DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE NAVEGACION AUTONOMA PARA UN VEHICULO DE CAMPOS AGRICOLAS DE MAIZ

SANTIAGO NIÑO RODRIGUEZ

JUAN DAVID SALAZAR TRIANA

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS

ESCUELA DE INGENIERÍA MECÁNICA

BUCARAMANGA

2018

DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE NAVEGACION AUTONOMA PARA UN VEHICULO DE CAMPOS AGRICOLAS DE MAIZ

SANTIAGO NIÑO RODRIGUEZ
JUAN DAVID SALAZAR TRIANA

Trabajo de grado para optar al título de ingeniero mecánico

Director
CARLOS BORRÁS PINILLA
Ingeniero Mecánico

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍA MECÁNICA
BUCARAMANGA
2018

A Dios.

Por estar siempre a mi lado brindándome amor, protección, sabiduría, salud y perseverancia para poder lograr mis objetivos como estudiante de Ingeniería Mecánica.

A mis padres.

Gracias a mi madre "Elisa Triana Vargas" y mi padre "José Salazar Benavides" por ser pilares fundamentales en mi vida, por brindarme un amor incondicional desde mi primer día en este mundo, por estar siempre cuando he caído y brindarme una mano y una voz alentadora a seguir luchando, por sus consejos que son sabios los cuales me han permitido vencer infinidad de obstáculos. Padres gracias, todo el esfuerzo durante el transcurso de mi vida y este proyecto no habría sido posible sin su apoyo y amor incondicional.

A mis amigos.

Por brindarme una gran amistad y camaradería durante el transcurso de mi vida universitaria, gracias por estar en esos momentos difíciles dentro y fuera de la universidad dando una mano amiga y consejos alentadores para vencer esos grandes obstáculos que se nos presentan día a día como estudiantes.

A mi familia.

Gracias por siempre confiarme en mí, darme un amor incondicional como unos segundos padres, en ser un gran apoyo emocional en esos momentos donde uno necesita voces de aliento. Solo puedo decirles gracias y este logro también les pertenece.

Juan David Salazar Triana

A mis padres Luis y Gloria por su apoyo incondicional

A mi hermana Laura que me enseño que todo tiene solución y a mi hermano Sebastián que me mostro como encontrarla.

A todas las personas que me acompañaron durante la carrera e hicieron de esta un viaje.

Santiago Niño Rodriguez

TABLA DE CONTENIDO

LISTA DE FIGURAS

Pág.

LISTADO DE TABLAS

Pág.

RESUMEN

TITULO: DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE NAVEGACION AUTONOMA PARA UN VEHICULO DE CAMPOS AGRICOLAS DE MAIZ*

AUTORES: SANTIAGO NIÑO RODRIGUEZ
JUAN DAVID SALAZAR TRIANA**

PALABRAS CLAVE: Robótica autónoma, localización, percepción, navegación, Dijkstra, Campos potenciales.

DESCRIPCION:

El objetivo de este proyecto de grado es el de proveer a la Escuela de Ingeniería Mecánica una plataforma robótica autónoma de exploración enfocado en campos agrícolas controlados, para que sea la base del desarrollo tecnológico e investigativo en modelos autónomos para las futuras generaciones.

La plataforma robótica se diseñó en base a unos requerimientos de diseño preestablecidos para abarcar los aspectos relacionados al movimiento autónomo del mismo. Para la simulación de los modelos matemáticos se empleó la herramienta computacional MatLab para determinar el comportamiento de los modelos matemáticos de generación de trayectorias Dijkstra y Campos Potenciales en un campo agrícola simulado. Así mismo, se empleó el lenguaje de programación Python para la implementación de dichos modelos en la tarjeta-computador Raspberry Pi, la cual servirá como controlador de la plataforma robótica.

El resultado es un sistema de navegación autónomo implementado en un robot tipo rover con un modelo de locomoción por deslizamiento, el cual está apoyado con un sistema de localización basado en procesamiento de imágenes mediante el uso de una cámara externa y un sistema de percepción dotado con una brújula magnética, encoders y sensores de distancia ultrasónicos. Además, permite el intercambio de datos en tiempo real con una estación remota mediante el uso de un módulo Wi-Fi integrado a la tarjeta-computador.

* Trabajo de Grado.
** Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería Mecánica. Director: Carlos Borras Pinilla.

SUMMARY

TITLE: DESIGN AND CONSTRUCTION OF AN AUTONOMOUS NAVIGATION SYSTEM FOR A CORN FIELD VEHICLE*

AUTHORS: SANTIAGO NIÑO RODRIGUEZ
JUAN DAVID SALAZAR TRIANA**

KEYWORDS: Autonomous robotics, localization. Perception, Navigation, Dijkstra, Potential fields.

DESCRIPTION:

The objective of this project is to provide the Mechanical Engineering School with an autonomous robotic platform for exploration focused on agricultural controlled fields, in order to set the bases for the technological and research development for future generations in autonomous models.

The robotic platform was designed based in a set of pre-established design requirements in order to compass the aspects related to autonomous execution. For the simulation of the mathematical models, was used the computational tool MatLab as a means to determine the behavior of the path planning models Dijkstra and Potential Fields in a simulated field. Likewise, it was used the programming language Python in order to implement said models in the computer-board Raspberry Pi, which will act as controller of the robotic platform.

The result is an autonomous navigation system implemented in a rover type robot with a skid steering drive, which is helped by a localization system based in image processing using an external camera, and a perception system that includes a magnetic compass, encoders and ultrasonic distance sensors. Additionally, it allows the real time data transfer with a remote station through a WiFi module integrated to the computer-board.

* Degree work.
** Physicomechanical Faculty of Engineering. Mechanical Engineering Faculty. Director: Carlos Borras Pinilla.

INTRODUCCÓN

El presente proyecto plantea el proceso de diseño y construcción de una plataforma móvil autónoma enfocada en la exploración de cultivos controlados, con el fin de apoyar el desarrollo de la investigación en el área de la robótica autónoma de la facultad de ingeniería mecánica de la Universidad Industrial de Santander. Para la creación de esta plataforma autónoma se diseñó un sistema de percepción que le otorgara la información requerida sobre el entorno próximo a él. Adicionalmente cuenta con un sistema de localización, que además de brindar la posición de la plataforma con gran precisión, entrega una imagen detallada del entorno completo de trabajo. Estos sistemas serán guiados por un algoritmo de búsqueda y creación de trayectorias, lo que permitirán que la plataforma se mueva de manera totalmente autónoma.

En este libro se elaborará un marco teórico sobre los diferentes tipos de algoritmos de navegación, así como también se entregará un estado del arte del uso de plataformas robóticas en el sector agrícola. Además, se hará un estudio detallado de los algoritmos de creación de trayectorias Dijkstra y Campos Potenciales, los cuales son objeto de trabajo en este proyecto, con el fin de apoyar la investigación en algoritmos de búsqueda.

Mediante el uso de plataformas robustas de hardware y software se facilitará el proceso de creación del modelo de control general, que tiene como fin crear las bases para la investigación y aplicación de modelos autónomos más complejos a plataformas robóticas móviles.

1. DESCRIPCÓN DEL PROYECTO

1.1 PLANTEAMIENTO DEL PROBLEMA

La industria agrícola moderna, es una de las industrias actuales con la menor automatización de los procesos, en donde la intuición y la experiencia del operador son la única garantía de éxito. La irregularidad de los ambientes a los que se enfrenta un operador de un vehículo agrícola opone una gran dificultad en la automatización de este proceso, siendo necesario la implementación de un sistema inteligente que se rija bajo las mismas normas lógicas del operador.

El aumento de la demanda en la producción de alimentos recae directamente sobre el sector agrícola, lo que genera una necesidad de mejorar los procesos de producción de alimentos. La industria agrícola presenta una numerosa cantidad de procesos, los cuales se extienden a cada uno de los productos que se requiere cultivar, regular y extraer; esta razón junto con su pobre automatización es lo que nos obliga a entrar a fondo en esta industria para explorar sus diferentes inconvenientes. Una de las más grandes dificultades a la hora de implementar procesos de automatización de procesos en movimiento, es la de realizar procesos de localización del sistema a la vez que se retroalimenta con los obstáculos que encuentra a su paso. Se hace entonces necesaria la implementación de una lógica que tenga en cuenta las variables del proceso de la misma manera que el operador usa su experiencia.

La industria agrícola actual cuenta con una numerosa cantidad de procesos mecánicos, los cuales son pobres y en la mayoría de los casos ineficientes debido a que estos no cuentan con mayor innovación desde hace ya algunos años. Aquí entra a jugar un papel primordial la ingeniería electrónica y la lógica computacional,

la cual viene innovando las diferentes áreas tanto de la vida cotidiana como de las industrias.

La creciente competencia con industrias agrícolas de otros países, los cuales disponen de vehículos agrícolas más eficientes, hace necesario la creación de un elemento innovador que ayude a introducir nuestro mercado mediante procesos más económicos y rentables a largo plazo, mejorando los tiempos de producción y reduciendo los márgenes de error al mínimo bajo la lógica computacional.

En este proyecto se desarrollará una serie de algoritmos que tiene como fin evaluar las condiciones en las que un vehículo de cuatro ruedas se desplazara interactuando con la información que recopila con sus diferentes sensores. El algoritmo deberá evaluar las condiciones del terreno como también las capacidades de sus motores para realizar una acción adecuada.

1.2 JUSTIFICACIÓN PARA SOLUCIONAR EL PROBLEMA

El propósito de este proyecto es generar un modelo de localización espacial, el cual dejara una herramienta primordial en las librerías programables que se usaran posteriormente en modelos agrícolas e incluso en modelos industriales.

Poder dejar un aporte educativo a nuestra universidad, a fin de que las siguientes generaciones puedan seguir profundizando en esta rama de la ingeniería mecánica, y sobre todo en el campo agrícola, ya que es una faceta importante de nuestro país.

Al optimizar los procesos agrícolas actuales se harán avances en los métodos de plantación y producción de alimentos, donde la precisión de las maquinas abrirá nuevas formas de sembrar que eviten la erosión del suelo.

Los procesos actuales de producción de alimentos están limitados por la mano de obra actual, la cual toma en cuenta salarios, tiempos muertos, seguros médicos, entre otros recursos humanos. El proyecto tiene como fin generar un avance en la industria agrícola para que elimine todos estos inconvenientes en la producción; eso no significa que el ser humano será excluido de las tareas agrícolas, este, en cambio ira de la mano con todas las aptitudes que debe desarrollar la máquina.

1.3 OBJETIVOS DEL TRABAJO DE GRADO

1.3.1 Objetivo general. Contribuir con la misión de la Escuela de Ingeniería Mecánica mediante la formación de Ingenieros Mecánicos con alta calidad científica y apoyar la investigación robótica en el sector agrícola a través de la programación e implementación de un sistema de navegación autónoma a un prototipo de robot para ser utilizado en un campo de maíz con dimensiones establecidas.

1.3.2 Objetivos específicos

- Simular dos modelos matemáticos de navegación en la herramienta de software científico MATLAB con las condiciones aproximadas del campo de maíz controlado (terreno plano (5m x 5m) y anchura entre filas de 60cm). Los modelos matemáticos seleccionados son:
  - Algoritmo Dijsktra para la planeación de rutas
  - Algoritmo de planeación de trayectoria basada en campos potenciales.

- Determinar cuál de los dos modelos matemáticos resulta más viable frente a las condiciones aproximadas del terreno agrícola simulado, tiempo de procesamiento, uso del CPU y uso de memoria, para ser implementado en el robot Lynxmotion 4WD01 del grupo DicBot.

- Diseñar y construir los sistemas de percepción y localización para la plataforma robótica Lynxmotion 4WD01 con las siguientes características:
  - un sistema de percepción exteroceptivo que le brinde la capacidad de evadir obstáculos y que apoye el sistema de navegación autónoma.
  - sistema de posicionamiento absoluto que le permita ubicarse en el espacio explorado con un error menor o igual a 1 m.

- Desarrollar y programar las funciones necesarias para adquirir información de sus sistemas (sensores para la detección de obstáculos, dispositivos de posicionamiento y orientación) en la placa computador Raspberry Pi mediante el uso del lenguaje de programación libre Python, para realizar las acciones de control sobre la plataforma y facilitar al usuario la programación de los algoritmos de navegación.

- Implementar el algoritmo de navegación en la plataforma robótica a través de la placa computador Raspberry Pi para ser verificado en el campo agrícola de maíz emulado, donde las condiciones y parámetros del terreno están controladas.

1.4 JUSTIFICACIÓN DE LA SOLUCIÓN

El propósito de este proyecto es dejar un aporte educativo a la universidad industrial de Santander, con el fin de incentivar a las siguientes generaciones en seguir profundizando en esta rama de la ingeniería mecánica y como este puede tener un trasfondo funcional en el área agrícola, área cuyos procesos aun no son automatizados en Colombia.

Mediante la implementación de diferentes modelos de navegación autónoma se busca generar una herramienta básica en las librerías programables que se usaran posteriormente en modelos que requieran navegación y manejo de trayectorias. La

documentación de estos modelos matemáticos abrirá las puertas en un nuevo ámbito de la robótica en la escuela de ingeniería mecánica, el cual será la base para la evolución del prototipo a fases más avanzadas que abarquen más ramas de la robótica.

1.5 ESTUDIO DE LOS REQUERIMIENTOS DE DISEÑO

1.5.1 Requerimientos del cliente o consumidor. Mediante una encuesta se establecen las expectativas que los posibles usuarios (estudiantes de ingeniería mecánica de la UIS) tienen con el sistema de localización y de percepción del robot que el presente proyecto deba satisfacer de manera adecuada:

Requerimientos del consumidor para el sistema de localización:

- Que el sistema tenga poco error.
- Fácil lenguaje de programación
- De bajo precio.
- Sea construido en módulos separables.
- Que llegue a trabajar en diferentes entornos
- Bajo consumo de energía.
- Sea transportable.
- Que se pueda conectar a un equipo para adquirir datos.

Requerimientos del consumidor para el sistema de percepción:

- Poco error en lectura de información.
- Fácil lenguaje de programación.
- Tiempo de respuesta.
- De bajo precio.
- Trabajar en diferentes entornos.

- Bajo consumo de energía.
- Que se pueda conectar fácilmente para adquirir datos.

1.5.2 Requerimientos de diseño Se establecen los requerimientos de diseño para las alternativas basadas en una lluvia de ideas realizada por los autores.

Requerimientos de diseño para el sistema de localización:

- Amplio rango de medida.
- Precisión en la toma de datos.
- Calibración de sensores.
- Arquitectura del código de fácil lectura.
- Económico.
- Sensibilidad a factores externos.
- Tamaño reducido.

Requerimientos de diseño del sistema de percepción:

- Amplio rango de cobertura
- Gran resolución de datos.
- Alta adaptabilidad.
- Infraestructura sensorial simple.
- Arquitectura del código de fácil lectura.
- Económico.
- Tamaño reducido.

2. MARCO TEORICO

2.1 COMPETENCIAS PARA LA NAVEGACION

La navegación o la “cognición” como lo distingue Siegwart[1], representa la habilidad del robot basado en su “conocimiento” y los datos de sus sensores para alcanzar sus posiciones objetivo tan eficiente y confiable como sea posible. En el caso de un robot móvil este aspecto determina la robustez del mismo.

Diferentes enfoques se han planteado para la resolución de los problemas de navegación. La principal diferencia entre estos enfoques de arquitectura es la forma en que estos descomponen el problema en unidades más pequeñas y fáciles de solucionar.

Estos enfoques presentan dos competencias, opuestas conceptualmente, pero que son necesarias para cumplir los objetivos. La planeación y la reacción de los sistemas conjuntos del robot aseguran que el sistema móvil cumpla con su tarea y no se detenga ante la primera complicación.

El enfoque clásico se centra en la competencia de la planeación de las trayectorias la cual mira los problemas de manera estratégica, en donde el sistema decide que deberá hacer en el largo plazo para cumplir con su objetivo. El enfoque reactivo tiene como principal competencia la evasión de obstáculos, la cual debe modular la trayectoria del robot para que este no colisione con los obstáculos sean previamente conocidos o no.

[1] SIEGWART Roland, NOURBAKHSH ILLAH R, SCARAMUZZA Davide, Planning and navigating. En: Introduction to Autonomous Mobile Robots, Second edition, 2011. p.257

Finalmente, se presenta un enfoque hibrido en donde ambas competencias tratan de solucionar las falencias del otro y a pesar de que tengan objetivos diferentes, las dos competencias son necesarias para moverse en entornos complejos, en donde el plan expresado deberá modificarse de acuerdo a los obstáculos que llegue a detectar.

Con el fin de alcanzar el objetivo final, el robot debe modular su funcionamiento con la información que obtiene durante su ejecución. A medida que el avanza, las variables en su entorno pueden cambiar con el tiempo y esto se ve reflejado en la información que es obtenida por los sensores. Aquí es donde se marca la importancia de la reacción en los sistemas de navegación, donde se debe ajustar el comportamiento de los accionadores con el fin de corregir la trayectoria planteada. En un caso ideal, el sistema incorporaría toda la información obtenida en tiempo real.

2.1.1 Competencia para la planeación de trayectorias. El objetivo de la planeación de trayectorias es el de encontrar una ruta en el espacio físico desde la posición inicial hasta la final sin colisionar con los obstáculos. En la robótica móvil, la perspectiva más utilizada para la planeación de trayectorias es la de asumir que el robot es holonómico; esta característica indica que la cantidad de grados de libertad es igual a los grados de libertad sobre los que se tiene control, simplificando así el problema de girar. Esto llega a ser válido para vehículos con dirección diferencial (como el planteado en este proyecto) debido a que los movimientos sobre su propio eje facilitan el cumplimiento de estas rutas.

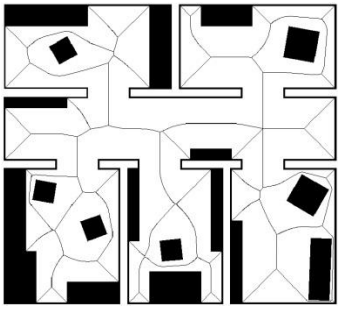
Una de las condiciones principales de esta competencia, es el conocimiento a priori del entorno de trabajo, lo cual requiere un juicio inicial de que el entorno no es dinámico. Por lo cual está sujeto a problemas de incertidumbre sensorial o posicional.

El entorno en donde el sistema de navegación realiza la planificación de la trayectoria se le conoce como espacio de configuración. En este entorno el robot es expuesto en el espacio físico como un punto, el cual reducirá el espacio a una representación en dos dimensiones. Debido a que se ha reducido el robot a un punto, se deberá aumentar las dimensiones de los obstáculos en una proporción igual al mayor radio del robot.

Las principales técnicas para realizar la planeación de trayectorias satisfactoriamente consisten en transformar el modelo continuo del sistema en un mapa discreto que sea amigable con el sistema de navegación. Para esto se identifican tres estrategias para la descomposición del entorno: planeación de rutas, descomposición por celdas y campos potenciales.

2.1.1.1 Planeación de rutas o "road maps". Las rutas o "roadmaps" generan una serie de redes, mediante la unión repetida y de líneas entre nodos los cuales se posicionan estratégicamente alrededor de los obstáculos. La planeación de ruta se reduce a la unión entre el punto inicial y el final mediante la selección de los nodos que le permitan la mayor libertad en el espacio libre a la vez que se minimiza la cantidad de rutas.

Figura 1. Diagrama Voronoi que demuestra las rutas creadas tras el análisis del mapa

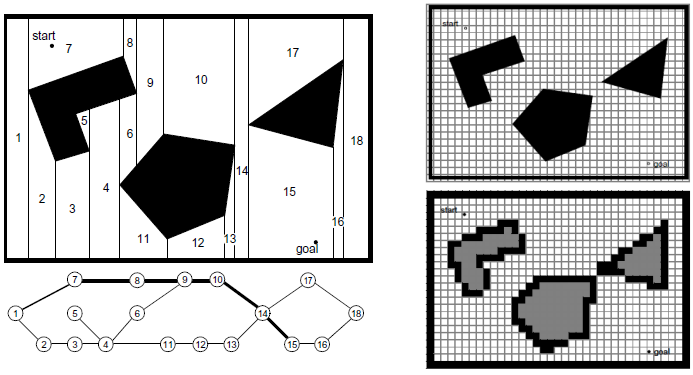


Fuente: Spatial cognition, Universidad de Bremen

La implementación de este tipo de estrategia de descomposición del espacio es bastante simple y su ejecución puede llegar a ser inmediata. Sin embargo, puede llegar a presentar problemas cuando el número de obstáculos crece y con ellos el número de nodos, lo que puede ralentizar el tiempo de ejecución en entornos complejos. Otro de los grandes problemas que tiene esta estrategia, es por lo general, los vértices que unen los nodos pasan muy cerca de los obstáculos, lo que puede implicar problemas en el sistema de percepción.

2.1.1.2 Descomposición por celdas. La descomposición por celdas consiste en discriminar las áreas que están ocupadas por los objetos y separar el mapa en celdas alrededor de los mismos. Posteriormente, se establece que celdas están adyacentes y se realiza una gráfica de conectividad. Luego se analiza que trayectoria une las celdas de tal manera que punto inicial con el final se conecten. Finalmente se busca un algoritmo apropiado que recorra el trayecto encontrado.

Figura 2. Ejemplos de descomposición de celdas a) Descomposición exacta, b) Descomposición aproximada.



Fuente: Computer science, Universidad de Stanford.

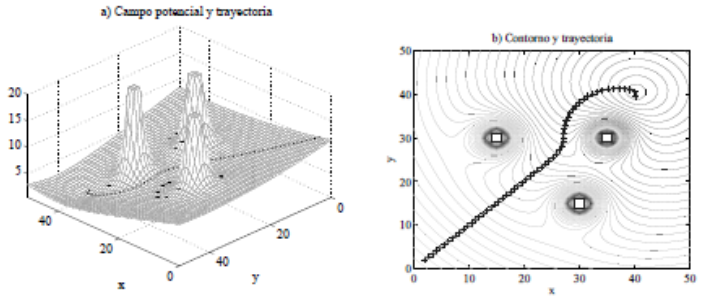
Unos de los aspectos más importantes en la descomposición por celdas de un mapa es el de la localización de las fronteras entre las celdas. Si las fronteras se ubican en función de la geometría del mapa (Figura 2, a) de tal manera que ninguna parte del mapa se pierde, se le llama descomposición exacta. Por el otro lado, si el

proceso transforma el mapa en una aproximación del mismo (Figura 2, b), se le llama descomposición aproximada.

Al igual que la anterior estrategia, el principal problema de esta descomposición exacta radica en el número de obstáculos que puede llegar a tener el mapa, aumentando el número de descomposiciones. Para el caso de la descomposición aproximada, la simplificación del espacio físico a casillas de igual distancia llega a fallar cuando se debe mantener los diferentes nodos en memoria.

2.1.1.3 Campos potenciales. El modelo de campo potencial simplifica al robot como una partícula bajo la influencia de un campo potencial, el cual está determinado por su objetivo y una serie de obstáculos. La única información relevante para el robot es la de su movimiento hacia su objetivo lo que reduce la pérdida de poder computacional.[2]

Figura 3. Planeación de trayectorias por medio de campos potenciales.



Fuente: ESPITIA CUCHANGO, Helbert Eduardo. Trabajo de grado: Propuesta de un algoritmo para la planeación de trayectorias de robots móviles empleando campos potenciales y enjambres de partículas activas brownianas. Universidad Nacional de Colombia, Bogotá. 2011. p 74.

Una de las debilidades que puede llegar a tener este tipo de algoritmos es el de las oscilaciones e inestabilidad de las rutas cuando cruza entre obstáculos en forma de

[2] ROBERTS, Eric. Motion planning in robotics. [En línea]. courses. Standford, CA.: Universidad de Stanford. [citado el 26 de septiembre 2017] Disponible en https://cs.stanford.edu/people/eroberts/courses/soxco/projects/1998-99/robotics/basicmotion.html

pasillo, generando una ruta no deseada para robots móviles. El principal problema que puede llegar a tener este algoritmo es el de los mínimos locales, en donde la partícula queda atorada entre obstáculos.

Este algoritmo es objeto de estudio por uno de los objetivos de este proyecto, por lo tanto, se dará una explicación más detallada de su funcionamiento más adelante en este libro.

2.1.2 Competencia para la evasión de obstáculos. Debido a que los sistemas de navegación basados únicamente en planeación de trayectorias llegaban a ser complejos y además resultaban incompletos para entornos dinámicos, se recurrió a la creación de un sistema basado únicamente en la evasión de obstáculos. Este enfoque está compuesto por procesos menores realizados en orden de importancia en un bucle de percepción, razonamiento y acción.

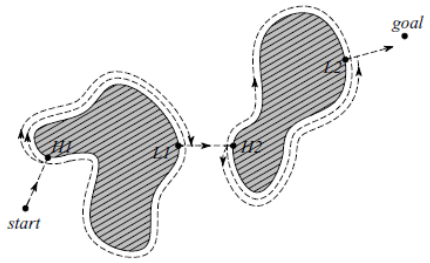
La evasión de obstáculos representa la serie de métodos de reestructuración de la trayectoria de un robot, con el fin de superar los obstáculos inesperados. Esta serie de métodos dependerá completamente de la posición inicial con relación a la trayectoria planteada y los datos obtenidos por los sensores. Hay una gran cantidad de algoritmos para la evasión de obstáculos que van desde la re-planeación de una trayectoria hasta los cambios reactivos en tiempo real; cada uno de estos métodos difiere en la forma en que usan la información obtenida por los sensores y posteriormente las estrategias que proponen para superar los obstáculos.

Un sistema de navegación basado en la competencia de evasión de obstáculos rechaza los modelos aproximados del entorno, lo que lo restringe de una planeación a largo plazo e incluso de la facultad preventiva que tenía el enfoque clásico.

A continuación, se nombrarán los métodos de evasión de obstáculos más utilizados.

