegación, ya que como se ha dicho anteriormente el objetivo fundamental de la navegación es establecer una secuencia de posiciones a la que debemos de ir en un determinado orden para evitar los diferentes obstáculos del entorno.

El funcionamiento de esta función es la siguiente. Nos encontramos en una posición conocida en un sistema de referencia global y nos indican a la posición a la que debemos ir en el mismo sistema de referencia. En primer lugar, tendremos que hacer un vector que vaya desde la posición inicial hasta la final. Una vez conocido este vector podremos

conocer el ángulo que forma este vector respecto a un eje de coordenadas global conocido este podremos hallar el giro a realizar para tener el sentido de este vector que será la trayectoria del movimiento con las calibraciones hechas anteriormente en este proyecto. Una vez estemos en la dirección de la trayectoria que debemos seguir solo hará falta realizar un movimiento recto hasta el punto final.

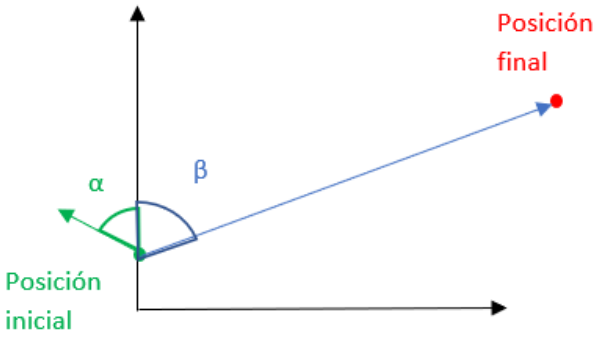


Figura 13 Desplazamiento entre dos puntos

Por último, cabe citar que deberemos referirnos a todos los parámetros en un sistema de coordenadas global.

8.3 Navegación absoluta con balizas QR

Este algoritmo tiene como base de funcionamiento la función de moverse entre dos puntos. En este programa daremos por hecho que el QR siempre estará dentro de nuestro campo de visión, lo que no hará falta el uso de la odometría, también cabe decir que entre dos puntos consecutivos dados no puede haber ningún obstáculo. Una vez aclarado las condiciones para su correcto funcionamiento el algoritmo hará lo siguiente:

Una vez dados los puntos que se tienen que recorrer por orden, Robobo observa en qué posición se encuentra mediante la visualización del QR que se encuentra en el techo, seguidamente calcula a que numero de posición ha de ir, esto se averiguará por medio de un contador que se inicializará en cero al inicio del programa y dividiendo este contador entre el número de posiciones diferentes por las que habrá que pasar y quedándonos con el resto de dicha división. Nos movemos a la posición resultante con la función de movimiento entre dos puntos y una vez llegamos al destino actualizamos nuestra posición e incrementamos el contador de las posiciones recorridas en una unidad. Este proceso lo repetiremos las veces que sea necesaria.

El pseudocódigo de este programa es el siguiente:

Algorithm 1: Mixed navigation with QR beacons.

```
initialise robot;
QR = false;
positions counter = 0;
if !QR then
    coord = readQR;
    QR = true;
end
for ever do
    if QR then
        destination = rest(positions counter / number positions);
        move(coord, points navigation [destination +1], robobo);
        start wait time;
        positions counter++;
        QR=false;
    end
end
```

Figura 14 Pseudocódigo navegación absoluta con balizas QR

8.4 Navegación mixta con balizas QR

Este algoritmo realiza la misma función que el anterior, pero con la salvedad de que este programa sí utiliza la odometría para estimar su posición en caso de no visualizar algún QR. De esta forma su funcionamiento es el que se define a continuación.

Conociendo nuestra posición relativa respecto al QR podemos desarrollar nuestro algoritmo de navegación, este recibirá como parámetros de entrada una matriz con los diferentes puntos que debe recorrer con su posición “x” e “y” respectivamente.

El funcionamiento del mismo será el siguiente, miramos si se detectó alguna vez una etiqueta QR. De no ser ese el caso, el programa no se iniciará hasta que se detecte por primera vez un QR, una vez visto por primera vez un QR guardamos nuestra posición dada por el mismo y no actualizaremos esta posición hasta que termine el movimiento por problemas de convergencia. Nos movemos al destino que corresponda, este se calculara dividiendo el número de posiciones que llevamos recorridas entre el número de puntos que hay que recorrer, calculando los movimientos a realizar por medio de trigonometría y con los datos obtenidos de la calibración de los movimientos. Seguidamente calculamos el ángulo final en el que terminaremos por si en la posición final no conseguimos visualizar ningún QR, incrementamos el número de posiciones recorridas en una unidad e inicializamos un contador de tiempo. El contador de tiempo tendrá la función de que si durante un determinado periodo de tiempo no consigue visualizar ningún QR haga los movimientos necesarios para llegar a la siguiente posición considerando que su ubicación es la del destino exacto del movimiento anterior. Este proceso se repetirá el número de veces que se desee. El algoritmo completo se encontrará en el anexo.

A continuación, se muestra un pseudocódigo del algoritmo:

Algorithm 1: Mixed navigation with QR beacons.

```
initialise robot;
QR = false;
AnyQR = false;
positions counter = 0;
if !QR then
    coord = readQR;
    QR = true;
    AnyQR = true;
end
for ever do
    if QR then
        destination = rest(positions counter / number positions);
        move(coord, points navigation [destination +1], robobo);
        alfa estimated = calcturn (points navigation [destination],points navigation [destination +1])[1];
        start wait time;
        positions counter++;
        QR=false;
    end
    if !QR and time - start wait time then
        QR = true;
        destination = rest(positions counter / number positions);
        start position estimated = [points navigation[destination],alfa estimated];
        move(start position estimates, points navigation [destination +1], robobo);
        alfa estimated = calcturn (points navigation [destination],points navigation [destination +1])[1];
        start wait time;
        positions counter++;
        QR=false;
    end
end
```

Figura 15 Pseudocódigo navegación mixta

8.5 Algoritmo de evasión de obstáculos

Una vez desarrollado el algoritmo para la navegación entre unas determinadas posiciones hemos realizado un algoritmo que nos permite esquivar obstáculos poligonales. De manera que, pasándole los puntos de las esquinas de cualquier obstáculo, nuestra posición final y el destino a cuál queremos ir en un sistema de coordenadas global todos los parámetros citados será capaz de evadir el obstáculo en cuestión.

El funcionamiento de dicho algoritmo será el próximo, como ya se ha dicho anteriormente los parámetros que le pasamos a dicho algoritmo son los de las esquinas del objeto a evadir mediante una matriz que contenga la coordenada “x” y la coordenada “y” de cada esquina, y por otra parte nuestra posición inicial y el destino al que queremos ir.

En primera instancia se calcula cual será la mejor manera de rodear el obstáculo, esto se calcula hallando la longitud de la trayectoria por la derecha y por la izquierda del objeto, y consecuentemente se cogerá el criterio que nos dé la menor distancia. Esta longitud de la trayectoria se calcula con la siguiente secuencia de operaciones, mientras la distancia entre nuestra posición inicial y la final sea menor de un umbral, como por ejemplo 5 centímetros, calculamos el punto de corte más cercano entre las rectas del obstáculo y la recta que irá desde nuestra posición inicial y el destino. Si existen puntos de corte calcularemos a la posición que debemos ir, que esta será la esquina de la izquierda o derecha, según el criterio establecido, de la recta con la que existe el punto de corte más cercano con un umbral de seguridad para que no colisione con el objeto, como se observa en la figura Z. En caso de que no exista punto de corte con el objeto, este no se encontrará entre el destino y nuestra posición. De esta manera se ira contabilizando la distancia recorrida.

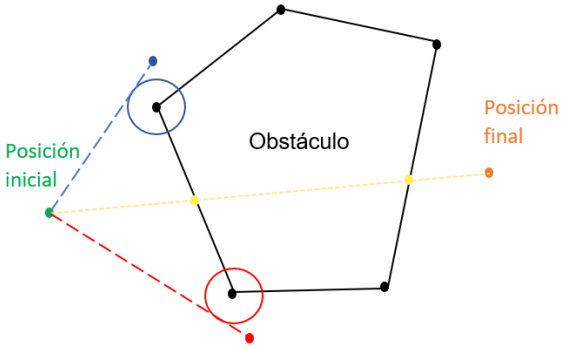


Figura 16 Algoritmo de evasión de obstáculos

La manera de calcular el punto de corte más cercano entre nuestra posición y el objeto será la siguiente, primero calcularemos todas las rectas que conforman entre las esquinas contiguas del objeto, seguidamente se hallará la recta que irá entre la posición final y la de destino, y calcularemos los puntos de corte con dichas rectas, miraremos si estos puntos de corte se encontrarán dentro del objeto y cogeremos el punto de corte que haya menos distancia respecto nuestra posición inicial.

Una vez calculado el mejor criterio para rodear el obstáculo haremos el proceso anterior para calcular los puntos a los que debemos ir para sortear el objeto y moviéndonos al mismo tiempo. Este proceso se hará intentando actualizar nuestra posición por medio de las etiquetas QR.