2.1.2.1 El algoritmo del insecto (Bug algorithm).[3] El algoritmo del insecto es una de las formas más simples para superar los obstáculos inesperados; dados un punto inicial y un punto final, el objetivo del algoritmo es el de generar una ruta sin colisiones entre los puntos y para esto tiene como principio básico el de rodear los obstáculos.

Figura 4. Ejemplo de una trayectoria generada por el algoritmo del insecto.



Fuente: SIEGWART Roland Bug algorithm. Introduction to autonomous mobile robots.

Uno de los principales problemas de este algoritmo es que no toma en cuenta la cinemática del robot lo que puede ser un impedimento para robots no holonómicos. El ruido obtenido por los sensores puede llegar a ser una debilidad de este tipo de algoritmos.

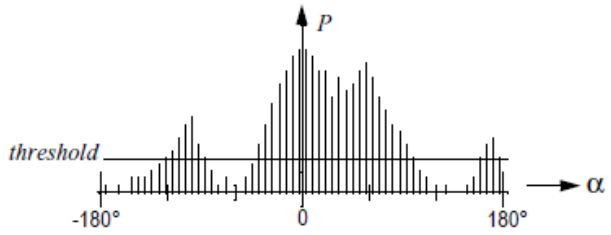
2.1.2.2 Histograma de campo vectorial.[4] El histograma de campo vectorial es un método de evasión de obstáculos que trabaja en tiempo real, además permite la detección de obstáculos desconocidos para evitar colisiones al mismo tiempo que redirección el vehículo hacia el objetivo. Para realizar esto, el algoritmo transforma un espacio bidimensional de la representación de los obstáculos en uno unidimensional, llamado histograma polar. El cual registra la probabilidad de las secciones polares de tener un obstáculo, basado en la información obtenida de los

[3] RIBEIRO, Maria Isabel. Obstacle Avoidance. [En Linea] Instituto de Sistemas e Robótica, Instituto Superior Técnico. Lisboa, Portugal. Universidad de Lisboa. Noviembre de 2005. [citado el 10 de enero de 2018] Disponible en: http://users.isr.ist.utl.pt/~mir/

[4] HAMAD, A. H., & IBRAHIM, F. B. Path Planning of Mobile Robot Based on Modification of Vector Field Histogram using Neuro-Fuzzy Algorithm. 2010. p 129-138.

sensores. Posteriormente calcula el ángulo de giro y la velocidad con la que lo realiza por medio de un proceso de optimización.

Figura 5. Histograma polar.



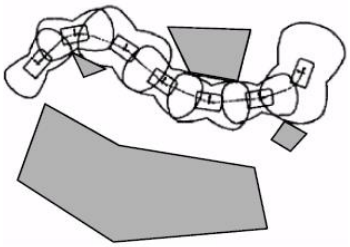
Fuente: SIEGWART Roland. Vector field histogram. Introduction to autonomous mobile robots.

Al igual que el método de navegación por medio de campos potenciales, este algoritmo puede llegar a presentar problemas de mínimos locales, en los que se queda atorado debido a que su representación no identifica un camino correcto. En pasillos estrechos, el algoritmo haría mover al robot oscilatoriamente, rebotando entre las paredes.

2.1.2.3 La técnica de la burbuja.[5] En este algoritmo, la burbuja hace referencia a una fracción de espacio libre alrededor del robot por donde puede transitar en cualquier dirección sin colisionar. El modelo utiliza la información disponible del mapa junto con la información obtenida en los sensores.

[5] SIEGWART Roland, NOURBAKHSH ILLAH R, SCARAMUZZA Davide, Planning and navigating: Obstacle avoidance. En: Introduction to Autonomous Mobile Robots, Second edition, 2011. p.259

Figura 6. Ejemplo de una trayectoria con la respectiva burbuja en cada posición del robot



Fuente: SIEGWART Roland. The bubble band technique. Introduction to autonomous mobile robots.

A pesar de que se asemeje a una estrategia de planeación de trayectoria la propiedad evasiva actúa cuando encuentra valores no conocidos de los obstáculos, es entonces que la burbuja debe arreglar su trayectoria original con el fin de evadir el obstáculo y mantener la burbuja alrededor del robot intacta.

2.2 MODELOS MATEMATICOS.

A continuación, se expondrán los dos modelos matemáticos seleccionados para el estudio de este proyecto. Las arquitecturas aquí presentadas están enfocadas en la competencia de planeación de trayectorias del robot móvil que se planea construir.

2.2.1 Planeación de trayectorias por campos potenciales Para mayor entendimiento se simplificará el modelo a un espacio bidimensional y con sistema de referencia X-Y donde se moverá el robot. Las coordenadas iniciales del robot serán dadas por "$q=(X_1,Y_1)$", las coordenadas del obstáculo serán dadas por "$q_o=(X_o,Y_o)$" y las de la meta serán dadas por "$q_m=(X_m,Y_m)$". La teoría de campos potenciales considerara al robot como una partícula eléctrica en el espacio que es atraída hacía un punto "q_m" mediante un potencial atractivo $U_a(q)$ (atrae al robot a la posición de destino), a su vez es repelida por los obstáculos que se pueden encontrar en el mapa por medio de un potencial repulsivo $U_r(q)$, entonces:

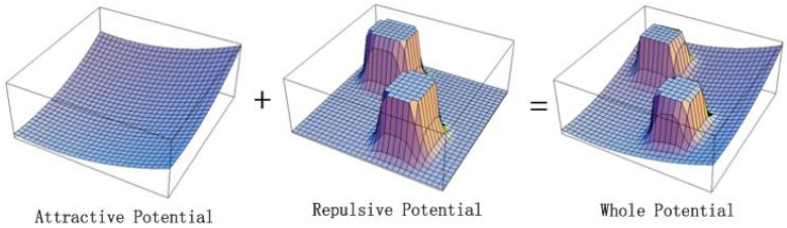
$$U(q) = Ua(q) + Ur(q)$$

El potencial total U(q) es definido por la fuerza resultante F(q) que afecta la partícula:

$$F(q) = -\nabla U(q)$$

La fuerza resultante es definida por dos fuerzas que están interactuando continuamente con la partícula, la fuerza atractivas Fa(q) (proveniente del destino) y las fuerzas repulsivas Fr(q) (proveniente de los obstáculos):

Figura 7. Mapa potencial.



Fuente: WORDPRESS Collision free path usig potential field method for highly redundant manipulators [en linea] disponible en: https://taylorwang.wordpress.com/2012/04/06/collision-free-path-planning-using-potential-field-method-for-highly-redundant-manipulators/

Como se aprecia en la figura 7, la unión del potencial atractivo y repulsivo genera un mapa tridimensional en caída con dirección a la meta, mostrando la zona de mayor potencial (esquina superior derecha) la cual está más alejado de la meta y la zona de menor potencial (esquina inferior izquierda) el cual es el destino de la partícula. A medida que la partícula avanza en dirección al destino, la fuerza resultante tendera a disminuir hasta tener un valor cero, mostrando que se encuentra en la zona de menor potencial (meta).

$$F(q) = Fa(q) + Fr(q)$$

- Potencial atractivo $U_a(q)$: El potencial atractivo se puede expresar con la siguiente ecuación:

$$Ua(q) = \frac{1}{2}\, K_a\, \rho^n$$

Donde K_a es un factor atractivo positivo, ρ es la distancia euclidiana entre la ubicación del robot respecto a la meta $\|q - q_m\|$ y n puede ser 1 si se desea la forma del campo potencial cónico y 2 si se desea la forma del campo potencial cuadrático.

Para un mejor uso se selecciona n=2, debido a que la fuerza atractiva converge linealmente conforme el robot se aproxima hacia q_m, es decir, cuando el robot está cerca de la meta, este se aproxima lentamente a esta.[6]

$$Fa(q) = -\nabla Ua(q)$$

$$Fa(q) = -\frac{1}{2} ka * \rho^2 * \nabla\rho(q)$$

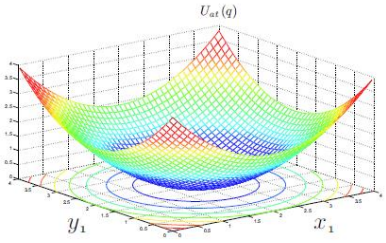
$$Fa(q) = -\frac{1}{2} Ka \left(\frac{\partial^2(q, qm)}{\partial_{x1}}\, \frac{\partial^2(q, qm)}{\partial_{y1}}\right)$$

$$Fa(q) = -Ka[(X_1, X_m)(Y_1, Y_m)]$$

$$Fa(q) = -Ka * (q - qm)$$

[6] CORDERO, Martín. Control de un robot móvil de ruedas mediante campos potenciales artificiales y procesamiento digital de imágenes en la evasión de obstáculos. México: Instituto Politécnico Nacional. Facultad de Ingeniería. 2011. p. 63

Figura 8. Potencial atractivo cuadrático.



Fuente: GARCÍA, José. Diseño y construcción de un robot móvil aplicando el método de campos potenciales en la evasión de obstáculos. México, Instituto Politécnico Nacional. Facultad de Ingeniería. 2008. p. 66.

- Potencial repulsivo Ur(q): El potencial repulsivo de puede explicar con la siguiente ecuación:

$$Ur(q) = \frac{1}{2} Kr \left(\frac{1}{\rho(q)} - \frac{1}{\rho_0} \right)^2$$

$$Fr = -\nabla Ur(q)$$

Donde Kr es un factor repulsivo positivo y $\rho(q)$ es la distancia euclidiana entre la ubicación de la partícula respecto al obstáculo ll q – q_0 ll.

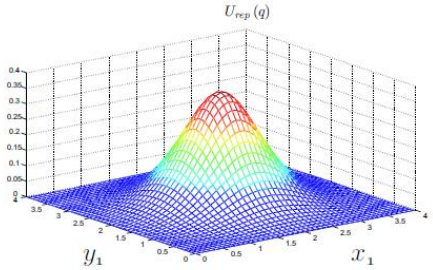
El potencial repulsivo solo afecta la partícula cuando $\rho(q) \leq \rho_0$ debido a que ρ_0 es una distancia perpendicular a la superficie del obstáculo que representa el área de efecto dentro de la cual la acción repulsiva del obstáculo influirá sobre la partícula; al momento en que la resultante entre (q – q_0) de un valor mayor a ρ_0 la partícula se encontrara demasiado lejos, por lo cual el obstáculo no afectara el trayecto de la partícula dando un $Ur(q)$ = 0.

Desarrollando la anterior ecuación obtenemos:

$$Fr(q) = Kr \left[\frac{1}{\rho(q)} - \frac{1}{\rho_0} \right] \frac{1}{\rho^2(q, qo)} \nabla \rho(q)$$

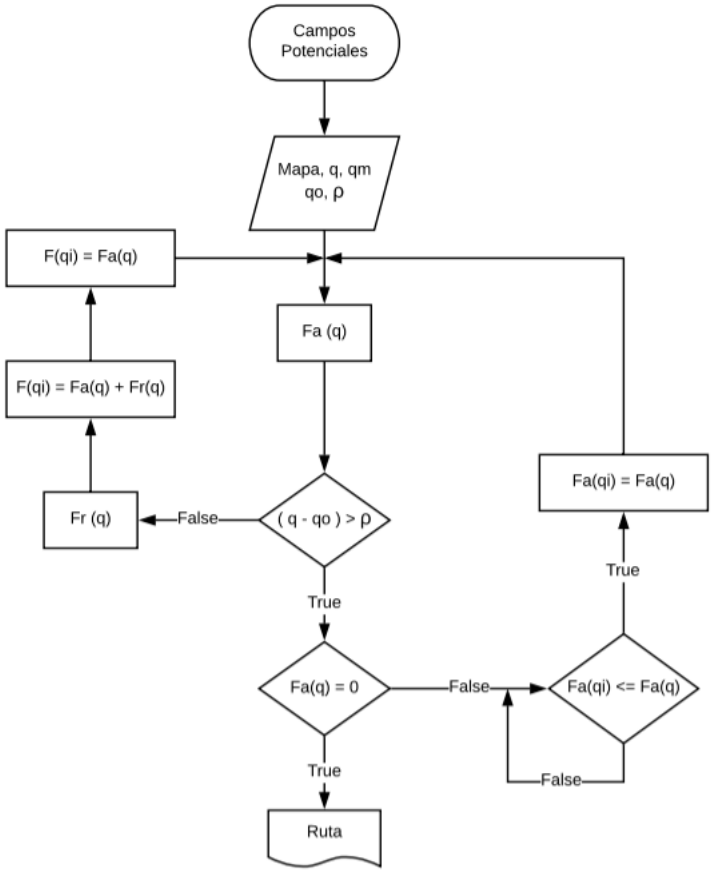
$$Fr(q) = Kr\left[\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right]\frac{(q - qo)}{\rho^3(q, qo)}$$

Figura 9. Potencial repulsivo.



Fuente: GARCÍA, José. Diseño y construcción de un robot móvil aplicando el método de campos potenciales en la evasión de obstáculos. México, Instituto Politécnico Nacional. Facultad de Ingeniería. 2008. p. 67

Figura 10. Diagrama de flujo para el algoritmo Campos Potenciales



Ventajas:

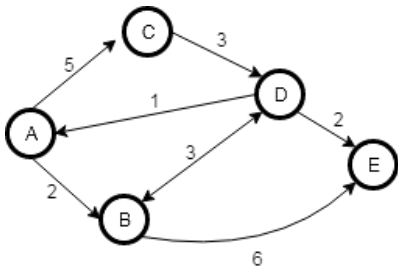
- Las fuerzas repulsivas se pueden implementar en tiempo real en un sistema robótico móvil.
- Fácil escalado a sistemas multidimensionales e implementación a diferentes tipos de robots.

Desventajas:

- Se presentan mínimos locales ya sean por un obstáculo o varios. En donde la partícula tendrá un movimiento errático.
- Se pueden generar movimientos oscilatorios al atravesar espacios estrechos.

2.2.2 Planeación de trayectorias por algoritmo de Dijsktra El algoritmo de Dijkstra es uno de los métodos más utilizados para hallar el camino más corto en un grafo cuyos vértices o uniones poseen un costo condicional. Dado un vértice inicial deberá encontrar el camino más corto para cualquier otro vértice incluido el vértice objetivo.

Figura 11. Grafo con sus respectivos vértices y el peso de las aristas.



El algoritmo de Dijkstra fue demostrado por primera vez en ejecución, por su creador Edsger Dijkstra en 1956, cuando fue utilizado para demostrar las capacidades de una nueva computadora llamada ARMAC. El objetivo era ofrecer un problema con su respectiva solución, cuyo único requerimiento fuera de que personas con pocos conocimientos sobre computación pudieran entenderlo. El algoritmo fue

implementado en un mapa de transportación entre 64 rutas de bus de los Países Bajos para encontrar el camino más corto entre Rotterdam y Groningen.[7]

Este algoritmo es ampliamente utilizado en diferentes procesos que requieran hallar rutas de mínimo costo entre dos entes. Algunos de estos son el protocolo de red OSPF (Open Shortest Path First)[8] y el protocolo IS-IS (Intermediate system to intermediate system)[9], los cuales calculan una ruta idónea entre dos routers, teniendo en cuenta variables de costo tales como ancho de banda y congestión del enlace. Uno de los usos más importantes que se le atribuye a este algoritmo es el de planeación de rutas en Google Maps[10] en donde los costos entre los vértices se dan por la distancia entre ellos y el flujo vehicular (Figura 12).

Figura 12. Representación gráfica de vértices y aristas con sus respectivos costos en un mapa de Google Maps

Fuente: KOSSAKOWSKI, Marcin. Finding shortest path using Dijkstra's algorithm and weighed directed graph. [Base de datos en línea]. Noviembre 3 de 2014. [citado el 10 enero 2018]. Disponible en http://www.marcinkossakowski.com/finding-shortest-path-using-dijkstras-algorithm/

[7] MISA, Thomas J.. An interview with Edsger W. Dijkstra. [Base de datos en línea]. Agosto de 2010. Revista Communications of the ACM, Vol. 53 (No.8), 41-47. [citado el 10 enero 2018]. Disponible en https://dl.acm.org/citation.cfm?doid=1787234.1787249 .

[8] MOY, J.. OSPF protocol analysis. [Base de datos en línea]. Julio de 1991. Network Working Group. [citado el 10 enero 2018]. Disponible en https://tools.ietf.org/pdf/rfc1245.pdf.

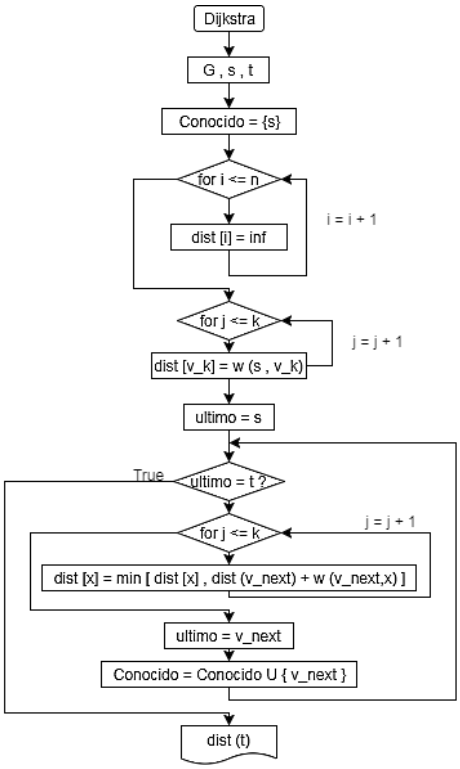
[9] HAWKINSON, J.. Guidelines for creation, selection and registration of an Autonomous System. [Base de datos en línea]. Marzo de 1996. Network Working Group. [citado el 10 enero 2018]. Disponible en https://tools.ietf.org/pdf/rfc1930.pdf.

[10] LANNING, Daniel; HARRELL, Gregory; WANG, Jin. Dijkstra's algorithm and google maps. [Base de datos en línea]. Marzo 28 de 2014. Revista Communications of the ACM, Articulo No. 30. [citado el 10 enero 2018]. Disponible en https://dl.acm.org/citation.cfm?doid=2638404.2638494 .

2.2.2.1 Modelo matemático de Dijkstra. Para entender el algoritmo de Dijkstra se debe empezar por la suposición de que el camino más corto entre un vértice s y un vértice t en un mapa de grafos G pasa por un vértice intermedio x. Evidentemente, esta ruta deberá tener el camino más corto entre s y x, por lo tanto, se debe encontrar primero la mínima distancia entre s y x antes de encontrar la ruta de s a t.

El algoritmo actúa en una serie de turnos, en la que cada una establece la mínima distancia entre s y un nuevo vértice. Para cada turno, x sería el vector que minimice $dist(s, v_i) + w(v_i, x)$ para todos los vértices no visitados $1 \leq i \leq n$, en donde $w(i, j)$ es el costo de movimiento del vértice i al j y $dist(i, j)$ es el costo de menor distancia entre ellos[11].

Figura 13. Diagrama de flujo para el algoritmo Dijkstra.



[11] SKIENA, Steven. The Algorithm Design Manual. New York, USA: Springer, 2008. p.206.(Dijkstra's Algorithm). ISBN 978-1-84800-070-4

El algoritmo Dijkstra funciona de manera adecuada solamente en grafos que no contengan valores negativos, esto se debe a que en el caso de encontrarse con una arista negativa cambiara la ruta de menor distancia de cualquier otro vértice que ya haya cruzado, haciendo esta arista una ruta de mínima distancia falsa a otras aristas. La mayoría de las aplicaciones no contienen grafos con aristas negativos, por lo cual el problema permanece teórico. El algoritmo sugerido en esta representación tiene una complejidad $O(n)^2$. Esta propiedad hace referencia al número de pasos al que tiende el algoritmo y se caracteriza por el número de bucles que repiten la función principal ($dist(s, v_i) + w(v_i, x)$).

Ventajas:

- La implementación de este algoritmo en un sistema de planeación de rutas resulta simple debido a su baja complejidad.
- Analiza varios vértices en cada iteración. Lo que resultaría adecuado para rutas con objetivos múltiples.

Desventajas:

- Es imposible trabajar con valores negativos. Esto resultaría en la eliminación de nodos analizados previamente y rutas incorrectas.

2.3 ESTADO DEL ARTE

La robótica ha tomado un gran lugar en las actividades realizadas por el ser humano, donde antes se dificultaba o era imposible realizar una actividad, ahora ya es una realidad poder realizarlas. El primer paso en la creación de robots comienza en 1948 por Argonne Nacional Laboratory, donde crearon el primer manipulador de elementos radiactivos sin riesgo para el operario, el cual manejaba un operador maestro detrás de un grueso cristal para realizar el levantamiento de sustancias

toxicas. Este nuevo sistema de telemanipuladores tomo un gran auge por industria submarina y en los años setenta la industria espacial se sumó a este interés.

En el año 1954 fue construido y patentado el primer dispositivo robótico, que posteriormente seria implementado por primera vez en General Motors de Trenton, Nueva Jersey en 1960. Era un robot automatizado de fundición a presión, su función era dejar caer las manijas y piezas similares del automóvil que se encontraban al rojo vivo en una piscina de líquido refrigerante, las cuales eran trasladadas a los trabajadores para recortarlas y pulirlas. La gran característica era su agarre de acero, el cual eliminaba la necesidad del hombre tener que tocar las piezas recién hechas en acero fundido.

Este acontecimiento dio a un nuevo mundo de implementación de robots manipuladores en la industria y a robots móviles capaces de realizar tareas en diferentes entornos, más conocidos como robots de servicio, aunque estos últimos es muy escasa su implementación se encuentra en desarrollo, generando de esta forma que sea un gran sector de investigación.

Figura 14. Primer robot industrial.

Fuente: TIMETOAST [en línea] disponible en: https://www.timetoast.com/timelines/1720199.

Un robot móvil puede navegar de forma guiada o de forma autónoma, aunque el concepto de autonomía es gradual, debido a que el robot puede seguir una trayectoria en un lugar determinado sin ninguna variante (obstáculos dinámicos),

hasta poder analizar en tiempo real todo a su alrededor y reaccionar a los cambios inesperados generando la cualidad fundamental de todo robot móvil autónomo.

En la actualidad podemos clasificarlos en tres generaciones:

- Primera Generación: Por lo general son brazos mecánicos y solo pueden realizar movimientos repetitivos con alta precisión.
- Segunda Generación: Estos robots poseen sensores los cuales le proporcionan información del entorno, generando de esta forma toma de limitadas decisiones para reaccionar ante algunas de las situaciones que se le pueden presentar.
- Tercera Generación: Estos robots emplean inteligencia artificial, capaces de realizar razonamientos lógicos e incluso aprender después de realizar varias iteraciones y repeticiones.

El proyecto “reconstrucción de mapas utilizando escaneo láser para la navegación de un robot móvil[12]” se basa en un robot móvil de tres ruedas con varios sensores infrarrojos ubicados alrededor de él, la función de estos sensores es medir la distancia de los objetos alrededor del robot por medio de extracción de rectas, este sistema consta en analizar la distancia recorrida por la luz de cada sensor infrarrojo y guardarlo en un mapa cartesiano, generando de esta manera un mapa del entorno donde va a trabajar, sin ayuda de imágenes o información externa.

[12] LÓPEZ ORTEGA, Jorge. Reconstrucción de mapas utilizando escaneo láser para la navegación de un robot móvil. México, 2016, 144h. Trabajo de maestría (Tecnología en Cómputo). Instituto Politécnico Nacional. Centro de Innovación y Desarrollo Tecnológico en Cómputo.

Figura 15. Mapa generado por medio de escaneo laser.

Fuente: LÓPEZ ORTEGA, Jorge. Reconstrucción de mapas utilizando escaneo láser para la navegación de un robot móvil. México, 2016, 144h. Trabajo de maestría (Tecnología en Cómputo). Instituto Politécnico Nacional. Centro de Innovación y Desarrollo Tecnológico en Cómputo. p 110.

2.3.1 Robots móviles en la agricultura. En los últimos 50 años la industria agrícola ha tenido un gran crecimiento gracias a la automatización e implementación de robots manipuladores encargados de la recolección, fumigación, poda, etc., pero estos posen la desventaja de ser transportados por un vehículo, ya sea con conducción manual o posee cierto grado de autonomía ideado para una determinada aplicación. El proyecto realizado por la empresa Vision Robotics Corporation denominado robot recolector de naranjas[13], muestra un vehículo con ocho brazos robóticos equipados cada uno con cámaras estereoscópicas, las cuales se encargan de crear una imagen 3D de todo el árbol especificando la ubicación y tamaño exacto de las naranjas, proporcionando coordenadas X, Y y Z respecto a la ubicación de cada brazo para el correcto desplazamiento y recolección del fruto, aunque el sistema de recolección es autónomo el movimiento del vehículo sigue siendo controlado por un operador.

[13] VISION ROBOTICS CORPORATION. Robot recolector de naranjas (Producto). San Diego. 1999

Figura 16. Robot recolector de naranjas.

Fuente: Vision Robotics Corporation.

También se mencionar otro proyecto encargado de recolección, como es el robot recolector de tomates[14] realizado por la universidad de Okasuma, los autores de este proyecto lo dividieron en dos sistemas: sistema de navegación y sistema de recolección.

El sistema de navegación se basa en conocimientos del área de trabajo (cantidad de filas, distancia entre plantas, distancia entre filas y longitud de las filas) y sensores de proximidad, los cuales se encargan de mantener al robot a una distancia prudente respecto a las filas y a su vez generar un movimiento casi uniforme entre ellas al momento de desplazarse por medio de intervalos de planta a planta.

El sistema de recolección se encuentra conformada por cámaras CCD, las cuales por medio de procesamiento de imagen se detecta el fruto diferenciándolo de otros elementos (ramas, hojas) y proporcionando a su vez la ubicación para poder ser recolectado por medio de un brazo mecánico de 7 grados de libertad.

Es por ello por lo que se están buscando sistemas más autónomos, donde se busca reemplazar la intervención del hombre, pasando de operario a teleoperador.

[14] KONDO, N., M. MONTA, T. FUJIJURA, Y. SHIBANO Y K. MOHRI. Robot recolector de tomates. EN: Proceedings of the Asian Control Conference Conferenca 1-4. 1994. Japon.