El pseudocódigo de este algoritmo será el siguiente:

Algorithm 1: function main dodge obstacs

```
initialise robot;
corner objects;
posi;
posf;
crit = calccrit(corner objects, posi, posf);
move(corner objets, posi, posf, crit, robobo)
```

Algorithm 1: Function calccrit

```
corner objects;
posi;
posf;
Lpos=longtray(corner objects, posi, posf, alfa, 1);
Lneg=longtray(corner objects, posi, posf, alfa, -1);
if Lpos ¡ Lneg then
    return -1;
end
return 1;
```

Algorithm 1: Function calccrit

```
corner objects;
posi;
posf;
alfa;
crit;
if dist(posi,posf) ¿ 5 then
    ptocort=ptocortnear(corner obstacs,posi,posf);
    if ptocort==null then
        dist=dist(posi,posf)+dist;
    end
    nextstep=calcnextstep(corner obstacs, posi,ptocort,alfa, crit);
    dist=dist(posi,nextstep)+dist;
    posi=nextstep;
end
return dist;
```

Algorithm 1: Function ptocortnear

```
corner objects;
posi;
posf;
rectobstac(corner obstacs,posi,posf);
rectoi(posi,posf);
ptoscort(rectoi, recobstac);
if ptoscort == null then
    return null;
end
ptocortnear=chooseptocort(ptocort,posi);
return ptocortnear;
```

Algorithm 1: Function move

```
initialise robot;
corner objects;
posi;
posf;
if true then
    if dist(posi,posf) then
        ptocort = ptocortnear(posi, posf, corner obstacs);
        if ptocort==null then
            moveto(posi,posf);
        end
        nextstep=calcnextstep(corner objects, posi, posf, alfa, crit);
        moveto(posi,nextstep);
    end
end
```

Figura 17 Pseudocódigo salvar obstáculos

Este algoritmo es de gran importancia ya que es el que nos servirá para la coordinación de varios robots. De esta manera podemos traducir a los diferentes robots en obstáculos poligonales de unas determinadas dimensiones móviles en el espacio, de esta manera cuando se presente un robot en la trayectoria de otro el primero lo sorteará sin problemas y sucesivamente seguirá con su camino.

El código completo de este algoritmo se encuentra en el anexo.

9 MAPEADO

El mapeado es la representación cartográfica de determinados factores en una zona específica. Este mapa es de gran utilidad porque determina el grado de cualquier aspecto al que está sometido un espacio concreto. A partir de este mapeado se pueden determinan las zonas críticas de cualquier factor y se podrán realizar medidas para actuar sobre dichas zonas.

En el presente trabajo de fin de grado se va a diseñar el algoritmo para el mapeado de un área de trabajo midiendo los niveles de ruido y luminosidad del mismo. De esta manera estudiar las zonas críticas en las cuales no se pueda realizar la labor requerida en las condiciones óptimas.

Este mapeado se ha realizado combinando los programas de navegación mixta y el algoritmo de evasión de obstáculos para su correcta navegación en el centro de trabajo, y los comandos de Robobo “readBrightnessSensor();” y “readNoiseLevel()” para contabilizar los niveles de luminosidad y ruido respectivamente. Estas lecturas se realizarán en parado en los distintos puntos indicados para realizar la navegación, principalmente no tener una cantidad inmensa de datos y también de esta manera no distorsionamos la medida de luminosidad. Los datos serán guardados en una matriz en el cual almacenará en cada fila de la misma la posición “x” e “y” del Robobo, el ángulo de orientación de este, si se ha detectado algún QR en la posición de medida, y los valores de intensidad lumínica y ruido respectivamente. El programa completo se encontrará en el anexo.

10 CASO DE APLICACIÓN: MONITORIZACIÓN DE NIVELES DE CONTAMINANTES

10.1 Riesgos laborales en el centro de trabajo

Los riesgos laborales son los peligros existentes asociados a una profesión o tarea profesional, al entorno en el cual se desarrolla dicha actividad o lugar de trabajo, que puedan ocasionar accidentes, daños o algún riesgo para la salud tanto físico como psicológico al individuo. La principal vía para evitar los riesgos laborales es a través de la prevención de estos, para ello se implementa un plan de Gestión y Seguridad en el trabajo.

10.1.1 La prevención de los riesgos laborales (PRL)

La Prevención de Riesgos Laborales (PRL) es la disciplina que busca promover la seguridad y salud de los trabajadores mediante la identificación, evaluación y control de los peligros y riesgos asociados a un entorno laboral, además de fomentar el desarrollo de actividades y medidas necesarias para prevenir los riesgos derivados del trabajo. Se trata también de un conjunto de técnicas orientadas a reconocer, evaluar y controlar los riesgos ambientales que pueden ocasionar accidentes y/o enfermedades profesionales.

El accidente laboral es aquel que se produce, por fallo humano o de otra índole, durante el trascurso de la jornada laboral de un trabajador. También son considerados accidentes laborales aquellos accidentes que tienen lugar en el trayecto del trabajador al puesto de trabajo o en el trayecto de vuelta del trabajo a casa (accidentes in itinere).

La legislación se basa en “el derecho de los trabajadores a un trabajo en condiciones de seguridad y salud, lo que conlleva el deber del empresario para conseguir esa protección”.

Una vez realizado el estudio sobre los diversos riesgos laborales que pueden afectar a los trabajadores, se actúa sobre el problema desarrollando una serie de medidas que pueden ser de diversos tipos:

- Información y concienciación de los trabajadores.
- Dotación de Equipos de Protección Individual (EPI).
- Mejora de infraestructuras y entornos de trabajo.
- Medidas para evitar el estrés o el acoso laboral.

10.1.2 Factores de riesgo

Los factores de riesgo tienen una relación o dependencia directa de las condiciones de seguridad. Éstas siempre tendrán su origen en alguno de los cuatro aspectos del trabajo siguientes:

- centro de trabajo.

- Organización de trabajo.
- Tipo de actividad.
- Materias primas.

En el presente proyecto nos vamos a centrar en el estudio de las condiciones del centro de trabajo por lo tanto seguidamente vamos a definirlo y las condiciones que se deben encontrar en el mismo.

10.1.2.1 Centro de trabajo

La definición de centro de trabajo es la siguiente, "*Área, edificada o no, a la que los trabajadores deban permanecer o a la que deban acceder por razón de su trabajo. Se consideran incluidos en esta definición los servicios higiénicos y locales de descanso, los locales de primeros auxilios y los comedores.".* los requisitos que vamos a citar a continuación quedaran exentos de aplicación en:

- Los medios de transporte utilizados fuera de la empresa o centro de trabajo, así como a los lugares de trabajo situados dentro de los medios de transporte.
- Las obras de construcción temporales o móviles.
- Las industrias de extracción.
- Los buques de pesca
- Los campos de cultivo, bosques y otros terrenos que formen parte de una empresa o centro de trabajo agrícola o forestal pero que estén situados fuera de la zona edificada de los mismos.

En el REAL DECRETO 486/1997 afectado por el REAL DECRETO 2177/2004 "se establecen las disposiciones mínimas de seguridad y salud para la utilización por los trabajadores de los equipos de trabajo, en materia de trabajos temporales en altura.". De tal forma, en el centro de trabajo se deben disponer unas condiciones adecuadas para la realización de la actividad con unas condiciones de confort mínimas, en este trabajo de fin de grado se estudiarán los niveles de luz y de ruido en el espacio de trabajo, por lo consiguiente desglosaremos en qué consisten estos factores.

10.1.2.1.1 Obligación general del empresario

El empresario deberá adoptar las medidas necesarias para que la utilización de los lugares de trabajo no origine riesgos para la seguridad y salud de los trabajadores o, si ello no fuera posible, para que tales riesgos se reduzcan al mínimo. En cualquier caso, los lugares de trabajo deberán cumplir las disposiciones mínimas establecidas en el presente Real Decreto en cuanto a sus condiciones constructivas, orden, limpieza y mantenimiento, señalización, instalaciones de servicio o protección, condiciones ambientales, iluminación, servicios higiénicos y locales de descanso, y material y locales de primeros auxilios.

10.1.3 Iluminación

La iluminación es una parte fundamental en el acondicionamiento ergonómico del centro de trabajo. A pesar de la gran capacidad del ser humano para la adaptación a los diferentes niveles de luminosidad, una mala iluminación en el puesto de trabajo puede producir fatiga visual, un peor rendimiento y en algunas ocasiones hasta accidentes. Un adecuado análisis de las características que deben disponer los sistemas de

iluminación, la adaptación a las tareas a realizar y las características individuales, son aspectos fundamentales que se deben considerar.

10.1.3.1 Nivel de iluminación

También conocido como iluminancia. Es el cociente del flujo luminoso incidente sobre un elemento de la superficie que contiene el punto por el área de ese elemento. Su unidad es el lux.

$$E = \Phi / S$$

E: nivel de iluminación expresado en luxes.

Φ: flujo luminoso incidente en una superficie expresado en lúmenes.

S: superficie expresada en m^2.

10.1.3.2 Luminancia

La luminancia de una superficie viene determinada por el flujo luminoso incidente y por el flujo luminoso reflejado. Ambos flujos están relacionados mediante un factor de reflexión característico del material de la superficie. En definitiva, es la magnitud que mide la claridad o el brillo con que vemos los objetos iluminados. Su unidad es la candela/m^2.

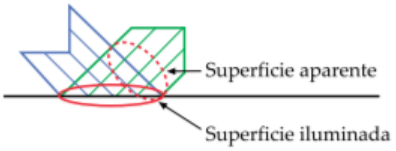
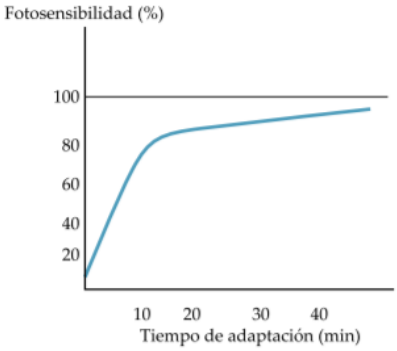


Figura 18 Refracción de luminancia sobre una superficie

10.1.3.3 La percepción de luminancias y su efecto

El ojo humano tiene la facultad de adaptarse a muy distintos niveles de luminancia. El nivel al que se encuentra adaptado el ojo en un momento dado se conoce como “luminancia de adaptación”. El ojo necesita tiempo para adaptarse a un cambio en el nivel de luminancia. Durante ese intervalo de tiempo un trabajador puede quedar “cegado” o su capacidad de visión notablemente disminuida. Por esta razón, se recomienda introducir periodos de adaptación antes de comenzar a trabajar en ambientes poco iluminados después de haber permanecido en otro muy iluminado. En cada caso, las curvas de adaptación permitirán estimar la duración de la adaptación necesaria para trabajar en lugares donde exista riesgo de accidente.



Gráfica 8 Fotosensibilidad respecto a tiempo

En condiciones normales, un aumento de luminancia conlleva una mejora del rendimiento visual esto sucede hasta un cierto umbral, a partir de ahí decrece.

10.1.3.4 Ecuación fundamental de la luminotecnia

El nivel de iluminación es directamente proporcional a la intensidad luminosa e inversamente proporcional al cuadrado de la distancia, este factor es multiplicado por el ángulo sobre el que incide el haz de luz en la superficie.

$$E = I/d^2 \cdot \cos\theta = I/h^2 \cdot \cos^3\theta$$

E: nivel de iluminación expresado en luxes.

I: intensidad luminosa expresada en candelas.

d: distancia en metros.

θ: ángulo formado por la superficie en la que se quiere medir y el plano perpendicular.

h: altura del foco luminoso.

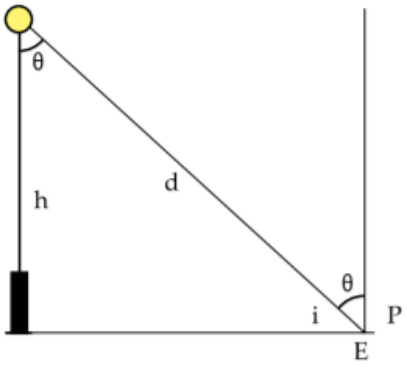


Figura 19 Incidencia de luz sobre una superficie

10.1.3.5 Percepción del color

Para una buena percepción del color es necesario tener en cuenta los siguientes factores:

- El nivel de luminancia de adaptación debe ser suficientemente elevado, para permitir la “visión fotópica” del ojo, responsable de la visión en color.
- Las lámparas utilizadas deben emitir la luz en un espectro continuo de luminosidad.
- La tonalidad de la luz no se debe alejar mucho de la tonalidad de la luz natural.

10.1.3.6 Iluminación natural e iluminación artificial

A la hora de diseñar un área de trabajo se deben tener en cuenta la iluminación artificial y la iluminación natural. La luz natural causa menor cansancio visual que la luz artificial, por este motivo en la actualidad se busca aprovechar al máximo este tipo de luz. Por el contrario, la principal desventaja de este tipo de luz es la variabilidad de esta, tanto por la variabilidad horaria como la estacional. Por lo tanto, la luz artificial se utiliza como complementación a la luz natural.

A la hora de evaluar o adecuar la iluminación en el puesto de trabajo se deben considerar aspectos relacionados con el trabajador y con el tipo de tarea que vaya a realizar. Las lámparas son aquellas que se van a encargar de generar la luz artificial, que estarán agrupadas en luminarias. Si las lámparas no emiten suficiente flujo luminoso, si hay zonas donde no se dispone de luminarias, si la luz no es la adecuada para la tarea que desempeña el trabajador, con tal de que uno de estos aspectos se cumpla se deberán de tomar medidas para la rectificación de la iluminación en el espacio de trabajo.

10.1.3.7 Criterios de iluminación interior

En el REAL DECRETO 486/1997 se establecen las condiciones mínimas de seguridad y salud en el trabajo. En el artículo 8 del mismo indica: “La iluminación de los lugares de trabajo deberá permitir que los trabajadores dispongan de condiciones de visibilidad adecuadas para poder circular por los mismos y desarrollar en ellos sus actividades sin riesgo para su seguridad y salud”. Donde en el anexo IV indica los niveles mínimos de luminosidad en el espacio de trabajo, que serán los siguientes:

Zona o parte del lugar de trabajo	Nivel de iluminación mínimo (Lux)
Zonas donde se ejecuten tareas con:	
1º-Bajas exigencias visuales	100
2º-Exigencias visuales moderadas	200
3º-Exigencias visuales altas	500
4º-Exigencias visuales muy altas	1000
Áreas o locales de uso ocasional	50
Áreas o locales de uso habitual	100
Vías de circulación de uso ocasional	25
Vías de circulación de uso habitual	50

Tabla 9 Niveles de iluminación mínimo en los diferentes espacios

La ley establece que “el nivel de iluminación de una zona en la que se ejecute una tarea se medirá a la altura donde esta se realice, en las zonas de uso general a 85 cm del suelo y en las vias de circulación a nivel del suelo.”

10.1.3.8 Medición de niveles de iluminación

El luxómetro es el instrumento que se utiliza para medir el nivel de iluminación. Este dispositivo está comprendido por una célula fotovoltaica, esta es la encargada de transformar la luz en electrones por medio del fenómeno fotoeléctrico de los materiales, sabiendo la capacidad fotoeléctrica de los materiales y por medio de un dispositivo capaz de medir la cantidad de corriente que circula es relativamente sencillo saber la intensidad lumínica que hay en el ambiente.

En este proyecto se utilizará el móvil para medir la iluminación del ambiente, los móviles para medir este parámetro suelen usar fotorresistencias, debido a su pequeño tamaño. este sensor los móviles lo suelen utilizar para ajustar el brillo de la pantalla o bien para saber cuándo es necesario el uso del flash.

10.1.4 Ruido

El ruido es uno de los agentes contaminantes más frecuentes en el espacio de trabajo, tanto en los del tipo industrial como de otro tipo. En los del tipo no industrial, como por ejemplo oficinas, no presenta un riesgo de tan alto nivel como la pérdida auditiva, pero si puede causar como alteraciones fisiológicas, distracciones, interferencias en la comunicación o alteraciones psicológicas.

En primera instancia deberemos analizar cuáles son las principales fuentes de ruido en el espacio de trabajo, para así actuar primero sobre estas fuentes de ruido ya que, si actuamos sobre las fuentes secundarias la mejora probablemente, no sería sustancial.

El segundo paso sería el de determinar por qué el ruido es molesto. Puede ser por exceso de niveles de presión sonora, en este caso bastaría con una medición continua del ruido equivalente. Pero en otras, será necesario conocer el espectro de la frecuencia del ruido. En la mayoría de los casos a parte de la medición de este, el estudio se deberá de complementar con el análisis de aspectos no físicos para determinar el grado de molestia que ocasiona.

10.1.4.1 Fuentes de ruido

El ruido proviene de varias fuentes y a través de varias vías. El ruido emitido por una fuente se propaga en todas las direcciones y puede llegar de forma directa al receptor o ser parcialmente absorbido, transmitido y/o reflejado por los obstáculos que se encuentra en su camino.

Por estos motivos, el nivel de presión sonora que existe en un recinto depende de las fuentes de ruido y de las características acústicas y geométricas del local.

La reflexión es el fenómeno en el cual una parte de la energía acústica de la onda sonora es absorbida con la colisión con una superficie, lo que disminuye la cantidad reflejada. Cada superficie tiene su correspondiente coeficiente de absorción del sonido (α). En la práctica, se toma un valor medio de a en función de la superficie y los materiales del local.

El nivel global de ruido en un espacio es la resultante del ruido que llega al receptor directamente desde las fuentes y el que llega después de haberse reflejado una o varias veces, a esta parte de la onda es denominada “reverberación”. Este fenómeno es inversamente proporcional al coeficiente de absorción.