Figura 17. Robot recolector de tomates.

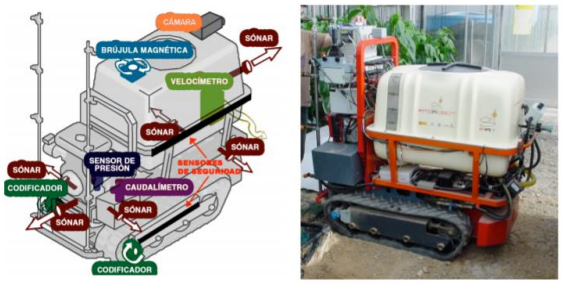
Fuente: Robots recolectores. Universidad de Okasuma.

Para solucionar el problema de los robots manipuladores, se comenzó a implementar sistemas de navegación con el fin de proporcionar mayor autonomía a la hora de tomar decisiones ante imprevistos y poder planear rutas asegurando un buen desplazamiento en un terreno determinado, reemplazando la intervención del conductor del agricultor a suministrador de información necesaria (dimensiones del terreno, tipo de desplazamiento, puntos maestros para el desplazamiento y punto final).

Estos robots autónomos en la agricultura son muy escasos debido a los grandes problemas que se pueden encontrar como: inconsistencia del terreno, distintos tipos de suelo, el cultivo se presenta de forma irregular, condiciones de ambiente hostiles (humedad, temperatura, lluvias) y obstáculos imprevistos. Estos vehículos se encuentran ayudados por un sistema de guiado, que tiene como objetivo el seguimiento de una trayectoria definida por un punto inicial y final, la cual fue calculada previamente o es dirigida por la percepción de marcas locales (sensores, brújulas, GPS o cámaras de video) a lo largo del cultivo.

El proyecto Sistema de visión de apoyo a la navegación de un robot móvil en invernaderos[15] realizado en la Universidad de Almería, muestra la construcción y diseño de un vehículo autónomo el cual se encarga de fumigar por medio de dos sistemas de navegación que trabajan mutuamente.

Figura 18. Robot móvil en invernaderos.

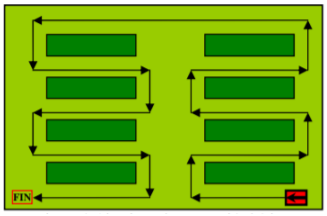


Fuente: MORENO, José, RODRÍGUEZ, Francisco and GONZALEZ, Richard. Sistema de visión de apoyo a la navegación de un robot móvil en invernaderos. Almería, 2014, 8h. Universidad de Almería. p 4.

Para poder comenzar con su respectivo trabajo de rocío, primero se ingresa el mapa donde se ve la estructura del invernadero y por medio del sistema de navegación Diagrama de Voronoi se caculo el punto medio geométricamente entre los pasillos evitando la colisión, después se es utilizado el método de búsqueda de grafos, similar al de Dijsktra, el cual se encarga de la trayectoria mínima que se debe realizar para no dejar de pulverizar ninguna fila de plantas. Definida la ruta y con ayuda de los sensores asegurando que se mantenga una distancia prudente entre las filas, procedieron a diseñar un sistema de navegación por visión, el cual debe suplir la necesidad de detectar obstáculos y generar nuevas rutas cuando el mapa es modificado o cuando los sensores no pueden detectar las filas, es por ellos que realizan una discriminación entre el suelo y el cultivo por medio de procesamiento de imagen, en la que finalmente se determina el centro de la imagen segmentada produciendo una nueva trayectoria.

[15] MORENO, José, RODRÍGUEZ, Francisco and GONZALEZ, Richard. Sistema de visión de apoyo a la navegación de un robot móvil en invernaderos. Almería, 2014, 8h. Universidad de Almería.

Figura 19. Navegación por Diagrama de Voronoi.



Fuente: MORENO, José, RODRÍGUEZ, Francisco and GONZALEZ, Richard. Sistema de visión de apoyo a la navegación de un robot móvil en invernaderos. Almería, 2014, 8h. Universidad de Almería p 191.

Figura 20. Navegación por visión.

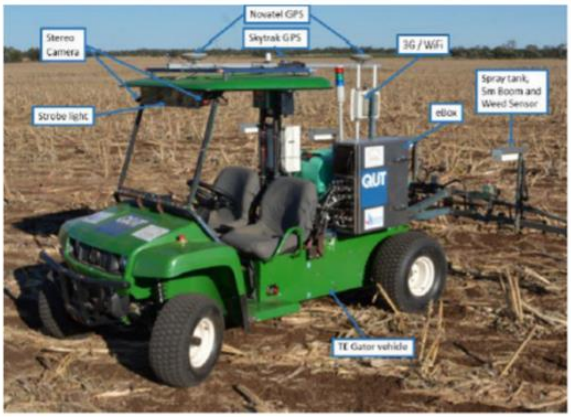
Fuente: MORENO, José, RODRÍGUEZ, Francisco and GONZALEZ, Richard. Sistema de visión de apoyo a la navegación de un robot móvil en invernaderos. Almería, 2014, 8h. Universidad de Almería. p 7.

Otro proyecto implementado en el campo agrícola fue realizado por Robotics for Sustainable Broad-Acre[16], donde implementaron un robot maestro y 12 robots esclavos los cuales se les asigna un área determinada del campo a trabajar. Ya ubicado dentro del terreno comienza a crear una serie de puntos que representan la superficie de un objeto por medio de distancias obtenidas a través de sensores laser y cámara, para poder detectar algún obstáculo dentro de la ruta trazada por el agricultor (perímetro del terreno, latitud y longitud) con ayuda de un GPS instalado en el robot. Para permitir el avance se crea una lista de nodos maestros en toda la ruta y a su vez un embudo que representa el área de cubrimiento de los sensores, asegurando que, aunque el robot evite un obstáculo se mantenga en la trayectoria

[16] REALE, Federico, PATIÑO, Charles and ARRIETA, Fernando. Estudio del estado del arte navegación en robots agrícolas. Montevideo, 2014, 55h. Trabajo de grado (Ingeniería en Computación). Universidad de la República. Facultad de Computación. Instituto de Computación.

planeada. Para la localización implementaron un filtro de partículas que combinan la información suministrada por sensores laser, GPS, IMU y encoders comparando de esta forma la posición del robot en cada momento con la información ya conocida.

Figura 21. Robot cosechador por Robotics for Sustainable Broad-Acre



Fuente: REALE, Federico, PATIÑO, Charles and ARRIETA, Fernando. Estudio del estado del arte navegación en robots agrícolas. Montevideo, 2014, 55h. Trabajo de grado (Ingeniería en Computación). Universidad de la República. Facultad de Computación. Instituto de Computación.p 33.

Figura 22. Sistema de detección de obstáculos y áreas de cubrimiento de los sensores.

Fuente: REALE, Federico, PATIÑO, Charles and ARRIETA, Fernando. Estudio del estado del arte navegación en robots agrícolas. Montevideo, 2014, 55h. Trabajo de grado (Ingeniería en Computación). Universidad de la República. Facultad de Computación. Instituto de Computación.p 38.

3. DISEÑO Y SELECCIÓN SISTEMA DE NAVEGACIÓN

En este capítulo se describe y desarrolla el proceso de diseño de las simulaciones de los modelos matemáticos para planeación de trayectorias. Posteriormente se realiza la selección del modelo matemático más adecuado para ser implementado en el robot móvil.

3.1 DISEÑO

En este inciso se hablará de los modelos matemáticos y de cómo fueron adecuados para el tipo de problema que se está solucionando en este proyecto. Además, se nombrarán los diferentes parámetros requeridos para su adecuada representación gráfica.

Los modelos matemáticos serán representados por el entorno computacional Matlab. Matlab es una herramienta interactiva basada en matrices para cálculos científicos y de ingeniería[17]. Posee un lenguaje propio (Lenguaje M) de alto desempeño en cálculos matemáticos, modelamiento de sistemas (lineales, discretos y no lineales), desarrollo de algoritmos y procesamiento de imágenes, proporcionando los resultados en gráficas y en notaciones matemáticas establecidas por el usuario.

3.1.1. Campos Potenciales. El desarrollo del código de Campos Potenciales se divide de la siguiente manera:

- Variables de entrada: terreno, punto inicial y punto final.

[17] MATLAB. Que es Mat Lab [en linea]. (2015). [Consultado: 10 de nomviembre de 2017]. Disponible en Internet: https://juancarlosusomatlab2015.weebly.com/definicion-matlab.html

- Análisis de vecinos.
- Mínimos locales.

3.1.1.1 Variables de entrada. En la primera etapa del algoritmo se requerirán las dimensiones del mapa, la posición de los obstáculos y los puntos de inicio y de fin. A partir de esto se hará la debida transformación del mapa para el algoritmo. A cada coordenada del mapa se le asignara un valor que representa su valor potencial, estos valores serán basados en los puntos de inicio y de fin. Valores más altos indican un potencial repulsivo mayor mientras que los más bajos representan un potencial de tipo atractivo.

Los valores potenciales de cada coordenada serán almacenados en una matriz binaria con las dimensiones correspondientes a la resolución deseada del mapa, en el caso de este modelo la resolución es de 1 centímetro y los valores potenciales serán almacenados en una matriz de 500x500

A continuación, se indican los parámetros de asignación de estos valores:

- Punto inicial: La coordenada de inicio se le asignará el mayor potencial posible por las dimensiones del mapa, debido a que el robot deberá alejarse de este punto. El valor asignado estará dado por la siguiente ecuación:

$$Punto\ inicial = \sqrt{(Filas^2 + columnas^2)}$$

$$Punto\ inicial = \sqrt{(500^2 + 500^2)}$$

$$Punto\ inicial = 707$$

Para una mejor visualización se decidió asignarle el valor de “700”.

Figura 23. Ejemplo del mapa inicial en una matriz 23x23, con robot valor a 33.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	33	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- Punto final: La coordenada final deberá tener el mayor potencial atractivo del mapa con el fin de que la ruta converja hacia él. Se le asignara el menor valor posible (0). Al contrario de la coordenada inicial este valor no depende de la geometría del mapa.

- Obstáculos: La representación inicial de los obstáculos tienen como valor la unidad binaria "1", para la correcta realización del algoritmo de campos potenciales se deben realizar algunos ajustes. Primero se adjudica el potencial repulsivo a los obstáculos, este depende de cuanta distancia se desea alejar al robot respecto a los obstáculos; finalmente se cambia el valor "1" por un valor mayor al punto de inicio y a cualquier otro punto dentro del mapa.

Figura 24. Representación campo potencial repulsivo.

0	0	800	800	0	0
0	0	800	800	0	0
0	0	800	800	0	0
0	0	800	800	0	0
0	0	800	800	0	0
0	0	800	800	0	0
0	0	800	800	0	0

- Mapa: Una vez asignados los valores potenciales de las coordenadas principales, se procederá a realizar el campo potencial alrededor y en base a estas coordenadas. Los valores más cercanos a la coordenada inicial tendrán un potencial repulsivo mayor mientras que las coordenadas cercanas al valor final tendrán un valor atractivo más alto. La asignación de este valor potencial está dada por la ecuación:

$$Potencial\ Atractivo = \sqrt{((Xp, Yp)^2 + (Xf, Yf)^2)}$$

Figura 25. Representación campo potencial atractivo.

3	3	3	3	3	3	3
3	2	2	2	2	2	3
3	2	1	1	1	2	3
3	2	1	0	1	2	3
3	2	1	1	1	2	3
3	2	2	2	2	2	3
3	3	3	3	3	3	3

Una vez realizado el anterior proceso de transformación del mapa se obtendrá una matriz de valores potenciales de este tipo:

Figura 26. Ejemplo del mapa de campos potenciales en una matriz 23x23.

21	20	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19
21	20	19	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18
21	20	19	18	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17
21	20	19	800	800	16	16	16	800	800	16	16	16	800	800	16	16	16	800	800	16	16	16
21	20	19	800	800	16	15	15	800	800	15	15	15	800	800	15	15	15	800	800	15	15	15
21	33	19	800	800	16	15	14	800	800	14	14	14	800	800	14	14	14	800	800	14	14	14
21	20	19	800	800	16	15	14	800	800	13	13	13	800	800	13	13	13	800	800	13	13	13
21	20	19	800	800	16	15	14	800	800	12	12	12	800	800	12	12	12	800	800	12	12	12
21	20	19	800	800	16	15	14	800	800	11	11	11	800	800	11	11	11	800	800	11	11	11
21	20	19	800	800	16	15	14	800	800	11	10	10	800	800	10	10	10	800	800	10	10	10
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	9	9	9	800	800	9	9	9
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	8	8	8	800	800	8	8	8
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	7	7	7	800	800	7	7	7
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	6	6	800	800	6	6	6
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	5	800	800	5	5	5
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	4	4	4
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	3	3	3
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	2	2	2
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	1	1	1
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	1	0	1
21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	1	1
21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	2	2	2
21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	3	3	3	3

3.1.1.2 Análisis de vecinos. Una vez definida el área de trabajo se procede a determinar la ruta con menor potencial en el mapa, para lograrlo es necesario analizar los ocho puntos adyacentes al robot.

Figura 27. Representación de los 8 puntos adyacentes al robot.

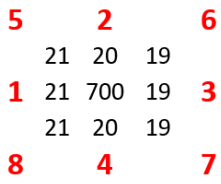


Figura 28. Ubicación espacial de los 8 puntos adyacentes respecto al robot.

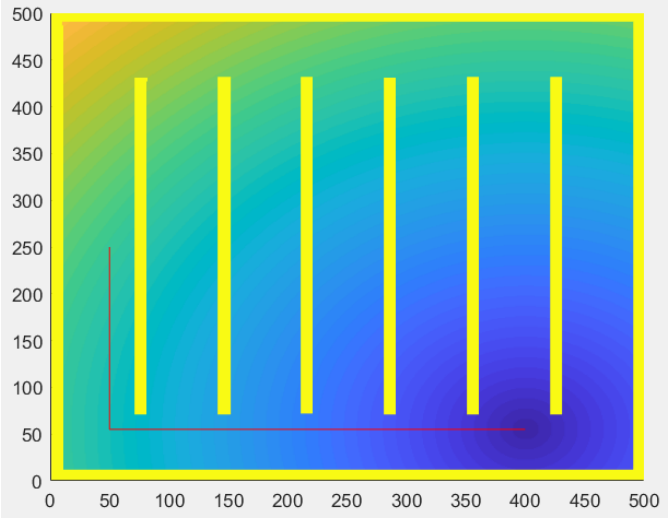
```
rr = 1/sqrt(2);
s1 = [-1 0];
s2 = [0 1];
s3 = [1 0];
s4 = [0 -1];
s5 = [-rr rr];
s6 = [rr rr];
s7 = [rr -rr];
s8 = [-rr -rr];
sv = [s1;s2;s3;s4;s5;s6;s7;s8];
```

Una vez determinada la casilla de menor valor respecto al valor actual, el robot se moverá a esta casilla, este proceso se realizará continuamente hasta llegar al objetivo con el valor “0”.

Figura 29. Ejemplo de ruta realizada por campos potenciales en una matriz 23x23.

21	20	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19
21	20	19	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18	18
21	20	19	18	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17	17
21	20	19	800	800	16	16	16	800	800	16	16	16	800	800	16	16	16	800	800	16	16
21	20	19	800	800	16	15	15	800	800	15	15	15	800	800	15	15	15	800	800	15	15
21	33	19	800	800	16	15	14	800	800	14	14	14	800	800	14	14	14	800	800	14	14
21	20	19	800	800	16	15	14	800	800	13	13	13	800	800	13	13	13	800	800	13	13
21	20	19	800	800	16	15	14	800	800	12	12	12	800	800	12	12	12	800	800	12	12
21	20	19	800	800	16	15	14	800	800	11	11	11	800	800	11	11	11	800	800	11	11
21	20	19	800	800	16	15	14	800	800	11	10	10	800	800	10	10	10	800	800	10	10
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	9	9	9	800	800	9	9
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	8	8	8	800	800	8	8
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	7	7	7	800	800	7	7
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	6	6	800	800	6	6
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	5	800	800	5	5
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	4	4
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	3	3
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	2	2
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	1	1
21	20	19	800	800	16	15	14	800	800	11	10	9	800	800	6	5	4	800	800	1	0
21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	1
21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	2	2
21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	3	3	3

Figura 30. Simulación de ruta realizada por campos potenciales en una matriz 500x500.

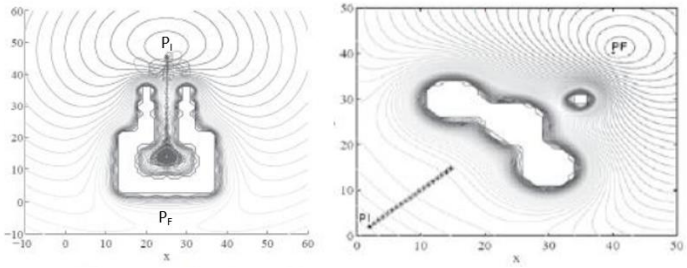


3.1.1.3 Mínimos locales. Uno de los grandes problemas que presente este modelo matemático es el de los mínimos locales, estos se pueden presentar debido a la geometría del mapa, en donde el potencial repulsivo y atractivo es igual en cualquier

dirección, lo que generaría un estancamiento de la partícula en el obstáculo antes de que pueda llegar a su objetivo.

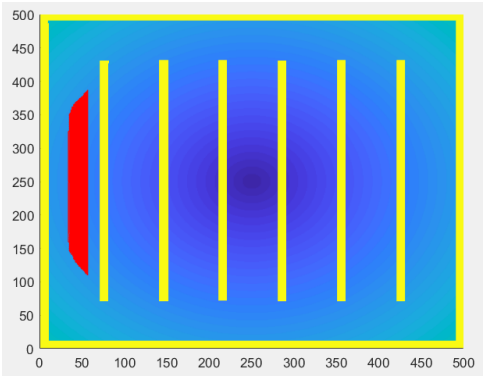
Los obstáculos que se pueden encontrar en el mapa pueden tener formas cóncavas que generaran un estancamiento en un punto diferente al objetivo como se puede apreciar en la figura# a) o el obstáculo posee una longitud extensa y a la vez se encuentra perpendicular entre la trayectoria del robot y la meta como se aprecia en la figura## b)

Figura 31. Mínimos locales. a) Obstáculo cóncavo, b) Obstáculo de longitud extensa.



Fuente: ESPITIA, Helbert y SOFRONY, Jorge. Algoritmo para planear trayectorias por robots móviles, empleando campos potenciales y enjambre de partículas activas brownianas. En: Ciencia e ingeniería neogranadina. No 22-2 (dic., 2012); p. 75-96. ISSN 0124-8170.p. 86

Figura 32. Mínimo local producido en la simulación.



Para solucionar este problema se diseñó un código el cual contiene dos procesos:

- Opuestos: Cuando el robot llego a una zona donde ninguno de sus 8 vecinos posee un potencial menor al actual, se determina cual fue el último desplazamiento realizado y se seleccionan las cinco casillas adyacentes opuestas a este.

Figura 33. Casillas posibles a la anterior posición del robot.

```
% Opuestos
o1 = [2 3 4 6 7];
o2 = [3 4 1 7 8];
o3 = [2 4 1 5 8];
o4 = [1 2 3 5 6];
o5 = [6 7 8 3 4];
o6 = [7 8 5 4 1];
o7 = [6 8 1 5 2];
o8 = [5 2 6 3 7];
%ov = [o1 o2 o3 o4 o5 o6 o7 o8];
ov = [o3;o4;o1;o2;o7;o8;o5;o6];
```

- Resultante: Al tener las casillas opuestas se procede a sumar sus magnitudes y direcciones para encontrar la zona de mayor potencial (línea roja). Además, se procede a encontrar la dirección y magnitud que posee el robot respecto al punto final (línea azul).

  Tomando los dos vectores se procede a realizar una resta entre ellos dando como resultado la nueva dirección del robot evitando el mínimo encontrado (línea verde).

Figura 34. Suma de vectores para solucionar mínimos locales.

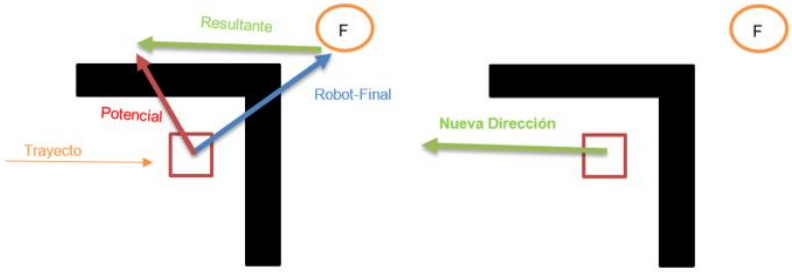
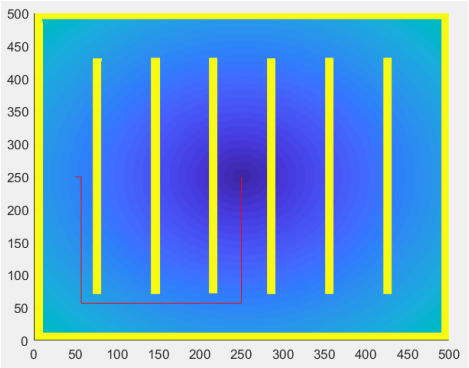


Figura 35. Mínimo local solucionado en simulación.



3.1.2 Dijsktra. La técnica seleccionada para la representación del espacio de configuración fue el de descomposición por celdas, lo cual tiene como fin aumentar la resolución en la solución obtenida. A pesar de ser un modelo matemático simple, esta técnica de representación complico el procesamiento de datos del algoritmo. A continuación, se expone como se adecuo el modelo matemático a nuestro entorno y además se realiza un análisis para la solución del problema mencionado.

3.1.2.1 Variables. Las variables de entrada del algoritmo de Dijkstra planteado en este proyecto son:

- obstacle = Vector de localización x-y de cada obstáculo en el mapa.

Figura 36. Clase del vector que representa obstacle.

- p = Terna de valores que ofrecen el valor inicial, valor final y la magnitud máxima en centímetros de uno de los lados del mapa (para el proyecto, este valor será 500).

El proceso de avance de este algoritmo está determinado por tres conjuntos primordiales:

La matriz *open*, la cual está encargada de conservar los nodos adyacentes al nodo que se está analizando. Para realizar esta función debe guardar las coordenadas del nodo que se analiza, las coordenadas del nodo al que se planea moverse y el costo de realizar esta acción.

Figura 37. Forma del vector inicial de la matriz open.

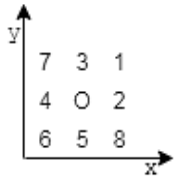
```
%         x_i         y_i     costo    last_x      last_y
open=[p.start(1) p.start(2) 0      p.start(1) p.start(2)];
```

La matriz *close*, la cual almacenara todos los nodos que ya han sido visitados. Cada uno de estos nodos es redireccionado desde el vector open una vez que han sido "*cerrados*". Los valores de esta matriz tendrán las mismas características a los de la matriz *open.*

La matriz *MotionModel,* esta matriz tiene los costos respectivos para el movimiento entre nodos. Siendo estos ocho, representaran la dificultad para moverse de un nodo a otro. Debido a que es una característica propia del mapa, cada costo representará una distancia y será proporcional al mínimo valor de resolución del mapa seleccionado.

Figura 38. Valores en el vector MotionModel y su representación en el plano x-y.

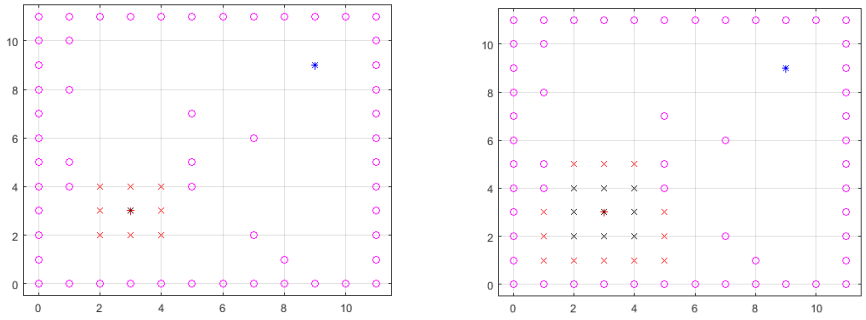
```
%   [ x  y costo]
next=[1  1  1.414  %1
      1  0  1      %2
      0  1  1      %3
     -1  0  1      %4
      0 -1  1      %5
     -1 -1  1.414  %6
     -1  1  1.414  %7
      1 -1  1.414];%8
```



3.1.2.2 Funcionamiento. El proceso de obtención de la ruta inicia una vez determinado la posición inicial (asterisco negro) y la final (asterisco azul). El modelo

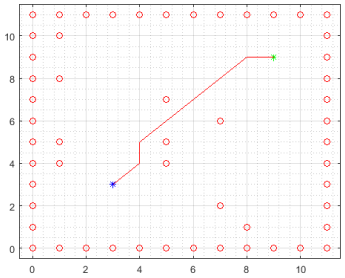
de movimiento analiza los 8 nodos adyacentes (nodos rojos) a la posición y determina si son obstáculos (círculos rosados) o no, con el fin de descartarlos y reducir el análisis de nodos, los nodos que son una posible ruta son puestos en la matriz *open* y el nodo actual se mueve a la matriz *close* (Figura 39, A).

Figura 39. Avance de los nodos en estudio bajo el algoritmo de Dijkstra.



En la siguiente iteración se tomarán en análisis los nodos adyacentes a los almacenados en la matriz open y los reemplazarán. Moviendo los nodos de *open* a *close*, para las siguientes iteraciones hasta alcanzar el objetivo. Antes de realizar el traslado entre las matrices determina cual es la ruta de mínima distancia entre la posición actual y la siguiente, de tal manera de no sobrescribir valores, hallando así las mínimas distancias de nodo a nodo (Figura 39, B).

Figura 40. Ruta obtenida por el algoritmo de Dijkstra.



Una vez alcanzado el objetivo, se tomará la matriz *close* y se analizará el vector final y la mínima distancia a la posición anterior, se realizará este mismo proceso hasta retornar al valor inicial y así obtener la ruta.