Dependiendo de donde provenga el ruido se pueden tomar diversas medidas para atenuarlo. Las cuatro fuentes de ruido principales en el espacio de trabajo son: el exterior del edificio, el de las instalaciones del edificio, el de los equipos del espacio de trabajo y el producido por las personas.

10.1.4.2 Nivel sonoro continuo equivalente

Es el valor en decibelios (dBA) de una señal de ruido constante es el equivalente a la misma cantidad de energía sonora que el ruido real considerado durante un periodo de tiempo T. El nivel de presión sonoro se calcula con la siguiente expresión:

$$L_{Aeq} = 10 \cdot \log[1/T \cdot (\sum T_i \cdot 10^{L_i/10})]$$

Donde:

L_i: Nivel de presión sonoro (dBA) en el periodo “i”.

T_i: Duración del período “i”.

T: Período de tiempo total.

TIPO DE EDIFICIO	LOCAL	L_{Aeq} (dBA)
Residencial (público y privado)	**Zonas de estancia**	45
	Dormitorios	40
	Servicios	50
	Zonas comunes	50
Administrativo y de oficinas	**Despachos profesionales**	40
	Oficinas	45
	Zonas comunes	50
Sanitario	**Zonas de estancia**	45
	Dormitorios	30
	Zonas comunes	50
Docente	**Aulas**	40
	Salas de lectura	35
	Zonas comunes	35

Tabla 10 Niveles de ruido en las distintas estancias

Los niveles sonoros continuos equivalentes de ruido aéreo que no se deben sobrepasar en los locales son los siguientes.

10.1.4.3 Nivel sonoro diario equivalente

En el Real Decreto 1316/1989 se establece este índice, que será utilizado para la valoración de la exposición al ruido del trabajador. Que es de gran utilidad para valorar el riesgo de pérdida de capacidad auditiva, aunque esta no da información de otras características del ruido. De la cual su expresión es la siguiente:

$$L_{Aeq,d} = L_{Aeq,T} + 10 \cdot \frac{T}{8}$$

donde:

T: Duración diaria de la exposición (horas).

$L_{Aeq,T}$: Nivel de presión sonora equivalente en el período de tiempo T (dBA).

10.1.4.4 Índice de ruido en las oficinas

Este índice involucra el nivel de contaminación sonora e índice de ruido de tráfico, parámetros que se utilizan para la valoración del ruido de las fuentes exteriores. Estos índices aparte de para determinar el nivel de aislamiento acústico que debe tener el edificio son de utilidad para la valoración de los ruidos generados por las distintas fuentes. Para su determinación es necesario conocer el nivel de presión sonora y su fluctuación en un determinado periodo de tiempo.

Para la elaboración de un estudio del ruido en una oficina se ha de pedir información del ruido a los ocupantes. Se analizan los más insatisfactorios y se relacionan con los valores de las mediciones realizadas. El índice de ruido en las oficinas se establece con la siguiente ecuación:

$$IRO = L_{90} + 2,4 \cdot \left(L_{10} - L_{90} \right) - 14$$

En la cual:

L_{10}: El nivel de presión acústica (dBA) que se sobrepasa durante el 10% del tiempo de observación.

L_{90}: El nivel de presión acústica (dBA) que se sobrepasa durante el 90% del tiempo de observación.

Las mediciones se llevarán a cabo durante una jornada normal de trabajo y se corresponden al ruido total en las oficinas. Este índice es uno de los parámetros principales que se pueden evaluar con el presente proyecto.

10.1.4.5 Medición de niveles de ruido

Los sonómetros son dispositivos capaces de medir el ruido del ambiente compuestos principalmente de un micrófono y una serie de circuitos electrónicos. El micrófono contiene una membrana flexible que se mueve ligeramente cuando las ondas sonoras lo golpean. Debido a este movimiento la membrana crea corriente eléctrica. Cuanto más se mueve la membrana, más corriente eléctrica se crea. El sonómetro mide entonces la intensidad de la corriente eléctrica producida por la membrana y la convierte en un indicador numérico.

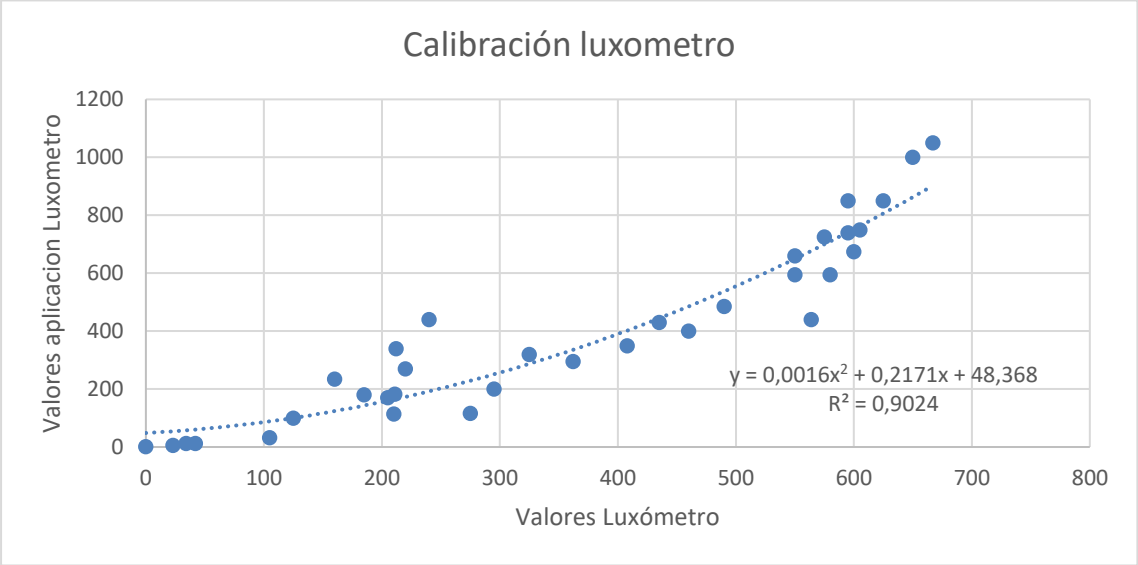
Figura 20 Sonómetro

10.2 Calibración de sensores de Robobo

10.2.1 Calibración del sensor de luminosidad

Los móviles de hoy en día utilizan unas fotorresistencias para saber cuál es la luz ambiental para así ajustar ciertos parámetros del móvil como por ejemplo el ajuste del brillo automático o la utilización del flash de manera automática.

Para la calibración del sensor de luminosidad del móvil se han hecho medidas con un luxómetro en diferentes ambientes de luminosidad, este valor lo consideraremos como valor exacto de luminosidad del ambiente, seguidamente sin modificar las condiciones en las que se ha hecho la medición anterior se ejecutará el comando de Robobo "readBrightnessSensor();" que mide la intensidad lumínica del entorno mediante la fotorresistencia citada anteriormente. De esta manera los datos obtenidos fueron los siguientes:



Gráfica 9 calibración luxómetro

Valores reales	605	550	460	575	435	205	42	105	220
Calibración aplicación luxómetro	750	660	400	725	430	170	12	32	270

Valores reales	160	185	23	34	212	125	595	625	490
Calibración aplicación luxómetro	235	180	6	12	340	100	850	850	485

Valores reales	0	667	650	600	564	580	550	595	408
Calibración aplicación luxómetro	2	1050	1000	675	440	595	595	740	350

valores reales	362	325	295	275	210	211	240
calibración aplicación luxómetro	295	320	200	116	114	183	440

Tabla 11 Datos de calibración sensor luz

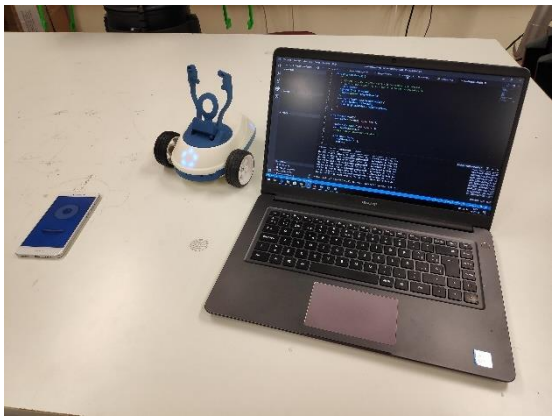


Figura 22 Calibración luxómetro

Figura 21 Calibración luxómetro

Como podemos observar los valores obtenidos presentan un cierto grado de desviación, a pesar de este factor los valores serán considerados correctos debido a que en las condiciones que queremos utilizar el Robobo son más que correctas, que serán principalmente en vías de circulación o en locales o áreas de uso. Que las condiciones buscadas tienen que ser mayores a cien luxes como máximo, donde mejor se comporta nuestra calibración. Cabe decir que los valores obtenidos fueron en interior, por lo tanto, las situaciones serán prácticamente las mismas a la aplicación, ya que en exterior su comportamiento era mucho peor.

10.2.2 Calibración sensor de sonido

Evidentemente, los smartphones poseen micrófono, tanto para realizar llamadas como para realizar algunos comandos por voz. El micrófono es el elemento principal de los sonómetros. El funcionamiento principal del micrófono es el que las ondas de sonido golpean las membranas del micrófono y este lo transforma en impulsos eléctricos.

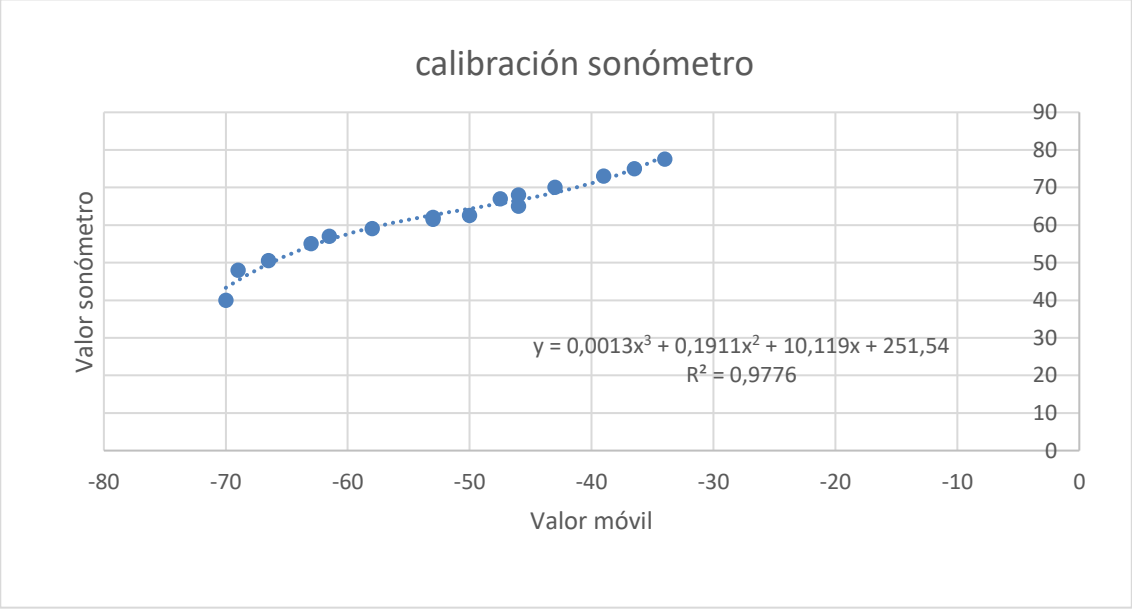
Para la calibración del micrófono se ha utilizado el comando de Robobo "readNoiseLevel()", que nos da un valor de medida de la presión sonora. La medición de este parámetro se ha hecho con la medición del sonómetro y el smartphone a la misma distancia de una fuente de sonido con distintos tipos de ruido. Los diferentes datos obtenidos son los siguientes:

Figura 23 Calibración sonómetro

Valor móvil	-53	-50	-47,5	-46	-70	-43	-39	-36,5	-34	-69	-66,5	-63
Valor real	61,5	62,5	67	68	40	70	73	75	77,5	48	50,5	55

Valor móvil	-61,5	-58	-53	-46
Valor real	57	59	62	65

Tabla 12 Datos calibración sonómetro



Gráfica 13 Calibración sonómetro

10.3 Mapeado de contaminantes a tiempo real

En este capítulo se va a exponer el experimento realizado para el mapeado de los parámetros de luminosidad y ruido en un área determinada de interior. Espacio que cumplirá las características en las cuales será de aplicación en un caso real.

El emplazamiento de este experimento será uno de los laboratorios del edificio de talleres de la Universidade Da Coruña (UDC) en el campus de Esteiro, Ferrol. En el cual, se colocarán etiquetas QR en el techo del mismo con el mayor tamaño posible dentro de un folio A3, ya que cuanto más grande con mayor precisión puede detectarlo la cámara del móvil, estos se estarán a una distancia de aproximadamente dos metros entre sí, cubriendo todo el espacio que queramos analizar. En la figura 24 podemos visualizar un croquis del espacio donde se realiza el experimento.

Hallamos los centros locales de los distintos QRs en el suelo, haremos el centro de uno el centro de nuestro eje de coordenadas global, mediremos la distancia exacta entre los QR e implementaremos en el programa de mapeado un cambio de coordenadas según el QR que este observando.

Buscaremos los mejores puntos para hacer las medidas de los diferentes parámetros y se los indicaremos al programa de mapeado con el orden que deseemos. Hay que tener en cuenta el error de los movimientos para establecer los puntos de medida, por lo tanto, no pueden ser demasiado cerca de obstáculos o de los límites de la propia área a estudiar. A continuación, se representará la sala que queremos analizar con sus distintos elementos.

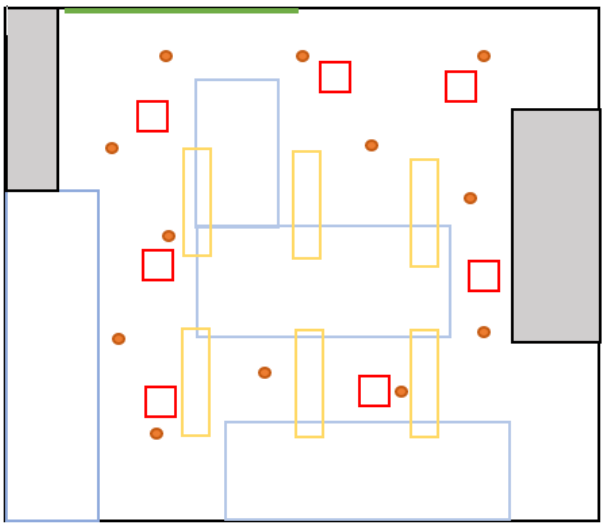


Figura 24 Mapa de espacio de trabajo

Negro =límites de la sala

Rojo = códigos QR

Gris = zonas de almacenaje

Naranja = puntos recorridos

Amarillo = luminarias de la sala

Verde = ventana

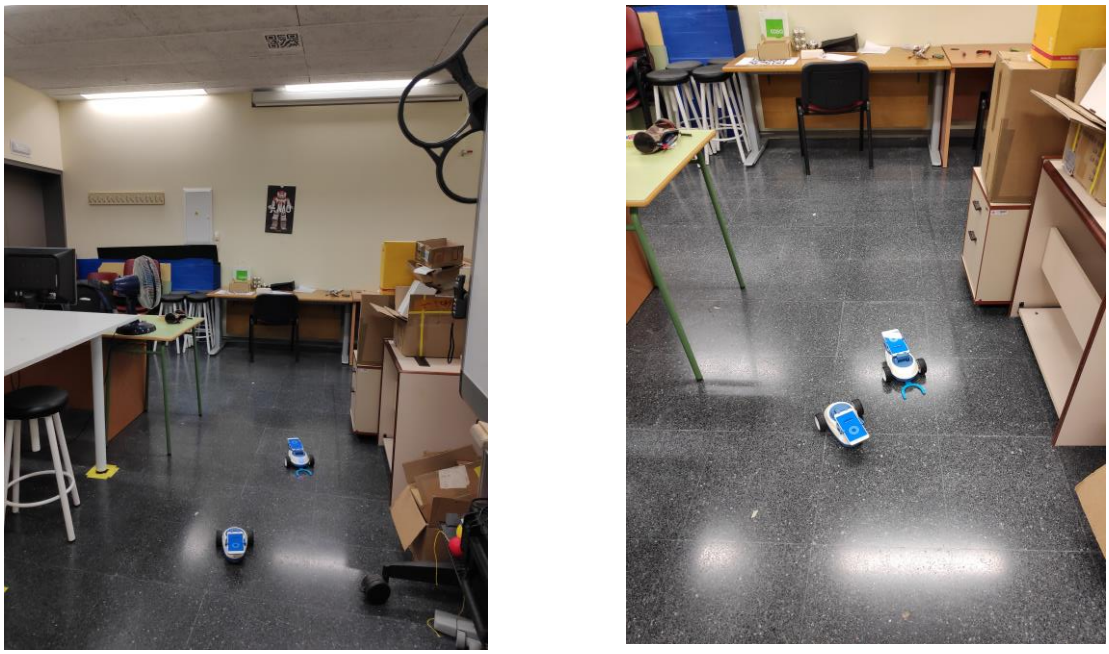
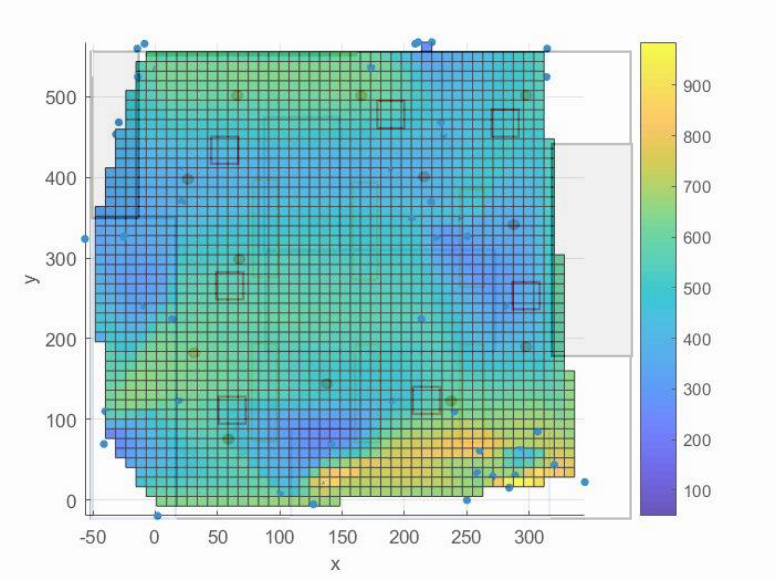