3.1.2.3 Mapa. Para que el algoritmo de Dijkstra planteado funcionara con el entorno simulado, se tomó una representación diferente de ella, en donde se listaban las coordenadas x-y de cada uno de los obstáculos con una resolución de un centímetro. Esto exige la lectura de 250.000 elementos de una matriz binaria (500^2 o un elemento por centímetro).

Figura 41. Vector de coordenadas de los obstáculos extraída de la matriz binaria de ocupación con una resolución de 1 cm.

La forma del algoritmo requiere el manejo de varios vectores de gran tamaño además de conservarlos y reescribirlos en cada iteración esto se debe a que los vectores nuevos se deben retroalimentar con los vectores guardados. A causa de esta estructura, la cantidad de funciones usadas en cada iteración aumenta a medida que los vectores se acumulan.

Figura 42. Representación de la misma ruta obtenida con el algoritmo de Dijkstra con diferentes resoluciones.

Dadas las circunstancias anteriormente mencionadas, el algoritmo planteado llego a realizar análisis de varios minutos para rutas triviales (líneas rectas de poca distancia). Este inconveniente se resolvió reduciendo la resolución de los datos del

mapa que el algoritmo recibe (matriz *obstacle*) a decímetros (Figura 43). Esta "mejora" redujo significativamente los tiempos de procesamiento y la memoria utilizada por el algoritmo, pero como efecto adverso redujo la resolución de la ruta obtenida.

Figura 43. El mismo vector de coordenadas mencionado anteriormente después de la reducción de resolución.

3.1.2.4 Análisis. El algoritmo de Dijkstra planteado en este proyecto reemplaza la representación gráfica del mapa entre los grafos comúnmente utilizados a una representación de celdas de ocupación; por lo que se dota a cada vértice un costo unitario de traslación. De tal manera que los cuatro puntos cardinales tienen un valor de uno y las combinaciones de estos serían iguales a $\sqrt{2}$.

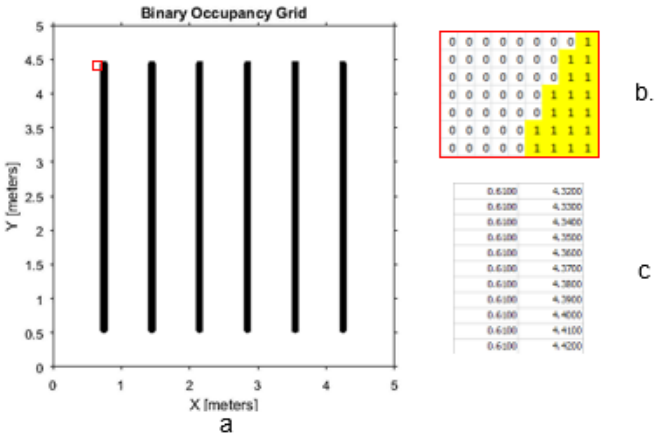
El principal inconveniente al realizar esta simplificación es que el mapa a analizar tendría una dificultad para trasladarse similar por todo el campo libre (es un mapa binario), de manera que no se podría introducir el concepto de terreno y las dificultades que estas abarcan en su movimiento.

Otro inconveniente presentado en el algoritmo es que la cantidad de nodos a analizar crecía de manera exponencial a cada iteración, lo que requería una mayor cantidad de tiempo y exigía mayor uso de memoria y procesamiento. Debido a este inconveniente se decidió simplificar el algoritmo a nodos ubicados cada diez centímetros, lo que redujo el tiempo de varios minutos a segundos a cambio de un poco de resolución.

3.2 SIMULACIÓN DE LOS MODELOS MATEMATICOS.

El concepto base para realizar la representación del entorno es el de binarización, en donde se tomarán los obstáculos, identificados con un valor de uno y el campo transitable con un valor de cero. Para realizar este mapa se tuvo en cuenta la geometría comúnmente utilizada en los campos de maíz, los cuales poseen un patrón organizado y además su terreno es generalmente controlado. Esto facilitó la simplificación del algoritmo a una matriz binaria.

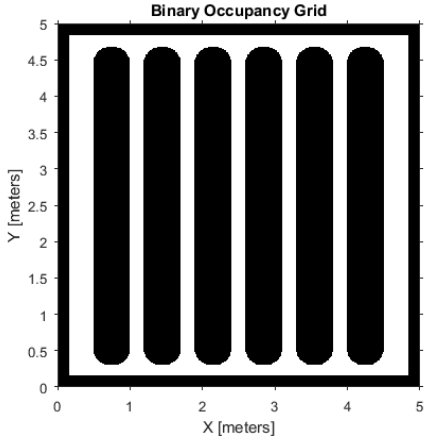
Figura 44. Representaciones del mapa. a. Cuadricula binaria de ocupación. En rojo se muestra el área a observar. b. Matriz binaria de ocupación en el área roja. c. Coordenadas XY de la ubicación de algunos de los obstáculos en el área roja.



Debido a que los algoritmos planteados tienen diferentes enfoques respecto a su manera de percibir el entorno, se plantearon tres formas diferentes de expresar el mismo mapa con el fin de satisfacer esas necesidades. La primera de ellas es la cuadricula binaria de ocupación, la que representa el mapa del entorno con una extensión fig a (figura 44) lo que facilitara la exposición de los resultados, para esto

se utilizó una de las librerías que ofrece MATLAB[18]. La segunda representación es la de obstáculos, la cual ofrece una matriz con las coordenadas geográficas de cada uno de los obstáculos (figura 44 c); esta representación es utilizada por el algoritmo de Dijkstra debido a que el algoritmo asume que los demás espacios son libres. La última representación es la matriz binaria de ocupación (figura 44 b) la cual genera una matriz de unos y ceros representando las condiciones del entorno; esta representación es utilizada en el algoritmo de Campos Potenciales, en donde se realiza la traducción de filas-columnas a coordenadas x-y.

Figura 45. Cuadricula binaria de ocupación expandida.



Debido a que los algoritmos simplifican su funcionamiento al asumir que el objeto en movimiento es una partícula con la resolución minina del mapa, las rutas que se obtuvieran serían imposibles de satisfacer por un robot real con dimensiones superiores a las de la unidad de resolución del algoritmo. Para solucionar este inconveniente se decidió realizar una expansión proporcional de los obstáculos en la matriz (Figura 45).

[18] MATHWORKS. Robotic System Toolbox. [Citado el 15 septiembre de 2017]. Disponible en: https://la.mathworks.com/help/pdf_doc/robotics/robotics_ref.pdf

3.3 SELECCIÓN DEL SISTEMA DE NAVEGACIÓN

Con el fin de determinar cuál de los algoritmos mencionados es mejor, se debe realizar un paralelo entre variables que ambos algoritmos compartan; a pesar de realizar la misma función, el funcionamiento de estas es muy distinto, por lo que una comparación cualitativa de sus funciones resultaría incompleta para el espacio que ocuparan finalmente en la placa computadora integrada al robot.

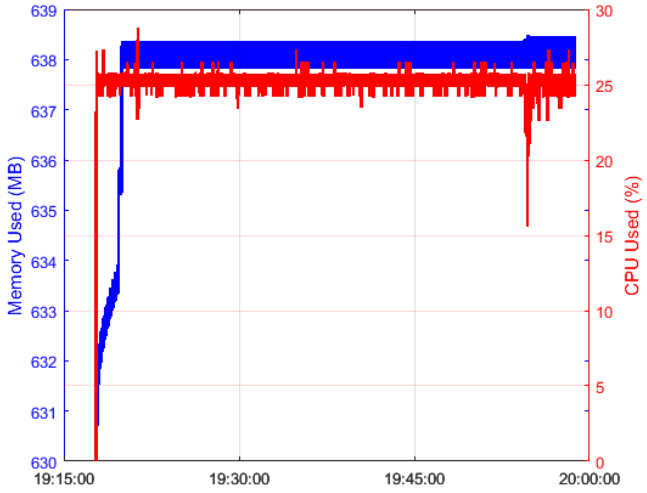
Se tomaron unas variables independientes a las del algoritmo, que tienen como objetivo evaluar los requerimientos del computador que cada algoritmo demanda, estos son:

- Tiempo de procesamiento: se necesitará determinar cuánto tiempo se demora el algoritmo en ofrecer una ruta en tiempo real.
- Uso de CPU (Unidad central de procesamiento): este aspecto hace referencia a la cantidad de instrucciones procesadas por el algoritmo, su objetivo es cuantificar que tanto ocupa las capacidades de la CPU.
- Memoria usada: la memoria que la CPU utiliza para guardar variables temporales de los algoritmos es la memoria RAM; con este aspecto se evaluara cómo evoluciona la memoria que el algoritmo utiliza.

Para realizar esta tarea se recurrió al uso de una biblioteca especializada de MATLAB[19] la cual mide estas variables en tiempo real y genera una gráfica que muestra la evolución del código en cada segundo.

[19] ZHAO, Xin. System information class for windows. [Base de datos en línea]. Febrero de 2010. MathWorks. [citado el 15 septiembre de 2017)] Disponible en: https://la.mathworks.com/matlabcentral/fileexchange/26662-system-information-class-for-windows?s_tid=prof_contriblnk

Figura 46. Grafica generada por la biblioteca utilizada donde se muestra la evolución de las variables vs el tiempo.



La curva azul evalúa la memoria RAM utilizada en megabytes, la curva roja evalúa el porcentaje de CPU usado; ambas curvas se evalúan en función del tiempo, aquí se utiliza el reloj interno del sistema.

Cabe resaltar que estos requerimientos que se van a evaluar evolucionaran de manera diferente de sistema a sistema, por lo que es necesario mencionar las características propias de cada uno de ellos ya que será un aspecto que influirá altamente en los resultados de la plataforma robótica.

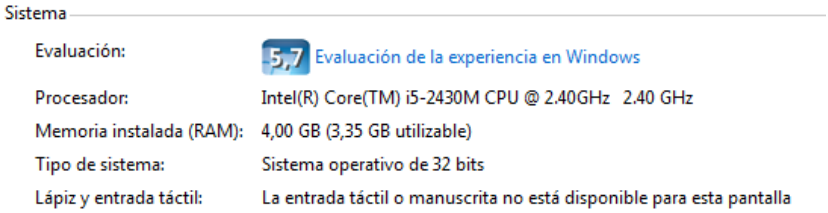
A continuación, se mostrarán las características de los equipos utilizados para la simulación:

Figura 47. Equipo 1: Sistema operativo Windows 10 Home.

Sistema

Procesador:	Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz 2.50 GHz
Memoria instalada (RAM):	8.00 GB (7.89 GB utilizable)
Tipo de sistema:	Sistema operativo de 64 bits, procesador x64
Lápiz y entrada táctil:	La entrada táctil o manuscrita no está disponible para esta pantalla

Figura 48. Equipo 2: Sistema operativo Windows 7 Ultimate.



3.3.1 Simulaciones Para la realización de las pruebas de selección del algoritmo se establecerá un paralelo del comportamiento de los algoritmos (Dijkstra vs Campos potenciales) para crear una trayectoria entre dos puntos del mapa, posteriormente se pondrán obstáculos en el entorno que obliguen al algoritmo a crear una trayectoria que evite las rutas triviales. Finalmente se realizará una prueba de barrido completo del mapa sin y con obstáculos.

3.3.1.1 Prueba 1 – Modo ruta

Coordenada inicial: (40,40)
Coordenada final: (460,460)

Figura 49. Prueba 1 - Trayectoria sin obstáculos.

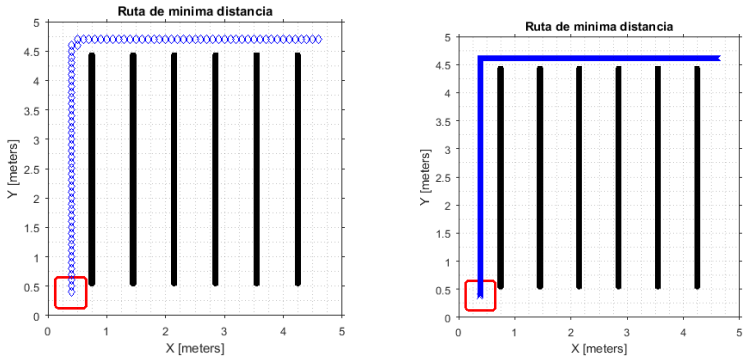
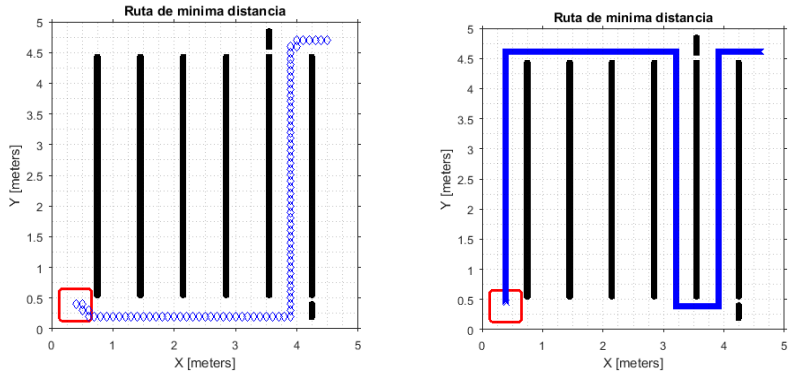


Figura 50. Prueba 1 - Trayectoria con obstáculos.



En la simulación sin obstáculos los algoritmos obtuvieron misma ruta trivial para llegar al objetivo. Con el fin de analizar a fondo el funcionamiento de los algoritmos se colocaron obstáculos que eliminaran las rutas triviales. Estos obstáculos lograron resaltar la naturaleza de los algoritmos; en el algoritmo de Dijkstra se puede observar una ligera reducción en el tiempo de procesamiento debido a la necesidad de analizar menos nodos libres entre el punto inicial y el final (la ruta es dirigida en una dirección en el último par de hileras). En el caso del algoritmo de campos potenciales se puede observar que el algoritmo tomo inicialmente la misma ruta que en el caso sin obstáculos, hasta que la partícula llego al obstáculo en donde encontró un mínimo local y posteriormente logro salir exitosamente de el para poder llegar al objetivo.

3.3.1.2 Prueba 2 – Modo ruta

Coordenada inicial: (40,250)
Coordenada final: (460,250)

A pesar de que ambos algoritmos obtuviesen los mismos resultados en ambos entornos, se seleccionaron estos puntos con el fin de inducir al algoritmo de campos potenciales en varios mínimos locales y así determinar en qué medida el procesamiento variaba.

Figura 51. Prueba 2 - Trayectoria sin obstáculos.

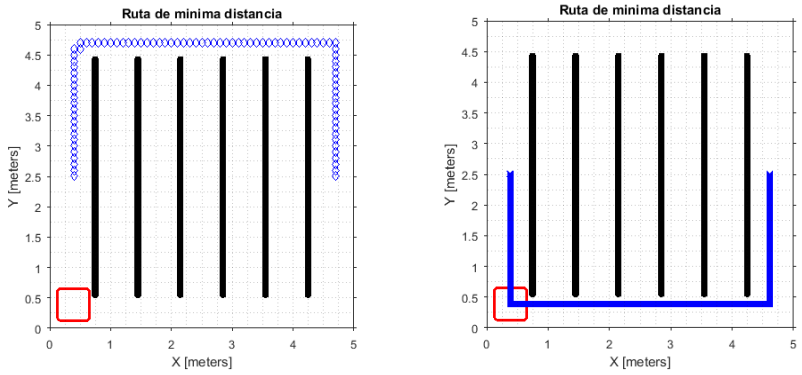
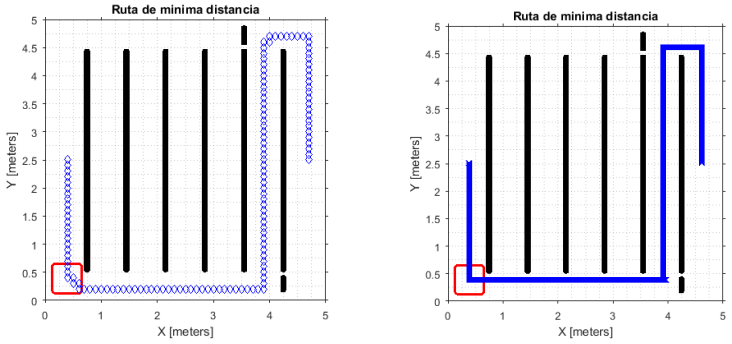


Figura 52. Prueba 2 - Trayectoria con obstáculos.



El algoritmo de Dijkstra obtuvo la ruta con poca variación de tiempo en ambos entornos lo que resalta la poca importancia de la posición de los obstáculos en el entorno. Por el contrario, el algoritmo de campos potenciales demando más tiempo cuando se colocaron los obstáculos. La posición inicial de la partícula es un mínimo local en este entorno al igual que la posición (3.9, 2.5), la fuerza potencial resultante es igual en cualquier dirección. Esta condición explica el aumento del tiempo de procesamiento. A pesar de esto, el algoritmo de campos potenciales obtiene la ruta de manera exitosa.

3.3.1.3 Prueba 3 – Modo barrido En este modo el algoritmo hallara la trayectoria necesaria para recorrer todas las hileras del cultivo.

Figura 53. Prueba 3 - Trayectoria sin obstáculos.

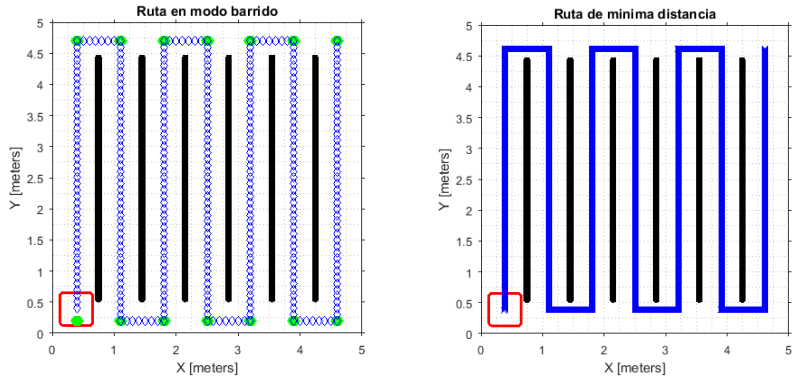
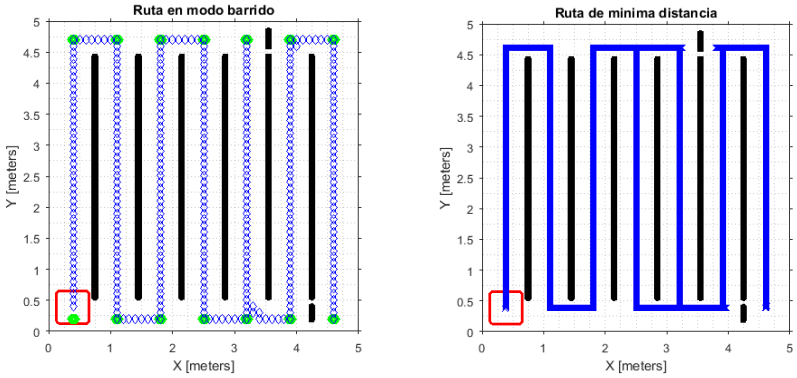


Figura 54. Prueba 3 - Trayectoria con obstáculos.



La trayectoria sin obstáculos resulto exitosa en ambos casos, obteniendo la ruta trivial entre los nodos maestros. En la trayectoria con obstáculos, por el contrario, se puede apreciar un comportamiento especial en ambos algoritmos. A pesar de que recorre todos los nodos maestros, el algoritmo de Dijkstra recorre dos veces la misma ruta (entre hileras 4 y 5, luego entre hileras 5 y 6) un comportamiento esperado pero que puede llegar a ser contraproducente para un tiempo de recorrido planificado. El algoritmo de campos potenciales realizo un giro inesperado al toparse con el primer obstáculo, lo que significa que el campo potencial en este mínimo local es demasiado alto, lo que lo obliga a retroceder en vez de avanzar por la misma ruta.

A pesar de que ambos algoritmos generen una ruta más larga, realizan la inspección completa del mapa de manera exitosa. El algoritmo de Dijkstra puede ser mejorado con un método de discriminación de nodos redundantes con el fin de no realizar una ruta repetida en la trayectoria entregada. El algoritmo de campos potenciales puede ser ajustado con unos valores repulsivos disminuidos lo suficiente para que no se quede atorado en un mínimo local.

3.3.2 Análisis de rendimiento del sistema

Tabla 1. Requerimientos de rendimiento de los equipos para la ejecución del algoritmo en un entorno sin obstáculos.

EQUIPO 1 SIN OBSTACULOS	Equipo 1					
	Prueba 1		Prueba 2		Prueba 3	
	Dijkstra	C.Pot	Dijkstra	C.Pot	Dijkstra	C.Pot
Tiempo (s)	7.9793	0.1465	8.1588	0.1474	16.2815	0.3711
Uso de CPU (Max. %)	29.97	26.7	26.27	28.38	25.86	30.06
Memoria Utilizada (Max. Mb)	950.5	1045.1	952.1	993.1	952.3	983.1

EQUIPO 2 SIN OBSTACULOS	Equipo 2					
	Prueba 1		Prueba 2		Prueba 3	
	Dijkstra	C.Pot	Dijkstra	C.Pot	Dijkstra	C.Pot
Tiempo (s)	24.67	0.2591	24.669	0.2654	46.3113	1.2766
Uso de CPU (Max. %)	28.86	49.5	25.74	75.6	27.3	52
Memoria Utilizada (Max. MB)	575.75	704.5	790.6	745.8	668.5	747.3

Como era de esperarse el impacto del procesador y de la cantidad de memoria del equipo influye de manera directa en el tiempo de ejecución de los algoritmos, por lo que se asegura que el tiempo de ejecución en la tarjeta Raspberry Pi (quad core 900 MHz y RAM de 1GB) será mayor al del equipo con características inferiores. El

tiempo de ejecución del algoritmo de Dijkstra es evidentemente mayor al de Campos Potenciales y en el mejor de los casos (prueba 3) el tiempo es 30 veces mayor. Lo que indica un tiempo de procesamiento con resolución en decenas de segundos y no en segundos, para el algoritmo Dijkstra.

Los requerimientos de uso de CPU fueron más evidentes en el equipo de características inferiores, en donde se puede observar la mayor exigencia por el algoritmo de Campos Potenciales.

Finalmente, la cantidad de memoria utilizada fue proporcional a las características del equipo, pero al igual que el requerimiento de uso de CPU se evidencio un mayor uso de memoria en el uso del algoritmo de Campos Potenciales.

Tabla 2. Requerimientos de rendimiento de los equipos para la ejecución del algoritmo en un entorno con obstáculos.

EQUIPO 1 CON OBSTACULOS	Equipo 1					
	Prueba 1		Prueba 2		Prueba 3	
	Dijkstra	C.Pot	Dijkstra	C.Pot	Dijkstra	C.Pot
Tiempo (s)	7.6159	0.2815	7.7383	0.2907	17.1943	1.4673
Uso de CPU (Max. %)	25.85	59.6	25.95	64	25.77	64.1
Memoria Utilizada (Max. Mb)	1038.5	1074	1032.3	1076	1035.1	1092.8

EQUIPO 2 CON OBSTACULOS	Equipo 2					
	Prueba 1		Prueba 2		Prueba 3	
	Dijkstra	C.Pot	Dijkstra	C.Pot	Dijkstra	C.Pot
Tiempo (s)	22.092	0.9094	24.17	0.6543	48.8264	4.9039
Uso de CPU (Max. %)	39	45.8	25.74	56.4	27.3	48.89
Memoria Utilizada (Max. MB)	621.8	750.7	640.4	757.5	769	798.2

Al igual que en las pruebas sin obstáculos, los tiempos obtenidos por el algoritmo de Dijkstra son inferiores, pero en estas pruebas se pueden evidenciar las características de los algoritmos.

El algoritmo de Dijkstra en las pruebas 1 y 2, presentó tiempos menores en el entorno con obstáculo; este comportamiento se le atribuye a que el algoritmo debe procesar menos casillas desocupadas. Por el contrario, en la prueba 3, se obtiene un tiempo mayor debido al retorno innecesario a unos nodos previamente visitados. En cuanto al tiempo de procesamiento y a la memoria utilizada, es evidente la poca variación de estos parámetros al aplicar obstáculos, como se explicó antes, se debe a que el algoritmo se desempeña mejor en entornos con menos espacios vacíos.

En cambio, el algoritmo de Campos Potenciales registró tiempos mayores y esto se debe a que los obstáculos planteados tienen como fin generar mínimos locales que evidenciaran el desempeño del algoritmo para superarlos en un tiempo reducido. En este entorno se observa la demanda superior de procesamiento al exigir el cambio de posición de múltiples mínimos locales. A pesar de los obstáculos el algoritmo de Campos Potenciales realiza una trayectoria que cumple los parámetros establecidos.

3.3.2 Selección Ambos algoritmos se desempeñaron de manera adecuada en la creación de trayectorias efectivas entre los puntos y en modo barrido, las rutas creadas pueden mover a la plataforma robótica de manera exacta a su objetivo; por esta razón la selección del sistema de navegación está basada en su desempeño informático. El parámetro más importante a la hora de implementar el algoritmo de navegación será el del tiempo que este tome para realizar una ruta debido a la necesidad de crear nuevas trayectorias a mitad de carrera en caso de que encuentre un obstáculo. Posteriormente se tendrá en cuenta el porcentaje del CPU que este requiere durante su ejecución, debido a que la plataforma robótica ejecutará otros procesos simultáneamente para mantener la trayectoria y para moverse.

Figura 55. Comparación de rendimiento en equipo 1 sin obstáculos.