Figura 25 Medición de niveles contaminantes con sistema multirobot

10.3.1 Resultados

Una vez obtenida la matriz de resultados estos serán representados gráficamente mediante el programa MATLAB, mediante un gráfico 3D en el cual podremos observar los niveles de ruido e iluminación donde la altura es el ruido en la posición marcada y el color la luminosidad. Las gráficas resultantes son las siguientes:



En esta gráfica podemos apreciar la vista superior de los datos, de esta manera apreciamos las diferentes zonas de mayor y menor luminosidad. Las zonas de mayor luminosidad están representadas por colores más claros, como por ejemplo la zona naranja, y las zonas más oscuras a las de menor luminosidad, con tonos azules.

Gráfica 14 Vista central superior del mapa de resultados

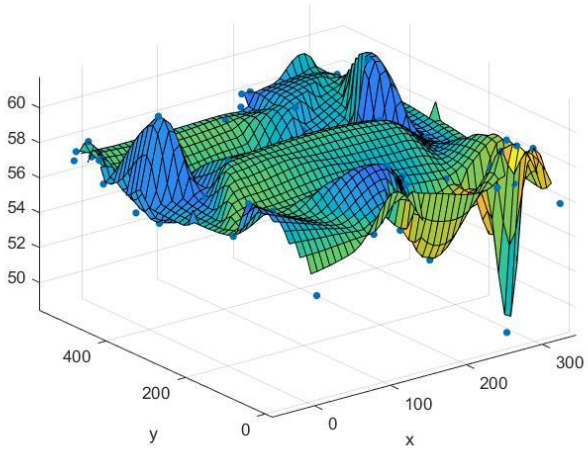
Observando estos resultados debemos comprobar si se corresponden con la realidad.

Podremos observar que los datos son ciertos ya que la zona más clara corresponde a donde se encuentran dos luminarias sin ningún tipo de obstáculo que impida la proyección de luminosidad en el espacio estudiado. La zona superior izquierda sin embargo corresponde con una luminaria en la cual se encuentra justo debajo de esta una estantería que obstaculiza la proyección de luz.

En este caso se aplica el estudio de un laboratorio, considerado una zona de exigencia visual moderada, que como se ha dicho anteriormente el nivel mínimo de luminosidad es de 200 luxes, por lo que tendremos que comprobar que se supere este nivel en toda la sala.

De la gráfica podemos observar que los niveles de luminosidad están en torno a 500 luxes por toda la sala, pero podemos encontrar picos que llegan hasta los 900 luxes debajo de las luminarias del laboratorio y zonas que están entre los 400 y 200 luxes en los laterales, que se corresponden con las zonas que menos luz reciben de las luminarias, podemos observar también que al fondo del espacio de trabajo, donde se encuentra la ventana tendremos unos valores de entre 600 y 725 luxes gracias a la luz incidente del exterior.

Por lo tanto, concluiremos diciendo que los niveles de luminosidad en el laboratorio son correctos



En la gráfica 15 observamos los niveles de ruido y en qué zonas se generan, como podemos comprobar las medidas tienen una gran diversidad, debido a que las medidas son muy dispares según el momento en el que se realizan, de esta grafica simplemente tendremos que fijarnos en que no pase un determinado umbral de ruido.

Gráfica 15 Vista tridimensional del mapa de resultados

Como ya se ha dicho anteriormente, los niveles de ruido en la oficina no deben superar los 50 dBA, por lo tanto, no cumplen las condiciones requeridas. Esto es debido a que nos encontramos en un laboratorio y se generan ruidos relacionados con la actividad desempleada en el mismo.

11 PRESUPUESTO

A continuación, vamos a reflejar el presupuesto del proyecto, con sus diferentes componentes, obteniendo así el importe de ejecución material. Aplicándole a este el beneficio industrial (6%) y los gastos generales (13%) y se obtiene importe de ejecución. Al cual le aplicamos el IVA (16%) y obtendremos el importe de contrata.

		Precio (€) / unidad	Unidades	Coste
Componentes	Robobo	400	2	800
	smartphone	200	2	400
Mano de obra		400	40	16000
Importe de ejecución material				17200

Tabla 13 presupuesto de mano de obra y componentes del proyecto

Importe de ejecución material	17200
Gastos generales (13%)	2236
Beneficio industrial (6%)	1032
Importe de ejecución	20468
IVA (16%)	3274,88
Importe de contrata	23742,88

Tabla 14 Distintos importes del proyecto

12 CONCLUSIONES

El objetivo que nos ha llevado a realizar este proyecto se centró en el desarrollo de librerías de control para aplicaciones de robótica colectiva. Para su desarrollo fue necesario el estudio de los movimientos del robot en cuestión, en nuestro caso la plataforma móvil con smartphone desarrollada por el Grupo Integrado de Ingeniería (GII) de la Universidade da Coruña (UDC), el Robobo.

Para la coordinación de varios robots es necesaria la localización de estos en el espacio, de esta manera se ha diseñado un sistema basado en la localización de etiquetas QR que nos dará la posición del Robobo en cuestión.

Seguidamente se ha realizado un sistema de navegación en interior con sus respectivos algoritmos para su correcto funcionamiento, como el de movimiento entre dos puntos, navegación entre QRs y navegación mixta, y evasión de obstáculos.

Por último, se ha buscado una aplicación práctica a este sistema, la búsqueda de zonas fuera de los umbrales óptimos de ruido y luminosidad en un espacio de trabajo. Por consiguiente se han calibrado los sensores de luminosidad y sonido del móvil para así monitorizar los datos obtenidos y realizar un mapeado con ellos.

13 BIBLIOGRAFÍA

https://como-funciona.co/un-luxometro/

http://www.insht.es/InshtWeb/Contenidos/Normativa/TextosLegales/RD/1997/486_97/Ficheros/anexo_iv.pdf

http://www.insht.es/InshtWeb/Contenidos/Documentacion/Iluminacion%20en%20el%20puesto%20de%20trabajo.pdf

http://riesgoslaborales.feteugt-sma.es/wp-content/uploads/2017/02/486_lugares_trab.pdf

http://riesgoslaborales.feteugt-sma.es/portal-preventivo/riesgos-laborales/riesgos-laborales-segun-los-lugares-de-trabajo/

http://www.coordinacionempresarial.com/que-se-entiende-por-centro-de-trabajo/

https://www.isotools.org/2015/09/10/riesgo-laboral-definicion-y-conceptos-basicos/

https://www.quironprevencion.com/blogs/es/prevenidos/prevencion-riesgos-laborales-prl

http://riesgoslaborales.feteugt-sma.es/portal-preventivo/riesgos-laborales/riesgos-laborales-segun-los-lugares-de-trabajo/

https://www.fing.edu.uy/inco/cursos/robotica/teorico/IAR10_Clase07_Cooperacion.pdf

https://developer.mozilla.org/es/docs/Web/JavaScript

https://descubrearduino.com/ros/

https://therobobоproject.com/que-es-robobo/

https://e-archivo.uc3m.es/bitstream/handle/10016/11629/tesis_armingol_1997.pdf

http://wiki.robotica.webs.upv.es/wiki-de-robotica/introduccion/clasificacion-de-robots/

https://definicion.de/robotica/

https://www.monografias.com/trabajos107/evolucion-robotica/evolucion-robotica.shtml

http://sisbib.unmsm.edu.pe/bibvirtual/publicaciones/indata/v04_n1/actualidad.htm

Escola Politécnica Superior

TRABAJO FIN DE GRADO

CURSO 2018/19

DESARROLLO DE LIBRERÍAS DE CONTROL PARA APLICACIONES DE COORDINACIÓN EN ROBÓTICA COLECTIVA

Grado en Ingeniería en Tecnologías Industriales

Documento II

ANEXO

14 ANEXO

```
/* Programa javascript de evasión de obstáculos */
async function main(obstacs, posi, posf) {
  var Robobo = require("./lib/robobo");
  robobo = new Robobo("192.168.0.20");
  await robobo.connect();
  await robobo.pause(1);
  // calcula la mejor trayectoria a seguir para salvar al obstaculo
  //acordarse de que hay que poner obstacs con el primer punto al final
  var alfa = (-90 * Math.PI) / 180;
  obstacs.push(obstacs[0]);
  var mejorcrit = calcmejorcrit(obstacs, posi, posf, alfa);
  await moverserecorrido(posi, posf, obstacs, mejorcrit, alfa, robobo);
}
async function moverserecorrido(posi, posf, obstacs, crit, alfa, robobo) {
  var dist,
    recorrido = 0,
    pos = posi,
    ptocorte,
    vectori_f_x = posf[0] - posi[0],
    vectori_f_y = posf[1] - posi[1];
  dist = Math.sqrt(vectori_f_x * vectori_f_x + vectori_f_y * vectori_f_y);
  while (dist > 5) {
    ptocorte = ptocortecercano(obstacs, pos, posf);
    if (!ptocorte) {
      vectori_f_x = posf[0] - pos[0];
      vectori_f_y = posf[1] - pos[1];
      dist = Math.sqrt(vectori_f_x * vectori_f_x + vectori_f_y * vectori_f_y);
      recorrido = recorrido + dist;
      var beta = Math.atan2(posf[1], posf[0]);
      giro = normalizeAngleRadians(beta - alfa);
      await moversenextstep(dist, giro, robobo);
      dist = 0;
    } else {
      nextstep = calcnextstep(obstacs, pos, ptocorte, alfa, crit);
      await moversenextstep(nextstep[2], nextstep[3], robobo);
      recorrido = recorrido + nextstep[2];
      vectori_f_x = posf[0] - nextstep[0];
      vectori_f_y = posf[1] - nextstep[1];
      dist = Math.sqrt(vectori_f_x * vectori_f_x + vectori_f_y * vectori_f_y);
      pos = [nextstep[0], nextstep[1]];
    }
  }
}
async function moversenextstep(dist, giro, robobo) {
  // hace que se mueva el robobo una cierta distancia y un giro determinado
  var distgiro = giro * (14.75 / 2);
  var timegir = (Math.abs(distgiro) + 1.4243) / 6.3161;
  var speed = 10;
  if (giro > 0) {
    await robobo.moveWheelsByTimeBLK(-speed, speed, timegir);
  } else {
    await robobo.moveWheelsByTimeBLK(speed, -speed, timegir);
  }
  var timerecto = (Math.abs(dist) + 1.4243) / 6.3161;
  await robobo.moveWheelsByTimeBLK(speed, speed, timerecto);
  robobo.stopMotors();
}
function calcmejorcrit(obstacs, posi, posf, alfa) {
  // nos da el mejor criterio para hacer la trayectoria
  var mejorcrit = 1;
```

```
  var trayectopos = calctrayectoria(obstacs, posi, posf, alfa, 1);
  var trayectoneg = calctrayectoria(obstacs, posi, posf, alfa, -1);
  if (trayectoneg < trayectopos) {
    mejorcrit = -1;
  }
  return mejorcrit;
}
function calctrayectoria(obstacs, posi, posf, alfa, crit) {
  // nos da la distancia que hay que recorrer por un lado
  var recorrido = 0,
    pos = posi,
    ptocorte,
    dist = distancia(posi, posf);
  while (dist > 5) {
    ptocorte = ptocortecercano(obstacs, pos, posf);
    if (!ptocorte) {
      dist = distancia(pos, posf);
      recorrido = recorrido + distancia;
      dist = 0;
    } else {
      nextstep = calcnextstep(obstacs, pos, ptocorte, alfa, crit);
      recorrido = recorrido + nextstep[2];
      pos = [nextstep[0], nextstep[1]];
      dist = distancia(pos, posf);
    }
  }
  return recorrido;
}
function calcnextstep(obstacs, posi, ptocorte, alfa, crit) {
  // calcula el siguiente pto de la trayectoria y la distancia entre estos dos puntos y el giro
  // nos devuelve posx y posy del siguiente pto, la distancia que hay hacia el y el giro a realizar
  var i = 0,
    destino,
    dist,
    pos_destinox,
    pos_destinoy,
    Ds = 10,
    destino_x_1,
    destino_x_2,
    destino_y_1,
    destino_y_2,
    destinox,
    destinoy,
    giro;
  if (crit == 1) {
    i = 1;
  }
  destino = ptocorte[0] + i;
  pos_destinox = obstacs[destino][0] - posi[0];
  pos_destinoy = obstacs[destino][1] - posi[1];
  destino_x_1 =
    obstacs[destino][0] +
    (-pos_destinoy /
      Math.sqrt(pos_destinox * pos_destinox + pos_destinoy * pos_destinoy)) *
      Ds;
  destino_y_1 =
    obstacs[destino][1] +
    (pos_destinox /
```

```
      Math.sqrt(pos_destinox * pos_destinox + pos_destinoy * pos_destinoy)) *
      Ds;
  destino_x_2 =
    obstacs[destino][0] +
    (pos_destinoy /
      Math.sqrt(pos_destinox * pos_destinox + pos_destinoy * pos_destinoy)) *
      Ds;
  destino_y_2 =
    obstacs[destino][1] +
    (-pos_destinox /
      Math.sqrt(pos_destinox * pos_destinox + pos_destinoy * pos_destinoy)) *
      Ds;
  vectori_destino1 = [destino_x_1 - posi[0], destino_y_1 - posi[1]];
  vectori_destino2 = [destino_x_2 - posi[0], destino_y_2 - posi[1]];
  var a1 = Math.atan2(vectori_destino1[1], vectori_destino1[0]);
  var a1grad = (a1 * 180) / Math.PI;
  var a2 = Math.atan2(vectori_destino2[1], vectori_destino2[0]);
  var a2grad = (a2 * 180) / Math.PI;
  var d1 = normalizeAngleRadians(a1 - a2);
  var d2 = normalizeAngleRadians(a2 - a1);
  if (d1 < d2) {
    var CeligiendoV1 = Math.sign(d1);
  } else {
    var CeligiendoV1 = Math.sign(d2);
  }
  if (CeligiendoV1 == crit) {
    giro = normalizeAngleRadians(a1 - alfa);
    var termx = destino_x_1 - posi[0];
    var termy = destino_y_1 - posi[1];
    destinox =
      destino_x_1 + (termx / Math.sqrt(termx * termx + termy * termy)) * Ds;
    destinoy =
      destino_y_1 + (termy / Math.sqrt(termx * termx + termy * termy)) * Ds;
  } else {
    giro = normalizeAngleRadians(a2 - alfa);
    var termx = destino_x_2 - posi[0];
    var termy = destino_y_2 - posi[1];
    destinox =
      destino_x_2 + (termx / Math.sqrt(termx * termx + termy * termy)) * Ds;
    destinoy =
      destino_y_2 + (termy / Math.sqrt(termx * termx + termy * termy)) * Ds;
  }
  dist = distancia(posi, [destinox, destinoy]);
  giro = (giro * 180) / Math.PI;
  return [destinox, destinoy, dist, giro];
}
function normalizeAngleRadians(angle) {
  newAngle = angle;
  while (newAngle <= -Math.PI) newAngle += 2 * Math.PI;
  while (newAngle > Math.PI) newAngle -= 2 * Math.PI;
  return newAngle;
}
function ptocortecercano(obstacs, posi, posf) {
  // nos da el punto mas cercano en el que se cortan la recta pto inicial-origen con
el poligono
  // nos da el numero de la recta con la que se corta que sera el numero del primer
pto y la pos x e y de este
  var rectas = calculoderectas(obstacs);
  var rectaoi = rectaorigen(posi, posf);
  var ptoscort = ptoscorte(rectaoi, rectas, obstacs);
```

```
  var ptocortcerc;
  if (ptoscort.length < 2) {
    ptocortcerc = false;
  } else {
    ptocortcerc = escogerptocortecercano(ptoscort, posi);
  }
  return ptocortcerc;
}
function calculoderectas(obstacs) {
  // esta funcion nos da todas las rectas posibles entre los puntos consecutivos
  var n = obstacs.length,
    rectas = [];
  for (var i = 0; i < n - 1; i++) {
    rectas[i] = [
      obstacs[i + 1][1] - obstacs[i][1],
      -(obstacs[i + 1][0] - obstacs[i][0]),
      -obstacs[i][0] * (obstacs[i + 1][1] - obstacs[i][1]) +
        obstacs[i][1] * (obstacs[i + 1][0] - obstacs[i][0])
    ];
  }
  return rectas;
}
function rectaorigen(posi, posf) {
  // Esta función nos da la recta que va desde el pto origen hasta el pto final
  var rectaposi_f = [
    posf[1] - posi[1],
    -(posf[0] - posi[0]),
    -posi[0] * (posf[1] - posi[1]) + posi[1] * (posf[0] - posi[0])
  ];
  return rectaposi_f;
}
function ptoscorte(rectaoi, rectas, obstacs) {
  // nos da todos los puntos de corteque haya entre las rectasdel poligono y nuestra
recta trayectoria
  var n = rectas.length;
  var ptoscort = [];
  var rectascortantes = [];
  for (var i = 0; i < n; i++) {
    ptoscort[i] = [
      -(-rectas[i][2] / rectas[i][1] + rectaoi[2] / rectaoi[1]) /
        (-rectas[i][0] / rectas[i][1] + rectaoi[0] / rectaoi[1]),
      -(-rectas[i][2] / rectas[i][0] + rectaoi[2] / rectaoi[0]) /
        (-rectas[i][1] / rectas[i][0] + rectaoi[1] / rectaoi[0])
    ];
    if (
      ptoscort[i][0] <= Math.max(obstacs[i][0], obstacs[i + 1][0]) &&
      ptoscort[i][0] >= Math.min(obstacs[i][0], obstacs[i + 1][0])
    ) {
      var ptocortreal = [i, ptoscort[i][0], ptoscort[i][1]]; // numero de la recta
que corta y los ptos de corte
      rectascortantes.push(ptocortreal);
    }
  }
  return rectascortantes;
}
function escogerptocortecercano(ptoscort, posi) {
  // escoge entre los puntos de corte el pto de corte mas cercano
  //y nos da el numero de la recta y la posicion delpto de corte
  var n = ptoscort.length;
  var dist,
```

```
    cercano = 10000,
    poscercano,
    vectorptoi_x,
    vectorptoi_y;
  for (var i = 0; i < n; i++) {
    vectorptoi_x = posi[0] - ptoscort[i][1];
    vectorptoi_y = posi[1] - ptoscort[i][2];
    dist = Math.sqrt(vectorptoi_x * vectorptoi_x + vectorptoi_y * vectorptoi_y);
    if (cercano > dist) {
      cercano = dist;
      poscercano = i;
    }
  }
  return [
    ptoscort[poscercano][0],
    ptoscort[poscercano][1],
    ptoscort[poscercano][2]
  ];
}
function distancia(posi, posf) {
  var vectori_f_x = posf[0] - posi[0];
  var vectori_f_y = posf[1] - posi[1];
  var dist = Math.sqrt(vectori_f_x * vectori_f_x + vectori_f_y * vectori_f_y);
  return dist;
}
main([[20, 15], [30, 20], [40, 10], [20, -5]], [25, 50], [30, -20]);