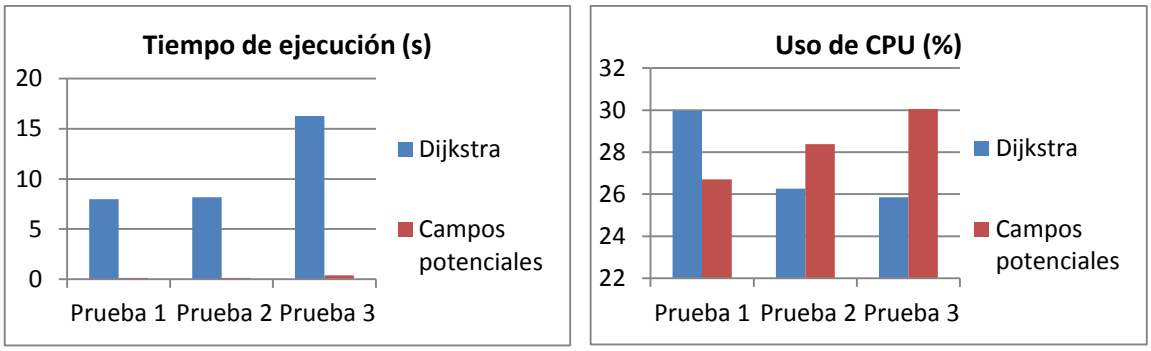
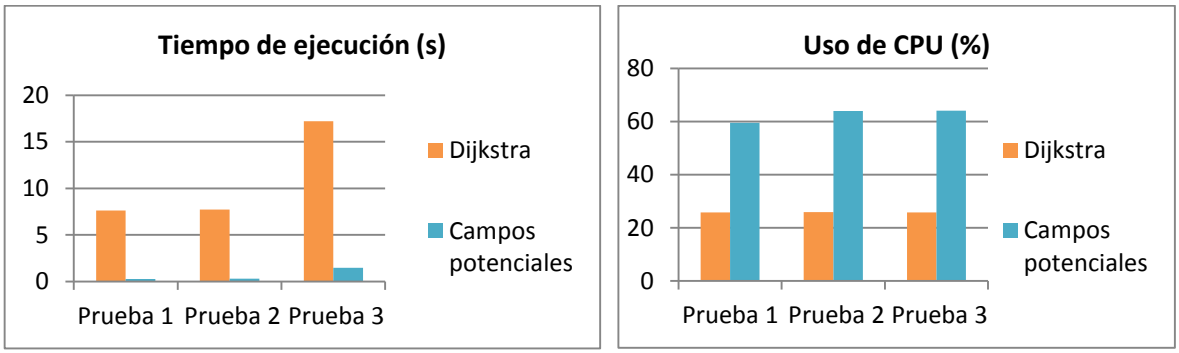


Figura 56. Comparación de rendimiento en equipo 1 con obstáculos.



El algoritmo que se desempeñó con mejores tiempos fue el de campos potenciales que en el mejor de los casos obtuvo un tiempo 36 veces menor al del algoritmo de Dijkstra. A pesar de esto obtuvo valores altos en el porcentaje de CPU requerido lo que podría ralentizar los procesos que se realizan en paralelo, sin embargo, los tiempos de ejecución reducidos permitirán a la computadora liberarse de la función de navegación más rápidamente. Por estas razones el algoritmo seleccionado es el de Campos potenciales.

4. SISTEMA DE PERCEPCIÓN

Al igual que un ser humano requiere de los sentidos para realizar tareas, los robots móviles autónomos requieren de sensores de percepción que le den información de su estado y de su entorno; la información capturada por estos sensores se usará por los sistemas de control y navegación para la corrección de la ruta o de las tareas que debe realizar, en paralelo al seguimiento de la ruta, esto se traducirá finalmente al debido accionamiento de los actuadores.

Los sensores que captan información del estado del robot se le llaman propioceptivos, al igual que el ser humano siente la posición relativa de sus músculos para regular los movimientos y sus fuerzas, un robot requiere el conocimiento de su orientación y de la posición de sus actuadores. La plataforma robótica Lynxmotion tiene integrado en cada uno de sus cuatro motores un encoder incremental de efecto Hall el cual se usará para determinar la posición relativa del robot, debido a la cinemática del vehículo (Skid-steering o dirección por deslizamiento) y al error que este tipo de sensores acumula en largos recorridos, los encoder no son confiables a la hora determinar la posición relativa del robot, por lo que se requieren otros sensores que complementen esta información. Para determinar la orientación del robot es necesario el uso de un compás náutico, que tiene como objetivo apoyar a los encoder cuando se realiza un giro y corregir la ruta. A pesar de que este sensor use un marco de inercia global (campos magnéticos o principios giroscópicos) este se modificara para crear un marco de referencia relativo a la orientación del entorno con el fin de que el robot se mueva con facilidad.

4.1 ENCODER

Los motorreductores, incluidos en la plataforma robótica LynxMotion, tienen acoplados en su parte posterior un encoder incremental de cuadratura, el cual determina la cantidad de revoluciones por medio de un sensor de efecto Hall de dos canales. Este encoder ofrece una resolución de 48 cuentas por revolución (cantidad de veces que una de las juntas magnéticas pasa cerca de uno de los canales del sensor Hall) en el eje principal del motor. Para determinar la resolución con el reductor de velocidad, se multiplica la relación de transmisión por la resolución del encoder.

Figura 57. Vista posterior del motorreductor, se observa el encoder acoplado a él.

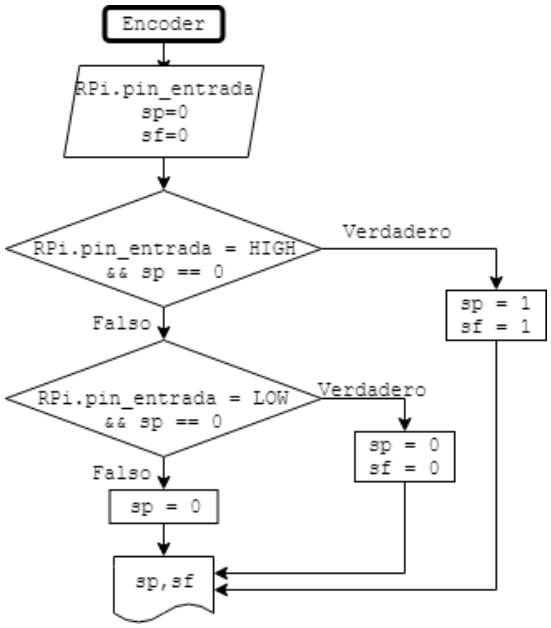
Fuente: POLOLU. Metal gearmotor 37Dx73L mm with 64 CPR Encoder. [En línea] <https://www.pololu.com/product/2827> [Citado en 8 marzo de 2018]

El sensor de efecto Hall requiere una entrada de voltaje de entre 3.5 y 20 V y usa un máximo de 10 (mA). Los canales A y B son ondas cuadradas entre 0 V y el valor de voltaje de entrada, están desfasados aproximadamente 90°. La frecuencia de transición determina la velocidad del motor mientras que el orden determina la dirección.

La función de los encoder en la plataforma robótica será la de mostrar la cantidad recorrida en función del número de cuentas de la junta magnética. Para esto se

realiza una lectura continua de esta entrada binaria y se cuenta la cantidad de veces que la junta magnética es percibida. Teniendo la información de las dimensiones de la llanta y las especificaciones del motor, se puede determinar qué cantidad de ranuras representa un centímetro de distancia recorrida. Para esto es necesario llamar dos variables que graben la posición nueva del encoder y la posición anterior, con el fin de poder realizar el debido conteo de posiciones magnéticas que se traducirá a una distancia en centímetros.

Figura 58. Diagrama de flujo del funcionamiento del encoder.



Debido a la cinemática del vehículo, en donde se requiere el deslizamiento de las llantas para realizar giros; no se puede utilizar el encoder como una herramienta confiable para contar en los giros sobre su propio eje. Una de las fallas más probables en el conteo realizado por el encoder es el deslizamiento, en donde erróneamente el robot asumiría que se ha movido. Por estas razones es necesaria la complementación del sistema propioceptivo del robot con otros sensores.

4.2 COMPAS NÁUTICO

Para la selección de este sensor se tuvieron en cuenta algunos parámetros determinados por las limitaciones de la computadora RPi, tales como: voltaje de operación de 3.3v, protocolo de comunicación compatible (I2C), tamaño y costo reducido.

Se analizaron los diferentes datasheets de los compases encontrados que cumplían con estas características y se realizó una tabla comparativa que evaluara los parámetros más importantes para su posterior selección.

Tabla 3. Parámetros de selección compas náutico.

Parámetro de selección	Giroscopio de 3 ejes (ITG-3205)[20]	Magnetómetro digital de 3 ejes (HMC-5883L)[21]
Resolución	~2°	~1°
Corriente requerida	6.5 mA	100 µA
Tipo de comunicación	I2C	I2C
Tamaño	2.2 x 1.7 cm	1.8 x 1.7 cm
Precio	$55.000	$16.000

A pesar de que estos sensores tengan características muy similares su forma de obtener información de la variación angular es muy diferente. El magnetómetro obtiene información de los campos magnéticos terrestres, por lo que su sistema inercial es absoluto, esto exige una transformación de unidades magnéticas a angulares posterior a la toma de datos. El giroscopio a diferencia del magnetómetro capta la variación angular del dispositivo con respecto al tiempo, lo que posteriormente integra para adquirir la orientación.

[20] INVENSENSE. ITG-3200 Product specification. [En línea] < http://www.hobbytronics.co.uk/datasheets/PS-ITG-3200-00-01.4.pdf > [Citado en 8 febrero de 2018]
[21] HONEYWELL. 3-Axis Digital Compass IC HMC5883L. [En línea] < http://www.hobbytronics.co.uk/datasheets/HMC5883L-FDS.pdf > [Citado en 8 febrero de 2018]

Ninguno de los sensores mide directamente la variación angular del robot, por lo que es necesaria una transformación de datos adecuada y eficaz. El giroscopio puede determinar con facilidad los cambios rápidos de orientación, pero puede llegar a acumular gran error a lo largo del tiempo debido a la integración de los datos. Por el contrario, el magnetómetro, no puede llegar a detectar los cambios rápidos de orientación, pero si puede determinar con exactitud la orientación con base en el sistema de referencia global.

En base en la tabla de selección y en la anotación realizada anteriormente se selecciona el magnetómetro de tres ejes HMC-5883L como compas náuticos de la plataforma robótica. Para realizar los giros es más útil conocer el ángulo de orientación exacto, sin tener que llegar a integrar los datos ya que los errores pueden incrementar en cada ángulo rotado.

A continuación, se listan las especificaciones más importantes del sensor:

Tabla 4. Especificaciones del sensor HMC-5883L.

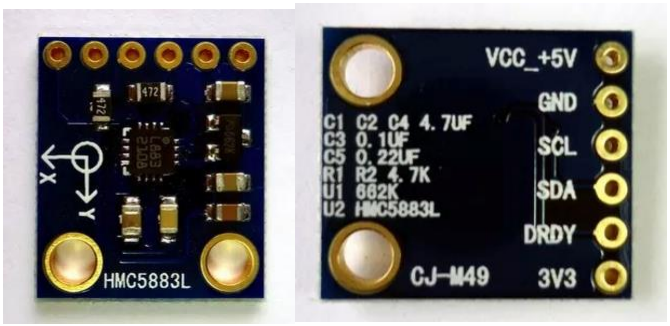
Dimensiones	1.8 x 1.7 cm
Requerimientos de poder	2.7 a 3.6 VDC
Comunicación	I2C
Rango de mediciones	+/- 8 Gauss
Resolución	1 mili-Gauss

Uno de los requerimientos más importantes para retroalimentar el sistema de localización es el de la orientación espacial; el uso adecuado de los encoder puede sugerir un valor aproximado de la orientación del robot, pero el error de este valor tiende a incrementar con el tiempo, por lo que se hace necesario un receptor confiable.

El magnetómetro es un artefacto utilizado para medir la dirección de una señal magnética; este sensor opera pasivamente, lo que quiere decir que traduce señales

externas en pulsos eléctricos que serán leídos por la computadora. Para utilizar este artefacto como apoyo al sistema de localización se debe medir las modificaciones de los campos magnéticos locales y grabar este cambio en razón del tiempo.

Figura 59. Vista frontal y posterior del Magnetómetro digital de tres ejes HMC-5883L utilizado en el robot.



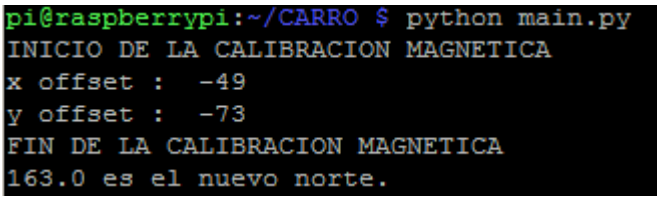
Debido a que la medición del campo magnético terrestre varía en función de la latitud, longitud y altitud; es necesaria una calibración previa a la realización de la ruta del robot. Esta calibración se realiza por medio de la toma de varias mediciones magnéticas con el fin de obtener las mediciones máximas y mínimas del campo magnético terrestre en ese punto de la tierra, posteriormente se determina el valor medio y los valores de compensación con los cuales trabajara el magnetómetro en el movimiento del robot.

Figura 60. Valores de compensación obtenidos en la calibración.

```
INICIO DE LA CALIBRACION MAGNETICA
x_min:-357
y_min:-406
x_max:239
y_max:215
x offset :  -59
y offset :  -96
FIN DE LA CALIBRACION MAGNETICA
```

Además de la rutina de calibración, el magnetómetro realiza otras funciones del robot. Una de estas es el de la creación de un marco de referencia global para el entorno en el que se encuentra; esta rutina de orientación se realiza antes de moverse a través del mapa. El objetivo de esta rutina es el de crear cuatro puntos de referencia similares a los puntos cardinales de la tierra, pero en vez de tomar el norte magnético de la tierra este tomara la orientación inicial que se le indique como el “norte” del entorno en cuestión. Luego se determinan los demás “puntos cardinales” del entorno los cuales serán constantes globales que se atribuirán a las diferentes funciones del robot. Estas constantes globales varían de entorno a entorno de acuerdo con la orientación de ellos y serán importantes a la hora de realizar giros de manera precisa.

Figura 61. Captura de la calibración magnética con valores de compensación y coordenada de referencia.



Las rutinas anteriores son la base de la rutina principal del magnetómetro, la rutina de confirmación de giro. El éxito de los giros realizados depende de una correcta calibración y de la creación de los puntos de referencia adecuados.

La rutina de giro del magnetómetro es la encargada de realizar una toma constante de cada rotación sobre su eje realizada por el robot. Debido a la trivialidad del mapa, las cuatro orientaciones principales (norte, sur, este y oeste del entorno) son las únicas direcciones necesarias para que el robot cumpla su objetivo. Esta rutina determina la orientación inicial del robot realizando una toma de datos y posteriormente determina la diferencia de orientación entre la posición inicial y la

final. En cada ángulo que se gire se realizara una toma de datos y se comparara hasta que la diferencia sea cero.

Figura 62. Rutina de enderezado a la ruta donde se ven las mediciones segundo a segundo durante el giro.

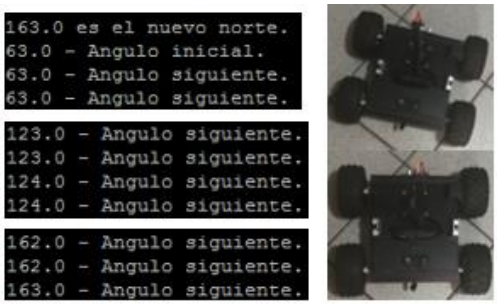
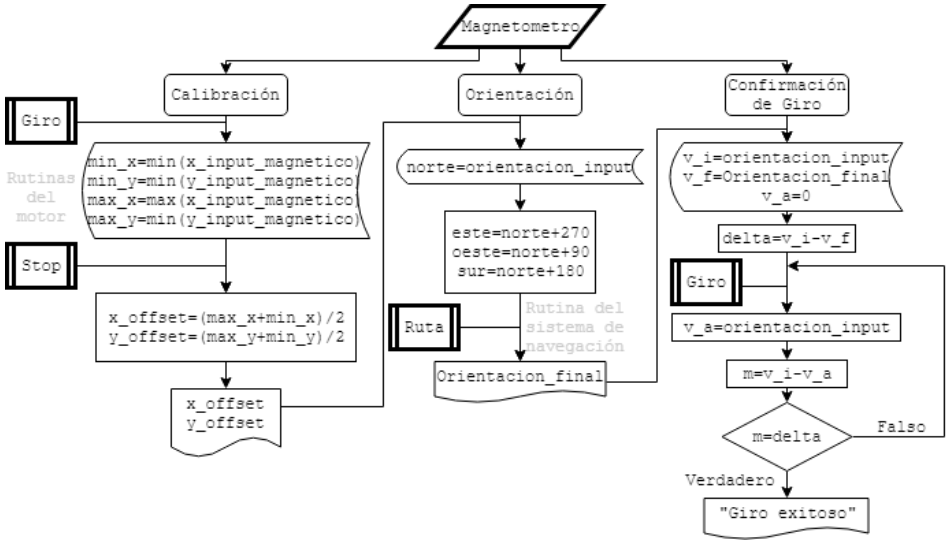


Figura 63. Diagrama de flujo del funcionamiento de las rutinas del magnetómetro.



4.3 SENSORES DE DISTANCIA PARA LA DETECCIÓN DE OBSTACULOS.

Para la selección de este tipo de sensores se tuvieron en cuenta parámetros similares a los del compás náutico. Voltaje de operación de 5v o menos, protocolo

de comunicación compatible (I2C), rango de medición de al menos entre 10 cm y 1m y costo reducido.

Entre los sensores de distancia encontrados en la teoría se encuentran clasificados los capacitivos, los inductivos, los fotoeléctricos y los ultrasónicos. Debido a que los dos primeros requieren una modificación del mapa ya sea para crear un campo eléctrico o un magnético, se descartaran.

Se analizaron los diferentes datasheets de los sensores de distancia encontrados que cumplían con estas características y se realizó una tabla comparativa que evaluara los parámetros más importantes para su posterior selección.

Tabla 5. Parámetros de selección sensores de distancia.

Parámetro de selección	**Infrarrojo Sharp GP2Y0A2[22]**	**Ultrasónico HC-SR05[23]**
Rango de medición	10 cm – 150 cm	2 cm – 450 cm
Resolución	1 mm	3 mm
Tipo de salida	Analógica	I2C
Velocidad de muestreo	50 Hz	40 Hz
Corriente requerida	33 mA	20 mA
Costo	$35.000	$20.000

A pesar de la resolución y velocidad de muestreo superior del sensor infrarrojo, la salida analógica exige un circuito complementario que traduzca la señal analógica a una digital para que pueda ser leída por la Raspberry Pi. Esto conlleva un mayor costo y, además, un circuito adicional que transforme la señal puede generar ruido en la señal de entrada; además, por las condiciones geométricas del entorno y del robot, una diferencia de resolución de milímetros no influye en gran medida en el

[22] POLOLU. Sharp GP2Y0A21YK0F Analog distance sensor. [En línea] <https://www.pololu.com/product/136> [Citado en 13 enero de 2018]
[23] SPARKFUN. Ultrasonic ranging module HC-SR05. [En linea] < https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR05.pdf> [Citado en 13 enero de 2018]

funcionamiento del algoritmo. Por estas razones se seleccionó el sensor ultrasónico HC-SR05.

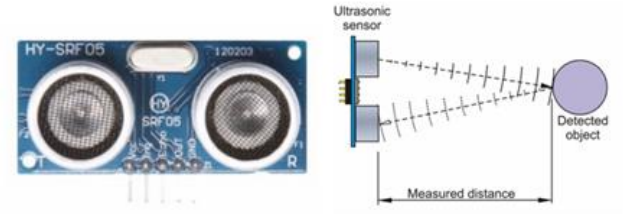
A continuación, se listan las especificaciones más importantes del sensor:

Tabla 6. Especificaciones del sensor de distancia ultrasónico SRF-05.

PARAMETRO	VALOR	UNIDAD
Dimensiones del circuito	43 x 20 x 17	mm
Tensión de alimentación	5	Vcc
Frecuencia de trabajo	40	KHz
Rango máximo	4	m
Rango mínimo	1.7	cm
Duración mínima del pulso de disparo (nivel TTL)	10	µS
Duración del pulso eco de salida (nivel TTL)	100-25000	µS
Tiempo mínimo de espera entre una medida y el inicio de otra	20	mS

El principal objetivo del sistema de percepción es la del reconocimiento del entorno de trabajo y de las propiedades del robot, los sensores anteriores ofrecen con exactitud una serie de datos propios del robot que evolucionan con el tiempo, pero quedan incompletos al generar una visión parcial del campo.

Figura 64. Sensor ultrasónico HY-SRF05 y su funcionamiento.



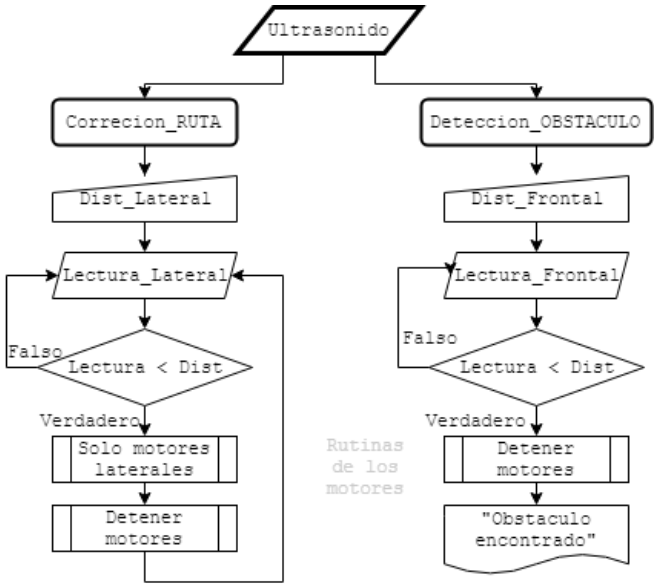
Fuente: TECNO VEGA [en línea] disponible en: https://www.tecnovega.es/moodle/mod/assign/view.php?id=230.

Para poder tener una imagen virtual de lo que se encuentra alrededor del robot se usaran tres medidores ultrasónicos de distancia los cuales se ubicaran en el frente y a los lados, con el fin de tener una visión clara tanto de las filas de maíz como de los obstáculos que pueda encontrar al frente.

Figura 65. Disposición de los sensores ultrasónicos sobre el robot.

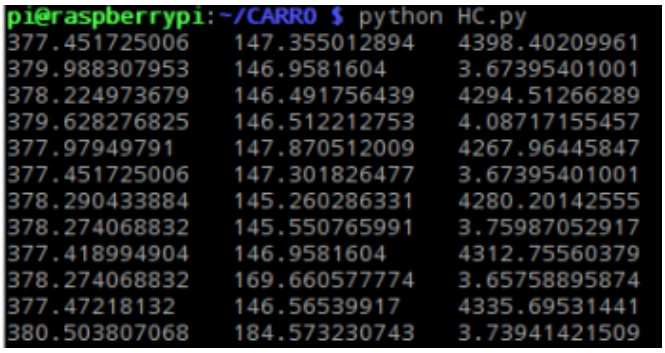
La disposición de cada uno de estos sensores tiene un objetivo específico a la hora de mantener en ruta al robot. El sensor frontal se encargará de generar un aviso oportuno ante posibles obstáculos no identificados en el mapa que se encuentre justo enfrente de el en un rango de 20 cm. Los sensores laterales se encargarán de medir la distancia entre los obstáculos laterales, en este caso las hileras de cultivo, con el fin de usar esta información para adecuar la trayectoria del robot y mantener un movimiento en línea recta ajustando los valores pertinentes en las velocidades de los motores.

Figura 66. Diagrama de flujo del funcionamiento de las rutinas de los ultrasonidos.



Las rutinas del subsistema de detección de obstáculos se realizarán en paralelo con la rutina principal de movimiento del robot, por lo tanto, las lecturas de los ultrasonidos se realizarán de manera constante hasta que el robot llegue a su objetivo.

Figura 67. Ejemplo de data obtenida por los sensores ultrasónicos (izquierdo, derecho y frontal) en un intervalo de 5 segundos en centímetros.



Si el robot llega a encontrar un obstáculo enfrente de él se detendrá inmediatamente y analizará una nueva medición con el fin de descartar posibles errores o ruido. Posteriormente creara una alerta visual en el computador avisando del obstáculo enfrente de él; esperara treinta segundos para que el obstáculo dinámico sea removido para poder continuar con su ruta. Si el obstáculo no es removido significa que es un obstáculo estático y por lo tanto no podrá continuar con la ruta planteada, se realizara un giro de 180 grados y se volverá a llamar al sistema de navegación para crear una ruta alterna que tenga en cuenta el nuevo obstáculo.

5. SISTEMA DE LOCOMOCIÓN

El robot Lynxmotion rover 4WD01 proporcionado por el grupo DicBot de la Universidad Industrial de Santander, Escuela de Ingeniería Mecánica posee las siguientes características:

Componentes de fábrica[24]:

- Componentes estructurales de aluminio
- Paneles Lexan cortados con láser
- Eje de montaje hexagonal de 2 x 12 mm - eje de 6 mm (par)
- Mazo de cables - Conector de la batería
- 4 x Cable de conexión del motor Gearhead
- Llantas Traxxas Stampede Tera (Firme) (TRC-01)
- 4 x 12vdc 30:1 200rpm (eje de 6 mm) con eje trasero para codificadores opcionales

Figura 68. Robot Rover 4WD01.

Fuente: LYNXMOTION. Products Rovers A4WD1 [en línea]. <http://www.lynxmotion.com/p-603-aluminum-4wd1-rover-kit.aspx

[24]LYNXMOTION. Products Rovers A4WD1 [en línea]. <http://www.lynxmotion.com/p-603-aluminum-4wd1-rover-kit.aspx>

Características del rover:

- Longitud Total: 30,48 cm.
- Ancho promedio: 34,29 cm.
- Altura del neumático: 12,06 cm.
- Longitud del chasis: 24,77 cm.
- Ancho del chasis: 20,32 cm.
- Altura del chasis: 10,16 cm.
- Claridad del piso: 4,14 cm.
- Peso: 1,81 Kilos.
- Velocidad: 45 cm/s

El robot 4WD01 se desplaza mediante tracción por deslizamiento (Skid-steering) gracias a cuatro ruedas (dos a cada lado del robot), equipada cada una de ellas con un motor de corriente continua gearhead.

La configuración Skid-Steer se basa en el control de velocidades relativas y giro de las ruedas, las cuales deben estar fijas y perpendiculares respecto al eje longitudinal del robot. Ya teniendo estas condiciones, el desplazamiento puede ser generado de las siguientes formas:

- Adelante y atrás: El avance se produce por la diferencia de giro entre las ruedas ubicadas en el lado derecho e izquierdo, es decir, si las ruedas en el lado derecho giran con las manecillas del reloj las ruedas en el lado izquierdo deben girar en dirección contraria para generar un desplazamiento frontal.
- Giro y avance: El cambio de dirección se produce por la diferencia de velocidades de las ruedas ubicadas en cada costado, es decir, si las ruedas ubicadas en el lado derecho aumentan su velocidad y las ruedas ubicadas en el lado izquierdo disminuyen su velocidad se provocarán un desplazamiento en dirección izquierda.

- Giro sobre su eje: El cambio de dirección se produce por la sincronización de giro entre las ruedas de cada lado, es decir, si las ruedas ubicadas en el lado derecho e izquierdo giran con las manecillas del reloj provocarán un giro en la dirección izquierda.

5.1 CARACTERIZACIÓN MOTORES

El robot 4WD01 del grupo DicBot está dotado con cuatro motores que corresponden al modelo motor Gearhead, con las siguientes especificaciones técnicas:

Figura 69. Metal Gearmotor 25Dx52L mm HP 12V with 48 CPR Encoder.

Fuente: POLOLU [en línea] disponible en: https://www.pololu.com/product/3242/specs

Figura 70. Características del motor.

Dimensions

Size:	25D x 66L mm
Weight:	104 g
Shaft diameter:	4 mm

General specifications

Gear ratio:	74.83:1
Free-run speed @ 12V:	100 rpm
Free-run current @ 12V:	200 mA
Stall current @ 12V:	2100 mA
Stall torque @ 12V:	125 oz·in
Free-run speed @ 6V:	50 rpm[1]
Stall current @ 6V:	1050 mA[1]
Stall torque @ 6V:	62 oz·in[1]
Lead length:	8 in[2]
Motor type:	2.1A stall @ 12V (MP 12V)
Encoders?:	Y

Fuente: POLOLU [en línea] disponible en: https://www.pololu.com/product/3242/specs

5.2 CARACTERIZACIÓN DE LAS RUEDAS

Las ruedas del robot 4WD01 tienen un diámetro 12,06 cm con un recubrimiento de caucho en la superficie de contacto para evitar cualquier tipo de deslizamiento en terreno arenoso.

Figura 71. Llantas Traxxas Stampede Tera.

Fuente: POLOLU [en línea] disponible en: https://www.pololu.com/product/3217.

5.3 CARACTERIZACIÓN DE LA PLACA-COMPUTADORA RASPBERRY PI.

Es una computadora de placa simple o más conocida como SBC (Single Board Computer) de bajo costo creada en la universidad de Cambridge en 2011. Se trata de una diminuta placa de 85x54 milímetros que funciona como un ordenador de tamaño reducido con la diferencia que no incluye cable de alimentación ni disco duro.

Figura 72. Comparación entre modelos de Raspberry PI.

Características	Raspberry Pi 1	Raspberry Pi 2	Raspberry Pi 3
Chip	Broadcom BCM2835	Broadcom BCM2836	Broadcom SCO BCM2837
Procesador	ARM 1176JZF-S a 700 MHz	ARM Cortex A7, 900 MHz quad-core	ARM Cortex-A43, quad-core a 1.2 GHz
Procesador gráfico	VideoCore IV 520 MHZ OPENGLES 2.0	VideoCore IV 250 MHZ OPENGLES 2.0	VideoCore IV 400 MHZ OPENGLES 2.0
Memoria RAM	256 MB LPDDR SDRAM 400 MHZ	1 GB LPDDR2 SDRAM 450 MHZ	1 GB LPDDR2 SDRAM 450 MHZ
Vídeo	HDMI 1.4 1920x1200	Hdmi 1.4 1900x1200	Hdmi 1.4 1900x1200
Entradas y salidas de vídeo	Conector MIPI CSI, Conector RCA, Conector HDMI	Conector MIPI CSI, Conector RCA, Conector HDMI	Conector MIPI CSI, Conector RCA, Conector HDMI
Entradas y salidas de audio	HDMI, Minijack	HDMI, Minijack	HDMI, Minijack
Puertos USB	Uno (En el modelo B dos; en el modelo B+, cuatro)	Cuatro	Cuatro
Almacenamiento integrado	SD (En el modelo A+, microSD)	MicroSD	MicroSD
Conexión red	Ninguna	10/100 Ethernal vía hub USB	WiFi 802.11n
Bluetooth	No	No	Bluetooth 4.1
Dimensiones	8.5 x 3.5 centímetros	8.5 x 3.5 centímetros	8.5 x 3.5 centímetros
Peso en gramos	45 (El modelo A+, 23)	45	45

Fuente: GADGETOS Paramos todos modelos raspberry [en linea] disponible en: http://www.gadgetos.com/noticias/comparamos-todos-modelos-raspberry/.

La Raspberry Pi 2 es implementada en el proyecto la cual fue proporcionada por el grupo DicBot de la Escuela de Ingeniería Mecánica de la Universidad Industrial de Santander. La Raspberry Pi 2 posee un procesador Broadcom ARM Cotex-A7 de cuatro núcleos y 1 GB de RAM lo cual es seis veces más potente a su antecesora. Además, tiene incorporado 4 puertos USB, una salida de audio y video a través de un conector HDMI y conexión internet 10/100 gracias a los puestos USB donde se puede conectar un adaptador Wi Fi.

Características adicionales:

- Microprocesador:
  - Broadcom ARM Cotex-A7.
  - Toma del procesador BCM2836.
  - 4 núcleos.

- Unidad de procesamiento gráfico:
  - Video Core IV.

- RAM:
  - 1 GB.

- Conectividad:
  - 40 pines GPIO.
  - 1 puerto ethernet 10/100.
  - 4 puertos USB 2.0.
  - 1 conector de video HDMI.
  - 1 conector de cámara CSI.
  - Ranura de tarjeta microSD.

- Alimentación:
  - 5 voltios a 2 A.

- Dimensiones:
  - 12,7x10,2x7,6 cm.
  - Peso de 136 g.

Figura 73. Raspberry Pi 2.

Fuente: RASPBERRYPI [en línea] disponible en: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/.

5.4 FUENTE DE ALIMENTACÓN

La fuente de alimentación es uno de los aspectos más importantes para el buen funcionamiento de un robot, por lo cual se debe establecer el consumo general de los dispositivos que se encuentran trabajando en el robot, con el fin de seleccionar la o las baterías necesarias.

Tabla 7. Especificaciones en el marcado de cada componente implementado.

Descripción	Consumo (mA)	Tensión (V)
Raspberry Pi 2	2000	5
Motores x 4	200	12
Ultrasonido x 3	15	5
Ventilador	120	12
Magnetómetro	0.6	3.3
Puente H L2 98	0 - 36	5

$$I_{total} = I_{Raspberrry} + I_{Motores} + I_{Ultrasonido} + I_{Ventilador} + I_{Magnetómetro} + I_{puente\ H}$$

$$I_{total} = (2000 + 800 + 45 + 120 + 0.6 + 36)mA$$

$$I_{total} = 3001.6\ mA$$

Como factor de seguridad, se aumentó un 25% de consumo de corriente, por lo tanto, el consumo final es:

$$I_{consumo} = I_{total} * 1.25$$

$$I_{consumo} = 3752\ mA$$

En el mercado se encuentran diferentes tipos de baterías (lipo, plomo, gel, etc.), por ello se establecieron tres características para la implementación de la batería:

- Altas prestaciones (alta autonomía).
- Voltaje Variados.

- Volumen pequeño y bajo peso.

Tabla 8. Comparación entre baterías.

Tecnología	Ventajas	Desventajas
Plomo Acido	Proporcionan altas corrientes. Voltaje a 12V	Gran peso. Auto descarga en un mes meses a temperatura de 50°C. Descargas mayores al 20% pueden causar grandes daños a la batería. Permite la fuga de gases tóxicos a la atmosfera. Derrama de acido sulfúrico. No resistente a las vibraciones. Requieren mantenimiento
Gel	Proporcionan altas corrientes. Voltaje a 12V. Mayores tiempos de auto descarga. Permite descargas mayores al 80% de su capacidad (no se recomienda). No permite fuga de gases.	Altos costos. Mayor peso a los de acido. Una batería de estas en una embarcación incita al robo o comercio del mismo.
	Permite inclinaciones de 90° Resistente a las vibraciones. Libres de mantenimiento. Mayores ciclos de carga/descarga.	
Ni-Cd	Descontinuadas	Descontinuadas
NI-MH	Bajo costo.	Voltajes de 1.2 V. Corrientes pequeñas respecto a las Li-Ion y LiPo. No soporta picos de descarga. Ciclos de Recarga entre 300 y 400 veces. Poseen efecto de memoria. Materiales tóxicos.
Li-Ion	Voltajes de 3.2 o 3.7 por célula. Altas corrientes por célula. Soportan picos de descarga. Peso menor. Mayor ciclos de Carga /recarga. No poseen efecto memoria. No son explosivas con respecto a la tecnología Li-Po.	Alto costo. Requiere de cargadores especiales.
LiPo	Voltajes de 3.2 o 3.7 por célula. Altas corrientes por célula. Soportan picos de descarga. Peso menor. Mayor ciclos de Carga /recarga. No poseen efecto memoria.	Costo menor a la tecnología Li-Ion. Peso menor a la tecnología Li-Ion. Requiere de cargadores especiales. Una mala recarga se daña la célula, con riesgos de explosión.

Fuente: ATOCHE, José; GÓNGORA, José; SANDOVAL, Jesús y LUJÁN, Carlos. Selección en tecnología de baterías para alimentación de dispositivos electrónicos portátiles.ECITYDOC.com [base de datos en línea], (mar 2018); p. 491 – 492 [citado en 7 de marzo de 2018]. Disponible en ECITYDOC p 491-492.

Dado estas características, se decidió utilizar baterías de tipo LiPo (Polímero de Litio). La batería LiPo tiene amplia densidad de energía, lo que le permite mantener mayor energía durante más tiempo en comparación con otros elementos químicos

que conforman a otro tipo de baterías. Con una batería de LiPo obtendremos más energía que una batería convencional del mismo peso.[25]

Figura 74. Batería Rhino seleccionada.

Fuente: Dynamo Electronics.

Características:

- Voltaje: 11.1 V 3-cell pack.
- Celdas: Paquete de 3 celdas 2620 mAh.
- Máxima descarga: 3C (7.86A).
- Máxima carga: 2C (5.24 A).
- Dimensiones: 100mm x 32mm x 26mm.
- Peso: 160g.

Debido al alto consumo de corriente requerida por el robot, fue necesario implementar 2 baterías, la primera batería se encuentra dentro del robot y se encargar de suministrar 2752 (mA) de corriente a la Raspberry Pi2, tres sensores ultrasonidos, ventilador y magnetómetro. La segunda batería se encuentra en la parte superior del robot suministrando 1000(mA) de corriente a los cuatro motores y sus respectivos puente H. Estos consumos ya poseen el factor de seguridad del 25%.

Con el fin de poder determinar la duración de cada batería a su máxima carga durante las pruebas, se utilizó la siguiente formula:

[25] TESLABEM. Baterías LiPo - ¿Cómo usar y cuidar una batería LiPo? [en línea]. <http://blog.teslabem.com/como-usar-y-cuidar-las-baterias-lipo/> [citado en 3 de mayo de 2017].

$$Duración\ de\ la\ batería = \frac{Capacidad\ de\ la\ batería\ (mAh)}{Consumo\ del\ circuito\ (mA)}$$

$$Duración\ de\ la\ batería\ 1 = \frac{2620\ (mAh)}{2752\ (mA)} = 0.95\ h \approx 57\text{min}$$

$$Duración\ de\ la\ batería\ 1 = \frac{2620\ (mAh)}{1000\ (mA)} = 2.62\ h \approx 2\ \text{h y}\ 37\text{min}$$

Proporcionando 57 minutos para la primera batería y 2 horas con 37 minutos para la segunda batería.

5.5 MODELO CINEMÁTICO DE MOVIMIENTO SKID STEERING

El movimiento Skid Steering (conducción por deslizamiento) es ampliamente utilizado en robots con llantas o de tipo oruga. La conducción está basada en el control de las velocidades relativas de los motores del lado izquierdo y el derecho[26]. Para que el robot pueda rotar se requiere el deslizamiento o el patinaje de las llantas. Este tipo de movimiento trae alta maniobrabilidad y además no requiere una estructura compleja de dirección lo que le da espacio al robot para tener más accesorios.

A pesar de su simple funcionamiento mecánico, es difícil realizar un modelo cinemático y dinámico que llegue a describir con exactitud el movimiento; esto se debe en parte a que es difícil realizar un modelo predictivo de la forma en que las llantas patinan a partir de los datos de entrada de los sensores propioceptivos. Otra

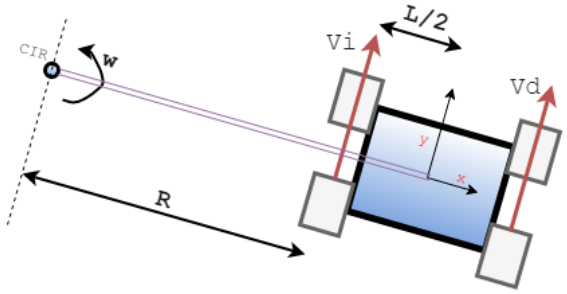
[26] MANDOW, Anthony & MARTÍNEZ, Jorge & MORALES, J & BLANCO, Jose Luis & GARCIA, Alfonso & González-Jiménez, Javier. 2007. Experimental kinematics for wheeled skid-steer mobile robots. 1222 - 1227. 10.1109/IROS.2007.4399139.

desventaja que tiene este tipo de movimiento es el de que las llantas se desgastan rápidamente por el derrape constante.

Los modelos predictivos de las configuraciones en los vehículos móviles más comunes exigen la asunción de que el vehículo no se deslizara y que además todas sus ruedas estarán girando. Estas conjeturas serán descartadas en este modelo. A continuación, se mostrarán las conjeturas necesarias para establecer el modelo cinemático:

- El centro de masa del robot está ubicado en su centroide, en el centro de su estructura.
- Las dos ruedas a cada lado rotan a la misma velocidad.
- El robot está moviéndose en tierra firme y las cuatro llantas siempre están en contacto con el suelo.

Figura 75. Diagrama cinemático para el análisis del movimiento.



Fuente: HELLSTROM, Thomas. (2011). Kinematics equations for Differential Drive and articulated steering. Umea University. Department of Computing science.[En linea] [Citado el 8 marzo de 2018] <http://www8.cs.umu.se/kurser/5DV122/HT13/material/Hellstrom-ForwardKinematics.pdf>

El marco de referencia inercial del robot estará situado en el cuerpo del mismo y el eje Y estará alineado con la dirección de avance del robot. Los vehículos “skid-steered” tienen dos variables de entrada en su movimiento, los cuales son las velocidades lineales del tren de llantas derecho e izquierdo, V_d y V_i respectivamente, con respecto a la velocidad del marco del vehículo. Las ecuaciones que modelan el movimiento de ambos carriles son:

$$\omega\left(R + {}^{l}/_{2}\right) = V_d$$
$$\omega\left(R - {}^{l}/_{2}\right) = V_i$$

Donde L es la distancia entre los centros de ambos trenes de ruedas, R es la distancia del centroide al centro instantáneo de rotación y ω es la velocidad angular del robot, la cual es la misma para ambos trenes en función del CIR.

Al resolver las ecuaciones para R y ω, se obtiene:

$$R = \frac{l}{2}\frac{Vi + Vd}{Vd - Vi}$$
$$\omega = \frac{V_d - Vi}{l}$$

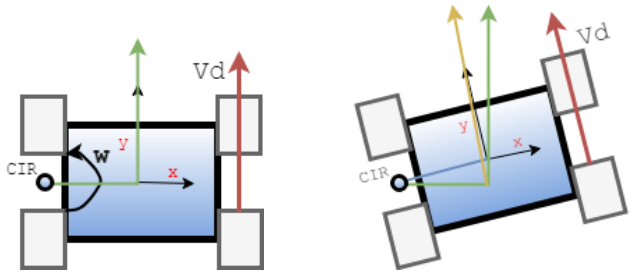
A partir de estas ecuaciones se obtienen tres casos de dirección que definirán el movimiento del vehículo.

- Si la velocidad de ambos trenes es igual, el robot se moverá en línea recta, el centro instantáneo de rotación R tendera a infinito y la velocidad angular será cero.
- Si la velocidad de uno de los trenes es inversa a la del otro, el robot girará sobre su propio eje (eje Z), el centro instantáneo de rotación R será cero, es decir se ubicará sobre su centroide. La velocidad angular será proporcional a dos veces la velocidad de uno de los trenes.
- Si la velocidad de uno de los trenes es cero, el centro instantáneo de rotación será el punto medio de este tren (R=L/2) y su velocidad angular estará determinada por $\frac{V_{tren}}{l}$.

Debido a las dimensiones reducidas de la configuración de trabajo del mapa y a las dimensiones del robot, no se realizará un control en las velocidades de los trenes

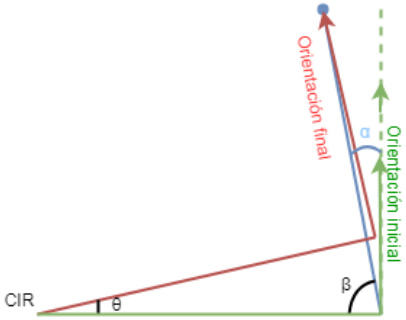
por lo tanto el desplazamiento del robot estará dado por los primeros dos casos mencionados anteriormente. La capacidad de girar sobre su eje permitirá realizar la ruta obtenida del algoritmo de navegación de manera más efectiva debido a los giros con ángulos rectos que se suelen presentar. Sin embargo, el tercer caso es considerado para ejecutar la corrección de la ruta por lo que se realizara el respectivo análisis cinemático.

Figura 76. Desplazamiento del tren derecho.



Para determinar la diferencia del ángulo de orientación entre la posición inicial y la final se requiere despejar las ecuaciones trigonométricas correspondientes a los triángulos generados por ambas posiciones.

Figura 77. Geometría entre las posiciones.



Por medio del teorema del seno y del coseno se obtiene la siguiente relación entre el ángulo θ y el ángulo α.

$$\alpha = 90 - \sin^{-1}\left(\frac{\sin\theta}{\sqrt{2(1-\cos\theta)}}\right) \quad (1)$$

Una vez obtenido el ángulo entre la orientación inicial y la final se buscará la relación del ángulo θ con la cinemática del robot.

$$\omega = \frac{\Delta\theta}{\Delta t} = \frac{V_{tren}}{L}$$

Despejando para θ se obtiene,

$$\theta = \frac{V_{tren} * \Delta t}{L} \quad (2)$$

Resolviendo la ecuación (2) en la ecuación (1) se obtiene la diferencia entre la orientación inicial y final en función del tiempo, esta ecuación será útil para realizar la calibración del protocolo de corrección de la trayectoria de la plataforma.

6. SISTEMA DE LOCALIZACIÓN

Uno de los problemas que afrontan los robots móviles es poder determinar su posición y orientación en el mapa donde se va a desempeñar, debido a que si no saben dónde se encuentran, les resultara imposible saber que tarea o acción deben realizar a continuación. Para poder solucionar este problema se debe suministrar información de su ubicación relativa y absoluta, ya sea a través de diferentes sistemas sensoriales, o por un sistema externo.

6.1 PLANTEAMIENTO DE ALTERNATIVAS

A continuación, se mencionaron algunos sistemas de localización absoluta tomados en consideración.

6.1.1 Sistema de localización por medio de modulo GPS. El uso de un módulo GPS nos permite trazar un marco de referencia inercial global, el cual es poco susceptible a cambios de distancia reducida (se podría reducir el error por medio de la triangulación de varios de estos módulos), debido a su antigüedad la adquisición de datos puede resultar más fácil que las demás alternativas.

Características:

- Fácil adquisición y procesamiento de datos
- El marco de referencia inercial se puede trasladar a cualquier ubicación fácilmente.
- Baja resolución para la magnitud del robot (Hay posibilidad de reducirlo).
- Sistema económico y fácil de instalar.
- Datos en función de la latitud y longitud WGS84.

6.1.2 Sistema de localización por trilateración de posición por medio de balizas estáticas con comunicación Bluetooth BLE. Las balizas son mecanismos emisores de señales de bajo consumo, los cuales se emplearán para determinar la distancia a cada una de estas. Por medio de ecuaciones matemáticas simples se realiza la trilateración de la posición relativa a las balizas, del robot.

Características:

- Sistema de bajo consumo de energía.
- Requiere módulos externos al robot.
- Alta resolución con los filtros de datos adecuados.
- Las balizas serian de fácil transporte.
- Datos en función de la distancia a cada una de las balizas, requiere procesamiento de los datos para localizar el robot.

6.1.3 Sistema de localización con el uso de un filtro de partículas por medio de sensores de distancia. Por medio de un filtro de partículas el robot realizara la deducción de su posición. El filtro de partículas genera posibles posiciones en función de la distancia desplazada y los datos adquiridos de los sensores; durante el movimiento se reducen las posibilidades hasta hallar la posición real. Este sistema requiere un sistema sensorial interoceptivo (Encoders, IMU, etc.) Y exteroceptivo (IR, Sonar, etc.) y un alto nivel de procesamiento de datos en tiempo real.

Características:

- Realiza una localización con altos niveles de exactitud en función exclusiva de datos adjuntos al robot (no requiere módulo ni marco de referencia externo).
- Realizar la ubicación en tiempo real exigiría altos niveles de procesamiento de datos, lo que dificulta la actualización de posición.
- El precio de este sistema seria relativamente bajo debido a que se apoya exclusivamente de los sensores que de igual modo se usaran en la navegación.

6.1.4 Sistema de localización por trilateración de posición por medio de señal Wi-Fi. Wi-Fi es una tecnología inalámbrica que permite la comunicación entre una estación móvil y varios puntos de acceso situados en una misma área de cobertura, enviando entre ellos señales las cuales por medio de ecuaciones matemáticas simples se realiza la trilateración de la posición relativa de los puntos de acceso, a la estación móvil (robot).

Características:

- Sistema de bajo consumo de energía.
- Requiere módulos externos al robot.
- Utiliza red inalámbrica.
- Se requiere las distancias entre los módulos Wi-Fi.
- Datos en función de la distancia a cada uno de los módulos Wi-Fi, requiere procesamiento de los datos para localizar el robot.

6.1.5 Sistema de localización por cámara (Procesamiento de imágenes). Las imágenes recolectadas por la cámara son convertidas de RGB (Red, Green y Blue) a escalas de grises y posteriormente a valores binarios (1 obstáculo y 0 vacío), proporcionando las dimensiones de lo obtenido por la imagen, la ubicación de los obstáculos en el mapa y la ubicación de algún elemento específico dentro de la imagen.

Características:

- Sistema de bajo consumo.
- Se requiere módulo Wi-Fi para envió de imágenes.
- Se requiere calibración de la cámara y estructura para ubicación.
- Campo de visión dependiente de la ubicación de cámara.
- Datos en función de pixeles.

6.2 SELECCIÓN SISTEMA DE LOCALIZACIÓN.

Para la selección del sistema de localización se creó la siguiente tabla con sus debidos criterios de selección.

Tabla 9. Alternativas del sistema de localización.

		ALTERNATIVAS DEL SISTEMA DE LOCALIZACIÓN									
Criterios	%	**GPS**		**BLE**		**FILTRO**		**WIFI**		**CAMARA**	
Rango de cobertura	23%	9	2.1	5	1.1	3	0.7	5	1.1	5	1.1
Resolución	26%	8	2.0	8	2.0	9	2.3	4	1.0	9	2.34
Infraestructura sensorial	30%	2	0.6	8	2.4	8	2.4	5	1.5	8	2.4
Economía	22%	7	1.5	5	1.1	3	0.7	5	1.1	5	1.1
Total		6.2		6.6		6.0		4.7		6.9	

Gracias al análisis realizado se concluyó que la alternativa que cumple de mejor manera los requerimientos de diseño es el sistema de localización por cámara.

El sistema de localización por cámara además puede generar diferentes tipos de información que son suministrar al robot con una sola imagen, proporcionando las dimensiones del mapa con la respectiva ubicación de los obstáculos y del robot, además, determina la orientación por medio de un distintivo en la parte frontal del robot. Todos estos datos están acorde a un plano de referencia (x, y) establecido en la imagen.

6.3 CARACTERIZACIÓN DEL SISTEMA

6.3.1 Cámara. La cámara implementada posee las siguientes características.

Figura 78. Cámara IP 960p Robótica, modelo HW0051.

Fuente: Mercado libre

Características cámara HW0051:

- Conexión Wifi.
- Seguridad del sistema: Nos permite controlar las personas que pueden ingresar a la información suministrada por la cámara por medio de una cuenta y contraseña (Máximo 6 visitantes).
- Sistema operativo: Posee un sistema operativo OS Linux.
- Protocolo soportado: TCP/ IP, http, FTP entre otros.
- P2P: Es una conexión de red de datos entre dos máquinas que se comportan como iguales entre sí. Esto significa que cada máquina puede actuar como servidor y como cliente al mismo tiempo entre los equipos que forman parte de la red P2P.[27] Esto permite el intercambio de archivos entre varios dispositivos conectados a una misma red, en este caso por Wifi.
- Manejo: Permite control por medio de un PC y celulares Android con la APP E-View 7
- Resolución: 1280x960 pixeles
- Parámetros de brillo: Permite ajuste de brillo y contraste
- Angulo de rotación: Horizontal 355° y vertical 88°

[27] MASSEGURIDAD. Conexión P2P en grabadores y cámaras IP [en línea]. [citado el 1 de abril de 2016]. <https://www.masseguridad.es/es/articulos/cctv/692-conexion-p2p-en-grabadores-y-camaras-ip.html>

- Velocidad de rotación: Rango de velocidad de 5-30°/s
- Detección de movimiento: Envió de imagen por protocolo FTP, permitiendo una comunicación directa entra la cámara y el sistema embebido (Raspberry)
- Compatibilidad con el sistema operativo: Microsoft Win XP, Win 7, Win8, Win 10 y Mac OS.