```

```
/* Programa javascript de MAPEADO */

async function main(ptos) {
  var csvWriter = require("csv-write-stream");
  fs = require("fs");
  writer = csvWriter({ sendHeaders: false });
  var Robobo = require("./lib/robobo");
  robobo = new Robobo("10.113.36.172");
  await robobo.connect();

  await robobo.pause(1);

  // calcula la mejor trayectoria a seguir para salvar al obstaculo
  //acordarse de que hay que poner obstacs con el primer punto al final
  var qrleido = false,
    qr,
    numptos = ptos.length,
    npos = 0,
    anyQR = false,
    recoordenada;

  robobo.whenQRCodeIsDetected(() => {
    if (!qrleido) {
      qr = robobo.readQR();
      recoordenada = newcoord(qr);
      qrleido = true;
      anyQR = true;
    }
  });

  robobo.whenATapIsDetected(() => {
    robobo.stopMotors();
  });

  ptos.push(ptos[0]);
  crearfichero("out.csv");

  while (true) {

    await robobo.update();

    if (qrleido) {

      var dest = npos % numptos;
      monitorizar(recoordenada, 1, robobo, alfa_estimado);
      await movnextp(recoordenada, ptos[dest + 1], robobo);
      var alfa_estimado = calcgiro(ptos[dest], ptos[dest + 1])[1];
      var t_inicio_espera = new Date();
      qrleido = false;
      npos++;

    } else if ((new Date() - t_inicio_espera > 2000) & anyQR) {

      qrleido = true;
      var dest = npos % numptos;
      var posi_estimada = [ptos[dest][0], ptos[dest][1], alfa_estimado];
      await movnextp(posi_estimada, ptos[dest + 1], robobo);
      alfa_estimado = calcgiro(ptos[dest], ptos[dest + 1], alfa_estimado)[1];
      monitorizar(ptos[dest + 1], 0, robobo, alfa_estimado);
      npos++;
```

```
      var t_inicio_espera = new Date();
      qrleido = false;

    }
  }
}

function readSound(robobo) {
  var vsound = JSON.stringify(robobo.readNoiseLevel());

  return noise2db(vsound);
}
function noise2db(noise) {
  dbanoise =
    0.0013 * noise * noise * noise +
    0.1911 * noise * noise +
    10.119 * noise +
    251.54;

  return dbanoise;
}

function readlux(robobo) {
  var brightness = robobo.readBrightnessSensor();
  return brightness2luxes(brightness);
}
function brightness2luxes(brightness) {
  var luxes = 0.0016 * brightness * brightness + 0.2171 * brightness + 48.368;

  return luxes;
}
function crearfichero(nombre) {
  writer.pipe(fs.createWriteStream(nombre));
}
function monitorizar(pos, fromQR, robobo, alfa_estimado) {
  var posampl = [];

  var L = readlux(robobo);

  var S = readSound(robobo);

  posampl.push([pos[0], pos[1], fromQR, L, S, alfa_estimado]);

  console.log(
    pos[0] +" " + pos[1] +" " +alfa_estimado +" " +fromQR +" " + L +" " +S);

  writer.write({
    C1: pos[0],
    C2: pos[1],
    angulo: alfa_estimado,
    leido: fromQR,
    Lux: L,
    dBA: S
  });

  return posampl;
}
async function movnextp(posi, posf, robobo) {
  // hace que se mueva el robobo una cierta distancia y un giro determinado
  var giro = calcgiro(posi, posf);
```

```
  var distgiro = (giro[0] * (14.75 / 2) * Math.PI) / 180;
  var timegir = (Math.abs(distgiro) + 1.4243) / 6.3161;
  var speed = 10;

  if (giro[0] < 0) {
    await robobo.moveWheelsByTimeBLK(-speed, speed, timegir);
  } else {
    await robobo.moveWheelsByTimeBLK(speed, -speed, timegir);
  }
  var dist = calcdistancia(posi, posf);
  var timerecto = (Math.abs(dist) + 1.4243) / 6.3161;
  await robobo.moveWheelsByTimeBLK(speed, speed, timerecto);
}
function calcgiro(posi, posf) {
  var vx_i_f = posf[0] - posi[0];
  var vy_i_f = posf[1] - posi[1];
  var beta = Math.atan2(vy_i_f, vx_i_f);
  var giro = beta - (posi[2] * Math.PI) / 180;
  giro = (giro * 180) / Math.PI;
  beta = (beta * 180) / Math.PI;
  return [giro, beta];
}
function calcdistancia(posi, posf) {
  var vx_i_f = posi[0] - posf[0];
  var vy_i_f = posi[1] - posf[1];
  dist = Math.sqrt(vx_i_f * vx_i_f + vy_i_f * vy_i_f);
  return dist;
}
function newcoord(qr) {
  var x = qr.p1.x - qr.p2.x;
  var y = qr.p1.y - qr.p2.y;
  var v1 = [x, y];
  var modV1 = Math.sqrt(x * x + y * y);
  v1[0] = v1[0] / modV1;
  v1[1] = v1[1] / modV1;
  x = qr.p3.x - qr.p2.x;
  y = qr.p3.y - qr.p2.y;
  var v2 = [x, y];
  var modV2 = Math.sqrt(x * x + y * y);
  v2[0] = v2[0] / modV2;
  v2[1] = v2[1] / modV2;
  x = 244.2 - qr.p2.x;
  y = 330.3 - qr.p2.y;
  var orgn = [x, y];
  var invMatCurrent = inverseMat(v1, v2);
  var C = multMat2221(invMatCurrent, orgn);

  var C1 = -C[0] / 2.256;
  var C2 = C[1] / 2.256;

  var angulo =
    (180 * Math.atan2(qr.p3.y - qr.p2.y, qr.p3.x - qr.p2.x)) / Math.PI;

  console.log("id: " + qr.id);

  if (qr.id == "a") {
    C1 = C1 + 210;
    C2 = C2 + 10;
  }
  if (qr.id == "c") {
```

```
    C1 = C1 - 10;
    C2 = C2 + 250;
  }
  if (qr.id == "d") {
    C1 = C1 - 5;
    C2 = C2 + 490;
  }

  return [C1, C2, angulo];
}

function inverseMat(v1, v2) {

  //[v1x v2x ; v1y v2y]

  a = v1[0];
  b = v2[0];
  c = v1[1];
  d = v2[1];
  det = a * d - b * c;
  ai = d / det;
  bi = -b / det;
  ci = -c / det;
  di = a / det;
  invmat = [ai, bi, ci, di];

  return invmat;
}

function multMat2221(mat, vec) {

  a = mat[0];
  b = mat[1];
  c = mat[2];
  d = mat[3];
  x = vec[0];
  y = vec[1];
  xo = a * x + b * y;
  yo = c * x + d * y;
  veco = [xo, yo];

  return veco;
}

main([
  [330, 60],
  [260, -10],
  [260, 60],
  [20, -60],
  [-40, 110],
  [20, 310],
  [-45, 440],
  [20, 550],
  [-15, 570],
  [-20, 330],
  [-10, 250],
  [130, 0],
  [250, 0]
]);

```