6.3.1.1 Justificación de la cámara. La implementación de cámaras en el campo agrícola ha comenzado a tomar fuerza a comienzos del año 2000 en diferentes partes del mundo incluyendo a Colombia, generando por medio de ellas y gracias a la unión de dispositivos aéreos (drones, aviones, satélites, etc.) fotos y videos del estado en que se encuentra el cultivo sobrevolado, ayudando al agricultor en reducir tiempo a la hora de inspeccionar el cultivo (grandes terrenos) y generando datos difíciles de captar por el ojo humano.

- Vigor del cultivo: Mapa para detectar problemas en cualquier tipo de cultivo
- Planificar una cosecha selectiva: Mapa donde se discrimina diferentes cualidades organolépticas o químicas del fruto.
- Generar un mapa de nitratos: Mapa donde se resaltan las concentraciones de nitratos.
- Generar un mapa de fertilización: Mapa y datos para planificar de manera óptima la fertilización.
- Determinar la gestión hídrica: Mapa de la transpiración y estado hídrico de las parcelas.
- Detección de enfermedades: Mapa para resaltar problemas en relación a plagas.[28]

Un ejemplo se encuentra en España, las imágenes proporcionadas por satélite ayudan a los agricultores a tener un registro del proceso que han desarrollado desde

[28] HEMAV. Drones para agricultura-teledetección agrícola [en línea]. [citado el 22 de mayo de 2015].<https://hemav.com/drones-para-agricultura-teledeteccion-agricola/>

el momento de la siembra hasta la cosecha, “Este mecanismo se usa para justificar que un agricultor no la cantidad incorrecta de agua, debido a que las dotaciones están definidas y se usan como herramienta de testigo (imagen satelital). Por ejemplo, si usted tiene derecho a 10 mil metros cúbicos solamente, con las imágenes se demuestra que el agua es utilizada en una superficie regable con esa cantidad y no más”[29]

Otro lugar donde se encuentran realizando investigaciones por medio de cámaras y drones es en Palmira, Valle del Cauca en cultivos de arroz. Estos investigadores por medio de un dron equipado con dos cámaras, una multiespectral y otra cámara RGB, “desean identificar cuáles variedades de arroz se comportan mejor ante la sequía y aprovechan de manera eficiente el nitrógeno que se aplica para hacer rendir el cultivo”[30]. Con la implementación de esta tecnología, desean reducir el método de inspección utilizado anteriormente, el cual constaba en un recorrido por el cultivo seleccionando plantas y realizándoles inspecciones a ojo, lo cual variaba de investigador a investigador generando problemas de recolección de información.

La cámara cumple la función de un dron que estuviere analizando un campo agrícola real, por lo cual la cámara seleccionada se encuentra posicionada por medio de una estructura como se aprecia en la figura 77, generando una posición estable y segura de la cámara para la toma de una imagen adecuada que abarque toda la información del terreno establecido con todos los componentes que se encuentren dentro (surcos, obstáculos y robot).

[29] EL MERCURIO. Teledetección, una herramienta para hacer más eficiente las tareas agrícolas. [en línea]. <http://www.elmercurio.com/Campo/Noticias/Noticias/2014/04/23/Teledeteccion-una-herramienta-para-hacer-mas-eficiente-las-tareas-agricolas.aspx> [citado en 6 de marzo de 2015].
[30] BETANCUR, Laura. Los drones son los nuevos ‘ojos’ de los científicos de la agricultura: Dron de ocho hélices capta imágenes de alta calidad de los cultivos del Valle del Cauca. El Tiempo [en línea] (2016) http://www.eltiempo.com/vida/ciencia/drones-usados-en-la-agricultura-39867> [citado en 14 de diciembre de 2016].

Figura 79. Estructura de la cámara. a) vista frontal, b) vista lateral.

Características:

- Altura: 5 metros.
- Ancho: 1,45 metros
- Profundo: 2,3 metros

6.4 IMPLEMENTACIÓN DEL SISTEMA

El sistema de localización por cámara implementado se encuentra dividido en tres etapas mencionadas a continuación.

- Adquisición digital de imagen: Durante esta etapa se adquiere la información a través del equipo electrónico, en donde se reconstruye de forma digital en función de las características producidas por la luz que captan los lentes receptores, para finalmente ser almacenados en la computadora.
- Adecuación de la imagen: Conociendo los factores que afectan las imágenes (enfoque, iluminación, contraste, etc.), la manera que estos inciden sobre el entorno y finalmente en la información adquirida por los equipos, se debe realizar

un proceso que corrija estos factores con el fin de facilitar el procesamiento final de la imagen.

- Procesamiento digital de la imagen: Por medio de herramientas computacionales, se desarrollarlo e implemento un método que este en capacidad de identificar de manera autónoma la información importante (mapa y ubicación del robot) y desechar la información irrelevante a las necesidades del usuario.

6.4.1 Adquisición digital de imagen Una imagen digital es una función de dos dimensiones acotada tanto en dominio como en rango (tiene valores finitos en resolución y cuantización), cada posición en la imagen es conocida como pixel. Dando así la conversión de una escena en 3D a un plano 2D

Para poder obtener dicha imagen se tuvieron que seguir los siguientes pasos:

- Activación del sistema: Alimentación de energía a la cámara ubicada en la parte superior del terreno, conexión a una red Wifi segura y a dispositivos receptores (celular, PC y Raspberry) de la información a enviar.
- Toma de imagen: Una vez posicionado el vehículo dentro del mapa, se procede a la toma de una imagen de resolución 1280x960 pixeles del tipo RGB del terreno con todos los componentes internos que contenga.
- Envió de información: La imagen es recibida a través del protocolo de comunicación FTP (protocolo de transferencia de archivos), el cual configura la Raspberry como servidor principal de la cámara, lo que significa que la información generada por la cámara será almacenada inmediatamente en la memoria externa de la Raspberry gracias a la conexión wifi.

6.4.2 Adecuación de la imagen. Una vez recibida la imagen en la dirección de la memoria estipulada, se procede a ejecutar el código de extracción de información deseada realizando las siguientes etapas.

6.4.2.1 Extracción de información. La imagen proporcionada por la cámara posee una forma rectangular (1280x960 pixeles) y el terreno de forma cuadrada (5m x 5m), por lo cual es necesario eliminar una parte de la imagen con información irrelevante por medio de una máscara como se aprecia en la figura 78, haciendo énfasis que el terreno se debe ubicar simétricamente en el centro, esto se logra gracias a la ubicación de la cámara.

Figura 80. Máscara para eliminar información irrelevante.

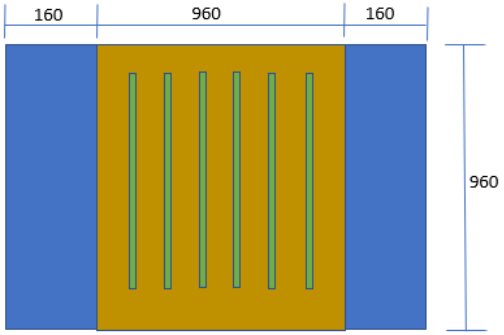
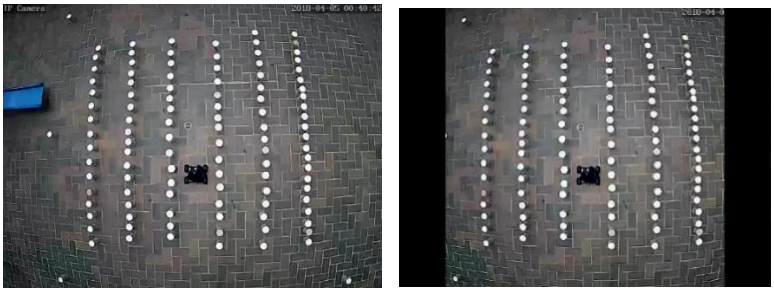


Figura 81. a) Imagen principal, b) Imagen con máscara.



6.4.2.2 Imagen RGB a escala de grises Una imagen RGB, “es representado por las combinaciones de los colores rojo (Red), verde (Green) y azul (Blue). Para poder representar este tipo de imágenes se utiliza una matriz tridimensional, MxNx3 en

donde M y N representan el alto y ancho de la imagen y cada dimensión de esta matriz representa la intensidad de cada uno de sus canales R, G, B."[31]

Figura 82. Combinación de colores RGB.

Fuente: PINTEREST [en línea] disponible en: https://www.pinterest.com.mx/pin/296041375490650713/.

Cada pixel ubicado dentro de la imagen tiene un valor asignado por las tres matrices, por lo cual a mayor intensidad de color obtendrá un valor alto y a menor intensidad un valor bajo.

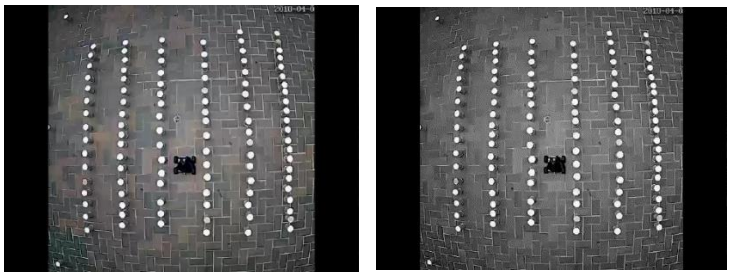
Ya asignados los valores de cada pixel dentro de la imagen se procede a realizar la transformación a escala de grises por medio de la librería OpenCV de Python, permitiendo asignar a cada pixel una tonalidad de gris partiendo desde el valor 0 para un color negro hasta 255 para un color blanco.

Figura 83. Ejemplo de imagen de RGB a escala de grises.

Fuente: http://www.cristalab.com/tutoriales/los-modos-de-color-en-photoshop-c105006l/.

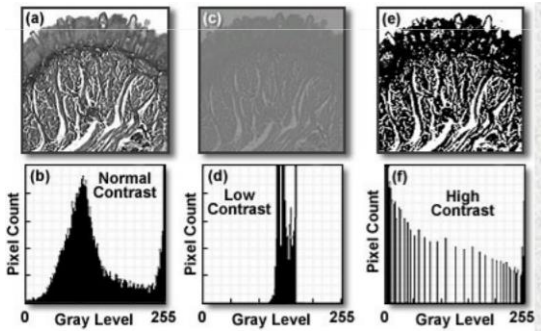
[31] BUSTOS, Nery, GODÍNEZ, Julio. Navegación de un robot móvil aplicando campos potenciales y reconocimiento de objetos usando on Android. Ciudad de México, 2016. p.34.

Figura 84. a) Imagen RGB, b) Imagen en escala de grises.



6.4.2.3 Revisión del histograma "Un histograma es una medida de frecuencia en la ocurrencia de cada valor o rango de valores de nivel de intensidad de gris dentro de una imagen. Usualmente se utiliza la frecuencia relativa, representando la probabilidad de ocurrencia para ese nivel de gris, pero también puede utilizarse una frecuencia absoluta dependiendo del uso que se le quiera dar."[32]

Figura 85. Imágenes con su respectivo Histograma.



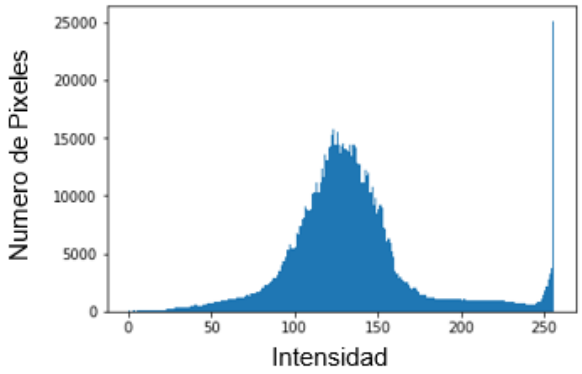
Fuente: Introducción a las imágenes digitales.

La imagen obtenida en escala de grises se puede presentar de diferentes contrastes como se puede apreciar en la figura 85, pero debido a las características propias de las imágenes, no fue necesario realizar el proceso de ecualización debido que la luminosidad de la escala de grises era uniforme proporcionando un contraste adecuado. A pesar de esto es necesario comprobar el estado del histograma de

[32] RUIZ, Javier. Filtrado de imágenes y ecualización de histogramas. Santiago de Chile: 2015. p.4.

intensidad con el fin de ofrecer una imagen adecuada antes del procesamiento de la imagen.

Figura 86 Calculo del Histograma.



6.4.2.4 Filtro espacial El filtrado espacial tiene como objetivo modificar los rangos de intensidad que posee cada pixel frente a los pixeles que lo rodean. Para este proceso se toma el valor del pixel a analizar y el de sus vecinos inmediatos, el promedio ponderado de la intensidad con respecto a la proximidad relativa entre ellos genera el nuevo valor que debe poseer el pixel analizado. Este proceso se realiza pixel por pixel hasta finalizar la imagen.

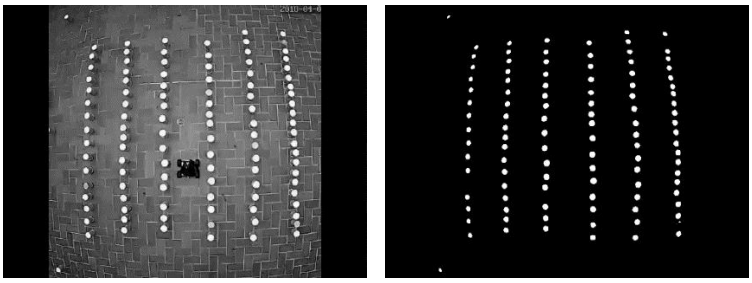
Este filtro es utilizado con la finalidad de reducir ruido, diferencia de intensidad entre pixeles y realce de bordes que se localizan en la imagen, suavizando la imagen.

El filtrado espacial implementado es de tipo gaussiano proporcionado por la librería Python OpenCV.

6.4.3 Procesamiento digital de la imagen. Una vez corregida la imagen, se procede a dividir la información de acuerdo con unos parámetros de evaluación y términos discriminativos para identificar los datos de interés, extraerlos y finalmente usarlos en el sistema de navegación.

6.4.3.1 Imagen binaria. El histograma de la imagen ecualizado permite mantener las propiedades esenciales de la imagen, dando la posibilidad de convertir los componentes de la imagen en dos valores, blanco y negro para tener un mejor proceso y lectura de la información. Para poder realizar este procedimiento primero se procede a binarizar la imagen eligiendo un umbral dentro de los niveles de grises suministrado por el histograma, asignando un valor de cero (negro) al rango de grises entre 0-200 y un valor de 1 (blanco) al rango de grises entre 201-255.

Figura 87. a) Imagen en escala de grises, b) Imagen binaria.



Para tener un mejor manejo de la información proporcionada, es necesario establecer una constante de conversión entre la unidad de medida de la imagen (pixeles) con el del terreno (metros). Por lo que se establece la altura de la imagen (960 pixeles) como igual medida al costado del terreno (500 cm), generando de esta forma una conversión entre los dos sistemas de medición. Para lograrlo es necesario cambiar la forma geométrica de la imagen, por lo cual se realizó un corte de 160x960 pixeles en cada costado con el fin de obtener una imagen cuadrada de 960x960 logrando de esta forma poder realizar la conversión de unidades.

$$Punto(cm) = \frac{(\ Pixel * 500cm\ )}{960\ pixeles}$$

6.4.3.2 Reconocimiento de forma. Una vez obtenida la imagen binarizada, se aplica una máscara sustractiva, la cual posee las dimensiones y características ideales del mapa conocido, dando como resultado una nueva imagen con algunas

imperfecciones, que se pudieron generar durante el proceso de la toma de imagen hasta la binarización, y el área de una figura rectangular, la cual corresponde al robot.

Para detectar correctamente la ubicación del robot, se crea un rango aproximado de las dimensiones de él, con el fin de descartar cualquier otro objeto que tenga dimensiones diferentes, por lo cual se colocó un margen de error de 2.500 a 6.000 pixeles. Ya detectado el robot, se procede a encontrar el centroide en coordenadas X y Y proporcionadas en pixeles para ser transformadas posteriormente en metros gracias a las siguientes ecuaciones.

$$X(cm) = \frac{(\,Pixel\_X - 160) * 500cm\,)}{960\ pixeles}$$

$$Y(cm) = 500 - \frac{(\,Pixel\_Y * 500cm\,)}{960\ pixeles}$$

Figura 88. a) Imagen en escala de grises, b) Imagen de reconocimiento del robot.

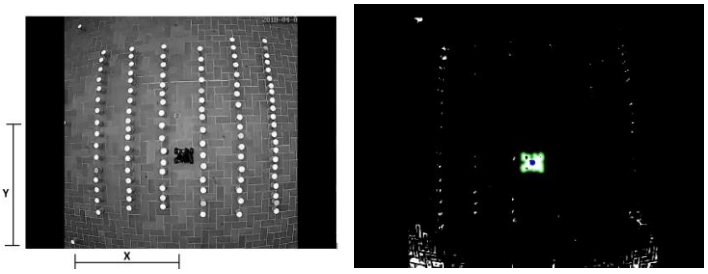


Figura 89. Distancia en pixeles y en metros.

```
('x en pixeles ', 650)
('y en pixeles ', 599)
('x en centimetros', 255)
('y en centimetros', 189)
```

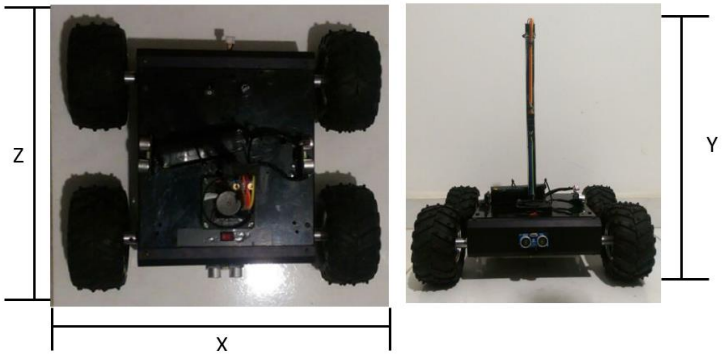
7. PRUEBAS Y RESULTADOS

Las pruebas que se documentaran a continuación tienen como fin evaluar los diferentes sistemas y verificar el cumplimiento de su función, además se documentara como la sinergia sensorial de los sistemas contribuyen en la creación de un sistema autónomo. Las pruebas serán verificadas y documentadas con parámetros cuantitativos con el fin de respaldar los resultados.

7.1 PRUEBAS ELÉCTRICAS Y MECANICAS

7.1.1 Peso y dimensiones finales Realizada las modificaciones necesarias a la plataforma robótica, las nuevas dimensiones son:

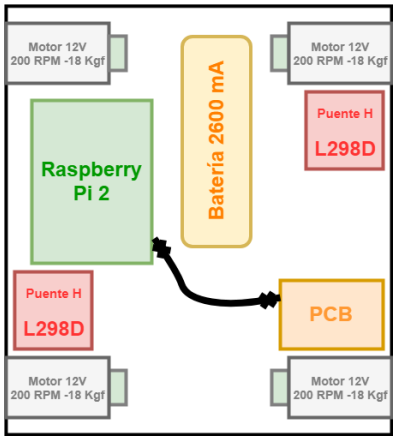
Figura 90. Dimensiones finales de la plataforma robótica.



- Altura (y): 40 cm.
- Ancho (x): 35 cm.
- Largo (z): 30 cm.
- Peso: 5,4 Kg.

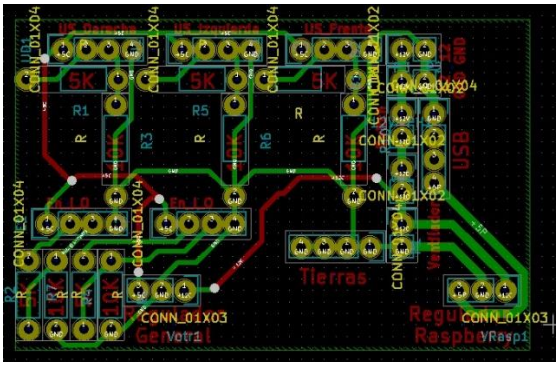
7.1.2 Integración de los elementos electrónicos A continuación, se describirá la estructura interna del robot, así como también la forma en que los sistemas periféricos fueron conectados para su correcto funcionamiento.

Figura 91. Diagrama de la configuración interna de los elementos electrónicos.



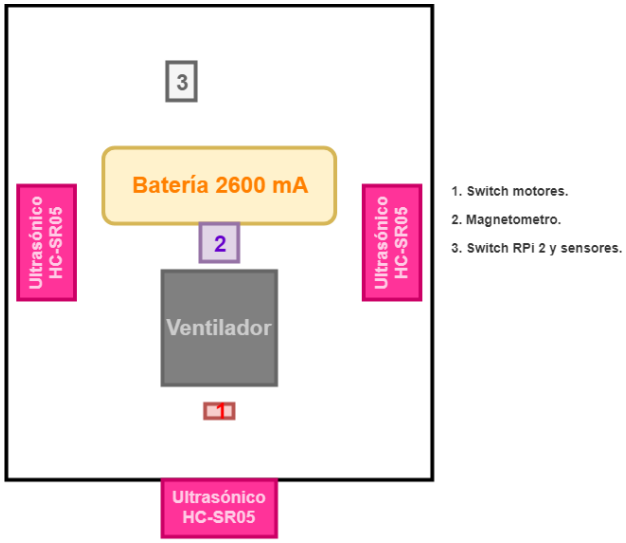
La plataforma robótica dispone de dos fuentes de alimentación separadas; una para la Raspberry Pi, los sensores y el ventilador, y la segunda para los cuatro motores. La fuente de alimentación interna está controlada por dos reguladores lineales de tensión fija L7805CV, el cual regula la salida de 12V de la alimentación a una de 5V para los periféricos. Estos reguladores están situados en un circuito impreso PCB diseñado por los autores. En este circuito impreso se conectarán los sensores de distancia, el ventilador y la Raspberry Pi.

Figura 92. Diseño de la PCB.



El magnetómetro al ser un sensor de bajo consumo y trabajar a 3.3V, puede ser alimentado directamente desde la Raspberry Pi.

Figura 93. Diagrama de la configuración externa de los elementos electrónicos.



Sobre la plataforma exterior del robot se encuentra los dos sensores ultrasónicos laterales, la segunda fuente de alimentación, el ventilador y los switches de encendido y apagado de todos los sistemas. En la parte frontal del chasis del robot se sitúa el tercer sensor ultrasónico.

El magnetómetro se encuentra situado sobre una antena acrílica de 30 centímetros de longitud con el fin de que las mediciones no sean perturbadas por el campo magnético de los motores u otros objetos metálicos.

7.1.3 Corriente consumida por los elementos periféricos Se realizaron pruebas eléctricas de los sistemas eléctricos con el fin de determinar el consumo real de cada uno de los periféricos alimentados por las baterías, además se realizará un análisis del consumo durante las ejecuciones.

Figura 94. Consumo medido de cada componente implementado.

Descripción	Consumo medido (mA)
Raspberry Pi 2	1200
Motores x 4	250 – 500 (A)
Ultrasonido x 3	6.9
Ventilador	112 – 1000 (A)
Magnetómetro	0.66
Puente H L2 98	0.6 – 3.6 (A)

*** (A) Corriente de arranque.**

$$I_{total} = I_{Raspberrry} + I_{Motores} + I_{Ultrasonido} + I_{Ventilador} + I_{Magnetómetro} + I_{puente\ H}$$

$$I_{total} = (1200 + 1000 + 20.7 + 112 + 0.66 + 3.6)mA$$

$$I_{total} = 2336.96\ mA$$

Tras la ejecución de numerosas pruebas se pudo determinar empíricamente un factor de uso de las baterías de la plataforma robótica con el fin de determinar el tiempo de vida útil de manera más exacta.

La duración de cada batería durante las pruebas es:

$$Duración\ de\ la\ batería = \frac{Capacidad\ de\ la\ batería\ (mAh)}{Consumo\ del\ circuito\ (mA)}$$

$$Duración\ de\ la\ batería\ 1 = \frac{2620\ (mAh)}{1337\ (mA)} * 0.8 = 1.57\ h \approx 1\text{h y } 34 \text{ minutos}$$

$$Duración\ de\ la\ batería = \frac{2620\ (mAh)}{1000\ (mA)} * 0.9 = 2.35h \approx 2\text{h y } 21 \text{ minutos}$$

La Raspberry Pi, al ser la placa de control consume la cantidad más alta de energía y además es proclive a variaciones en el consumo, de acuerdo con el procesamiento que realiza y a la temperatura que esta llega a alcanza; lo que le permite llegar a consumir 2 amperios para la ejecución de grandes procesos.

La influencia de la descarga de la batería se aprecia en las mediciones realizadas por los sistemas perceptivos, los cuales obtienen información errónea e incompleta o incluso pueden llegar a fallar. Se recomienda cargar ambas baterías completamente para la ejecución exitosa de una trayectoria.

7.2 PRUEBAS DEL SISTEMA DE PERCEPCIÓN

El método de cálculo de la posición relativa del robot se le llama localización a la estima y este se basa en el uso de los sensores perceptivos para calcular la posición y orientación actual del robot con base al conocimiento de la situación anterior y una razón de cambio aproximada. A continuación, se realizarán las pruebas de cada uno de los sensores propioceptivos para la calibración y posterior realización de su función en el método de localización a la estima. Luego se analizará la capacidad de corrección de ruta del robot por medio del uso de los sensores de distancia (sensor exteroceptivo).

7.2.1 Encoder. La función que tiene el encoder en la plataforma robótica es la de realizar un conteo de las ranuras leídas por el sensor de efecto hall para posteriormente traducirlos a una distancia de recorrido. La debida calibración de este sensor es importante para que realice la ruta exitosamente puesto que el algoritmo principal de movimiento de la plataforma robótica requiere del conteo centímetro a centímetro de la ruta otorgada por el algoritmo de navegación.

Para comprobar la eficacia del desplazamiento a la estima respaldado por el encoder se propusieron trayectorias rectas de diferentes longitudes para determinar la cantidad de error que el encoder puede llegar a acumular. Para realizar estas mediciones se utilizó un flexómetro.

Figura 95. Prueba para determinar exactitud de los encoders.

Tabla 10. Resultados de cada ruta.

Ruta de 1m	
Medición	**% de error**
101.0	1.00%
100.5	0.50%
99.05	0.95%
101.0	1.00%
100.1	0.10%

Ruta de 2m	
Medición	**% de error**
202.4	1.20%
198.1	0.95%
197.5	1.25%
195.0	2.50%
198.5	0.75%

Ruta de 4m	
Medición	**% de error**
405.2	1.30%
401.5	0.38%
392.5	1.88%
395.0	1.25%
396.3	0.92%

La desviación estándar de los datos obtenidos en cada prueba es:

$$\sigma_{1m} = 0.81\ cm\ ;\quad \sigma_{2m} = 2.67\ cm\ ;\quad \sigma_{4m} = 5.15\ cm$$

El error en los encoders incrementa a medida que las longitudes que debe contar aumentan. Se observa que el robot se desplaza distancias menores a las esperadas en la mayoría de los casos lo que significa que está contando una cantidad de ranuras de manera prematura.

La dispersión de los datos obtenidos aumenta entre las pruebas, lo que confirma el principal error de los encoders el cual se observa en el conteo incorrecto de ranuras por distancias relativamente largas.

A demás de los errores propios del encoder, se sugieren otra serie de posibles causas de falla en la prueba realizada: la inercia del robot hace que este se desplace más de lo requerido o el posible deslizamiento de la rueda con el encoder activo.

7.2.2 Magnetómetro La función que tiene el magnetómetro en la plataforma robótica es la de determinar la orientación, lo que posteriormente será utilizado para ejecutar los giros. La calibración automática y previa a la ejecución de la ruta determinara los valores de compensación que requiere el magnetómetro para entregar las medidas del campo magnético en grados adecuadamente.

Para comprobar la eficacia de la rotación a la estima respaldado por el magnetómetro, se propusieron giros de 90 grados en ambas direcciones para determinar el porcentaje de error posible al realizar los giros. Para realizar estas mediciones se utilizó un flexómetro.

Figura 96. Giro demostrado con el marcador.

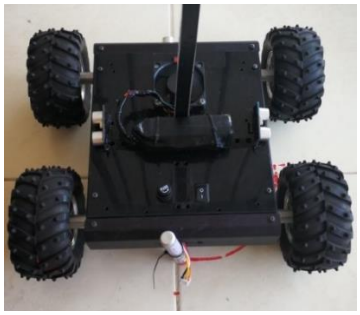


Tabla 11. Resultados pruebas de giro.

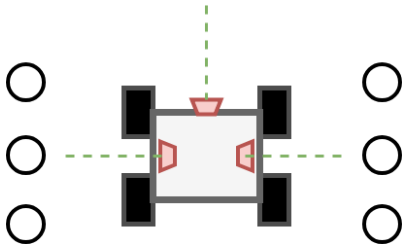
Giro 90° derecha	
Medición	**% de error**
89.1	1.00%
94.2	4.67%
92.1	2.33%
95.3	5.89%
92.1	2.33%

Giro 90° izquierda	
Medición	**% de error**
91.1	1.22%
90.5	0.56%
96.3	7.00%
97.2	8.00%
95.2	5.78%

A demás del error en la medición del magnetómetro se debe considerar la inercia del robot al realizar los giros y los tiempos de respuesta retrasados que puedan llegar a ocurrir.

7.2.3 Sensores de distancia Los sensores de distancia cumplen dos funciones en la ejecución del movimiento de la plataforma robótica, la primera es la de corregir la ruta del robot con el fin de que este se mantenga en el centro de la hilera y no se acerque o colisione con las plantas, esta función es ejecutada por todos los sensores. La segunda función es la de detener la plataforma robótica oportunamente en caso de que encuentre un obstáculo enfrente de ella.

Figura 97. Medición de los sensores dentro del surco.



Para determinar el tiempo en el que cada nivel de corrección será activado se utilizaran las ecuaciones obtenidas en el capítulo de locomoción de este mismo libro:

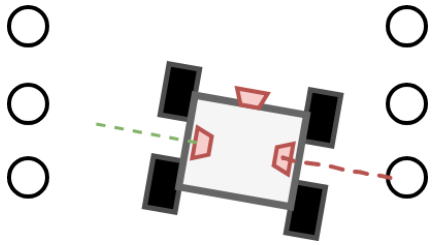
$$\theta = \frac{V_{tren} * \Delta t}{L} \quad (2)$$

$$\alpha = 90 - \sin^{-1}\left(\frac{\sin\theta}{\sqrt{2(1-\cos\theta)}}\right) \quad (1)$$

El protocolo de corrección de ruta tiene dos niveles de corregimiento los cuales están en función del tiempo que uno de los trenes se mueve mientras el otro se detiene, lo que provoca un giro sobre el centro instantáneo de rotación que en este caso se ubica en el punto medio del tren que se detiene.

El primer nivel de corrección se activará cuando la distancia calculada por alguno de los sensores laterales sea menor a 20 centímetros. El tiempo de corrección es de 0.02 segundos, lo que teóricamente se traduce a un ángulo de 0.82 grados de rotación por medio de las ecuaciones de cinemática (1) y (2).

Figura 98. Primer nivel de corrección.



Este valor teórico se comparó con los valores de corrección obtenidos por la plataforma robótica en ejecución.

Figura 99. Primer nivel de corrección realizado por la plataforma robótica.

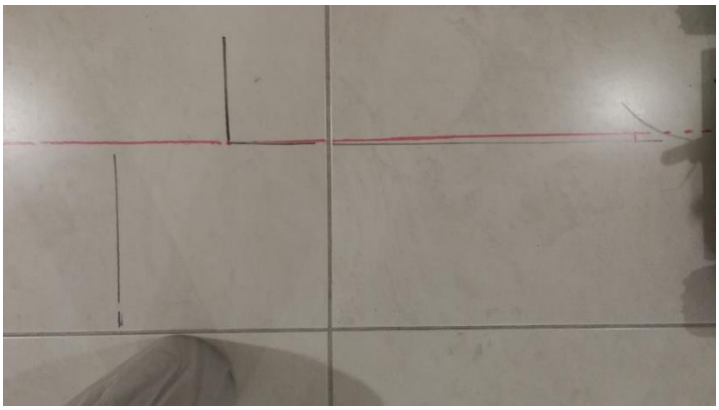


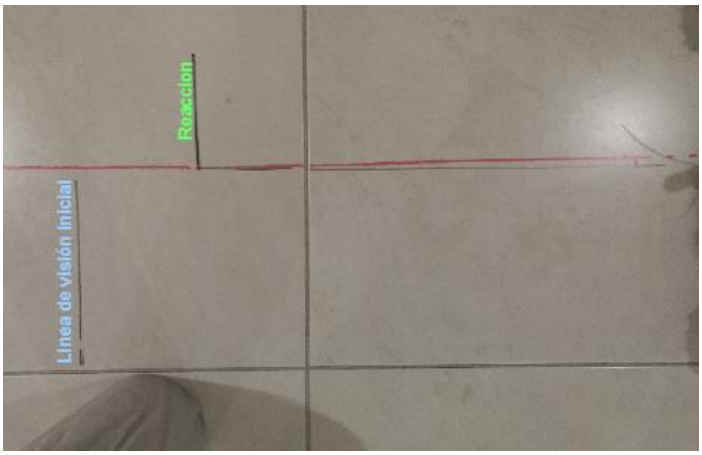
Tabla 12. Resultados primer nivel de corrección.

Corrección de primer nivel	
Medición	**% de error**
1.45	78.56%
0.95	17.28%
1.01	24.73%
1.34	65.43%
1.28	58.02%

La corrección de primer nivel llega a corregir la trayectoria del robot un promedio de 1.2 grados, lo que tiene una diferencia del 50% con el valor obtenido teóricamente. Esto se le puede atribuir al tiempo de respuesta retrasado y a la variación del funcionamiento de los actuadores y sensores bajo diferentes niveles de voltaje.

En la figura 97 se puede observar la distancia recorrida por el robot desde que el obstáculo atraviesa la línea de visión del sensor lateral hasta que este reacciona.

Figura 100. Reacción ante un obstáculo lateral.



A partir de esta distancia, las dimensiones del robot y la velocidad promedio de este, se determinará el tiempo de respuesta aproximado para la corrección de los obstáculos.

$$Distancia\ Recorrida = D_{sensor-marcador} + D_{recorrida\ antes\ de\ reaccion}$$

$$Distancia\ Recorrida = 13 + 11 = 24\ cm$$

$$Tiempo\ de\ respuesta = \frac{Velocidad}{Distancia} = \frac{42.4}{24} = 1.76\ s$$

Tras la ejecución de varias pruebas se observó que un nivel de corrección no era suficiente debido a que el tiempo de respuesta era alto. Por lo que se decidió implementar un segundo nivel de corrección que analice un mayor rango de distancia y a la vez corrija la ruta con un ángulo más grande.

El segundo nivel de corrección se activa cuando la distancia de uno de los sensores laterales sea menor a 35 centímetros y además la distancia del sensor frontal sea menor a 70 centímetros. El tiempo de corrección es de 0.2 segundos, lo que teóricamente se traduce a un ángulo de 8.21 grados por medio de las ecuaciones de cinemática (1) y (2).

Figura 101. Segundo nivel de corrección.

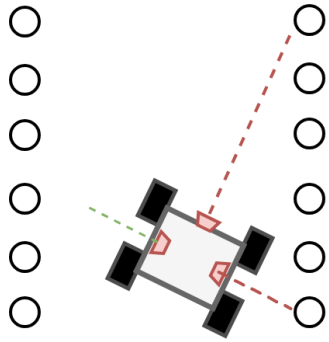


Figura 102. Segundo nivel de corrección realizado por la plataforma robótica.

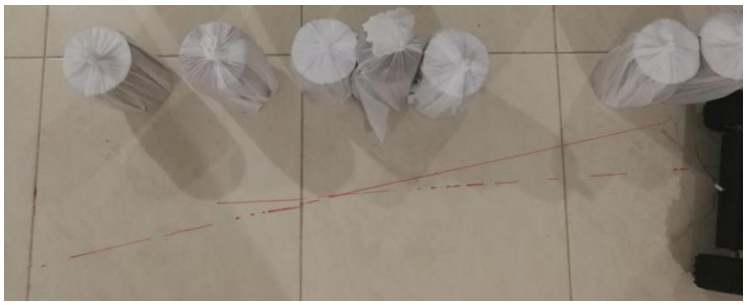


Tabla 13. Resultados segundo nivel de corrección.

Corrección de segundo nivel	
Medición	**% de error**
6.34	22.82%
6.39	22.12%
8.65	5.40%
5.02	38.87%
7.32	10.84%

El segundo nivel de corrección logro solucionar el problema del ajuste mínimo realizado por el de primer nivel. Este nivel de corrección presento un porcentaje de error menor a comparación del primer nivel, lo que se le puede atribuir a la inercia en el cambio de dirección.

Para la calibración del protocolo de detección de obstáculos realizado por el sensor frontal se realizaron pruebas que buscan analizar el tiempo de respuesta del robot. Para esto se encontrará la diferencia entre la distancia de detección del algoritmo y la posición final del sensor respecto del obstáculo.

Figura 103. Estructura de la prueba del sensor frontal.

Tabla 14. Resultados tiempo de respuesta.

Distancia (cm)	Tiempo de respuesta (s)
16.8	0.396
10.5	0.247
16.5	0.389
10.8	0.254
6.7	0.158
6.5	0.153

El tiempo de respuesta obtenido en esta prueba varía en gran medida, por lo que establecer un tiempo de respuesta promedio sería incorrecto. Una de las razones que para esta dispersión de los datos es el tiempo de muestreo del algoritmo de los sensores de distancia, el cual se llega a retroalimentar cada 0.05 segundos y está ligado con la diferencia de tiempo entre la emisión de la señal y la recepción.

7.3 PRUEBAS DEL SISTEMA DE LOCALIZACÓN

El sistema de localización debe satisfacer la necesidad de capturar el mapa a trabajar por medio de una imagen RGB, la cual recibe un procesamiento digital generando una imagen del campo en binaria y la ubicación más exacta posible del robot, por lo que fue necesario realizar cinco procesos: calibración de la cámara, ubicación de la cámara, ajuste de intensidad de luz producida por el exterior, determinación del umbral binario para el mapa y umbral para la ubicación del robot.

- Calibración de la cámara: La cámara ofrece la opción de acercar o alejar la imagen a voluntad, permitiendo ubicarla en una posición considerable por medio de una estructura mencionada en el capítulo anterior. Para determinar la calibración y altura adecuada, se realizaron tomas a diferentes alturas, obteniendo que la altura adecuada y por facilidad de construcción de la estructura, se debe ubicar a una altura de 5 metros, debido a que si la cámara se encuentra a una altura menor provoca una distorsión de barril en la imagen. Esto se puede apreciar en los surcos de la imagen, dándoles una forma curva y no lineal a como se encuentran en realidad.

Figura 104. Opción de calibración de imagen.

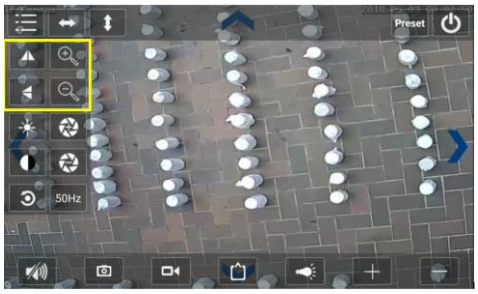


Figura 105. a) Cámara ubicada a 3m, b) Ubicada a 5m.

- Ubicación de la cámara: Después de calibrar y determinar la altura ideal, procedemos a buscar la ubicación adecuada de la cámara para la toma de fotos, por ende, se realizan diversas tomas en diferentes posiciones, dándole prioridad a la captura total del mapa sin deformación alguna; se concluyó que: el punto medio del campo es la ubicación en la que las tomas son las más aptas para realizar las pruebas, como se aprecia en la figura 104 c). Las demás ubicaciones (a un costado y sobre el mapa), muestran distorsiones del mapa como lo son: reducción de los espacios libres, imprecisión en la ubicación y avance del robot.

Figura 106. a) Cámara ubicada sobre el mapa, b) Cámara ubicada a un costado del mapa, c) Cámara ubicada en el centro del mapa.

- Intensidad de luz: Uno de los grandes problemas obtenidos en la toma de fotos fue el ajuste de la luz incidente, debido a que un día soleado provocaba un brillo sobre todo el mapa dificultando la distinción de las hileras, o un día nublado dificultaba la ubicación del robot. Generando modificaciones pertinentes al momento de la toma y análisis de fotos (Calibrar los umbrales binarios) ingresadas en el robot. Para poder establecer una intensidad de luz adecuada

en la imagen, la cámara permite ajustar la intensidad de luz incidente con el fin de obtener una imagen nítida con los parámetros necesarios para un buen funcionamiento del robot.

Figura 107. Opción de ajuste de intensidad de luz.

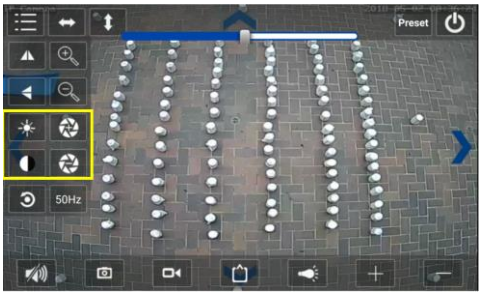
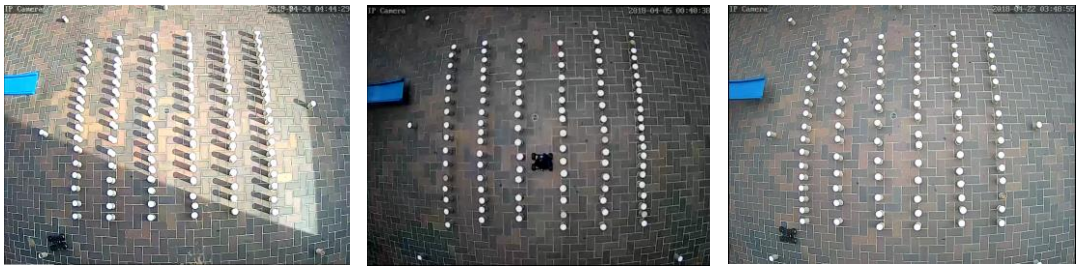


Figura 108. a) Imagen con día soleado, b) Imagen con día opaco, c) Imagen calibrada.



- Umbral binario para el mapa: Una vez establecidas las condiciones de la imagen a procesar, se procede a determinar el umbral adecuado para poder detectar el mapa a trabajar, como se explica en el capítulo 5, la imagen un procesamiento digital de imagen de RGB a binario (ceros y unos) en el cual después de la toma de varias fotos a diferentes horas del día, se estableció que el umbral adecuado se debe mantener en 190, enfatizando que si se encuentra el pixel en escala de grises dentro del rango 0 - 190 se convierte a 0 (color negro significando espacio libre) y dentro 191 - 255 se convierte a 1 (color blanco significando surcos y obstáculo).

Figura 109. Mapa original.

Figura 110. a) Mapa por debajo del umbral (150), b) Mapa por encima del umbral (230).

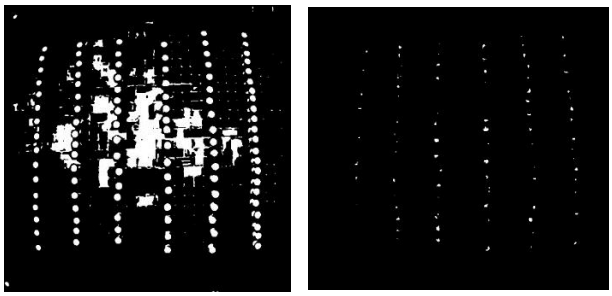
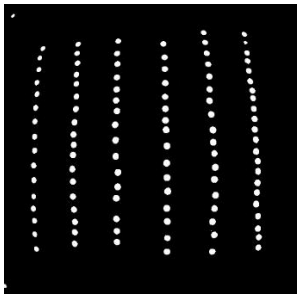


Figura 111. Mapa con el umbral adecuado (190).

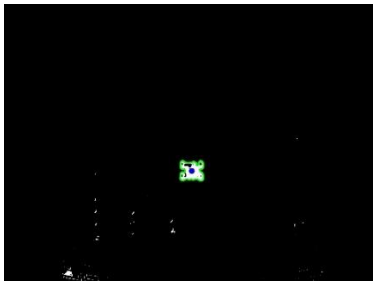


- Umbral ubicación del robot: El umbral utilizado para la detección del robot es diferente al implementado con el mapa, ya que, si se mantiene el mismo, se presentarán figuras amorfas que poseen la misma tonalidad (sombras), generando de esta forma varios centroides debido a que se encuentran en el rango de área establecido. Por esta razón se implementó un umbral más bajo

de 180, proporcionando una imagen más limpia como se aprecia en las siguientes figuras.

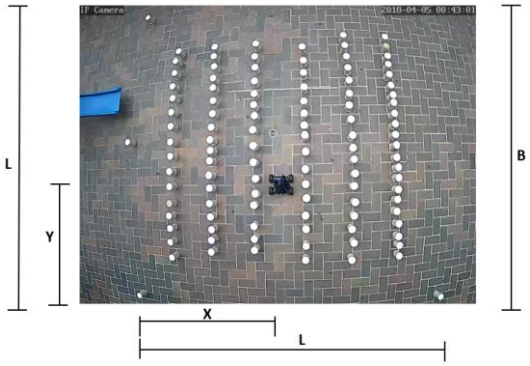
Figura 112. a) Mapa original, b) Umbral del robot igual al del mapa.

Figura 113. Umbral ideal para la ubicación del robot.



7.3.1 Pruebas Se procedió a realizar la toma de imágenes sobre un campo de 5x5 metros (representado por los conos) con la ubicación del robot en cinco posiciones diferentes. Por cada ubicación se tomaron cinco fotos con el fin de comprobar la extracción del mapa binario y estimar la exactitud del sistema de localización del robot, por lo cual con la ayuda de un flexómetro se midió la posición real (X y Y) respecto a la esquina superior izquierda del mapa como se aprecia en la siguiente figura.

Figura 114. Mapa con sus dimensiones.



- L= Longitud del campo (5m).
- A= Resolución ancho de la imagen (1280 pixeles).
- B= Resolución alto de la imagen (960 pixeles).
- X= Posición del robot a lo ancho.
- Y= Posición del robot a profundidad.

7.3.1.1 Prueba 1. Posición del robot medido por el flexómetro:

- Ancho (X): 257 cm.
- Profundidad (Y): 194 cm.

Figura 115. Imagen 1 tomada.

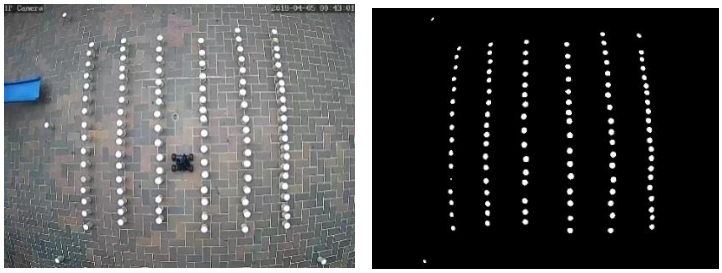


Figura 116. Ubicación de robot 1.

```
('x en pixeles ', 650)
('y en pixeles ', 599)
('x en centimetros', 255)
('y en centimetros', 189)
```

Tabla 15. Exactitud posición 1 del robot.

	Toma 1	Toma 2	Toma 3	Toma 4	Toma 5	Error
X (cm)	255	256	255	255	255	± 1 (cm)
Y (cm)	189	188	189	189	190	± 3 (cm)

7.3.1.2 Prueba 2. Posición del robot medido por el flexómetro:

- Ancho (X) = 185 cm.
- Profundo (Y)= 307 cm.

Figura 117. Imagen 2 tomada.

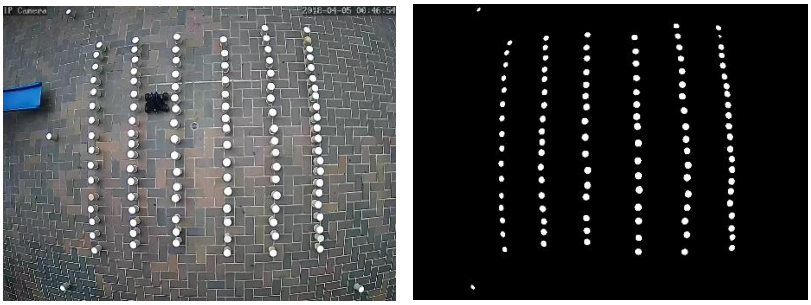
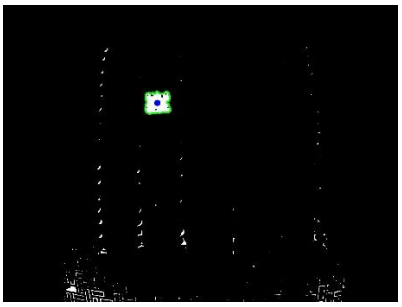


Figura 118. Ubicación de robot 2.



```
('x en pixeles ', 499)
('y en pixeles ', 354)
('x en centimetros', 176)
('y en centimetros', 316)
```

Tabla 16. Exactitud posición 2 del robot.

	Toma 1	Toma 2	Toma 3	Toma 4	Toma 5	Error
X (cm)	176	176	179	179	175	± 5 (cm)
Y (cm)	316	317	314	313	312	± 4 (cm)

7.3.1.3 Prueba 3. Posición del robot medido por el flexómetro:

- Ancho (X) = 40 cm.
- Profundo (Y)= 157 cm.

Figura 119. Imagen 3 tomada.

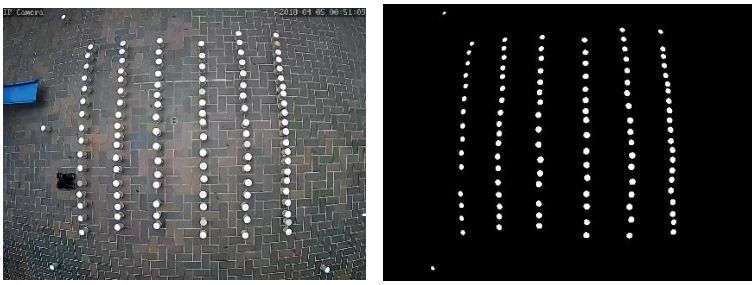
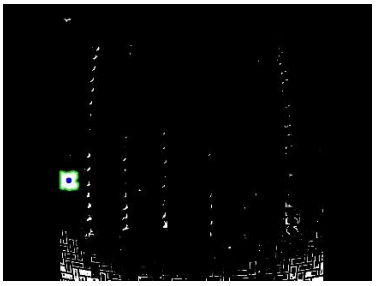


Figura 120. Ubicación de robot 3.



```
('x en pixeles ', 226)
('y en pixeles ', 649)
('x en centimetros', 34)
('y en centimetros', 162)
```

Tabla 17. Exactitud posición 3 del robot.

	Toma 1	Toma 2	Toma 3	Toma 4	Toma 5	%
X (cm)	34	34	35	35	35	± 3 (cm)
Y (cm)	162	165	160	161	163	± 3 (cm)

7.3.1.4 Prueba 4. Posición del robot medido por el flexómetro:

- Ancho (X) = 390 cm.
- Profundo (Y)= 430 cm.

Figura 121. Imagen 4 tomada.

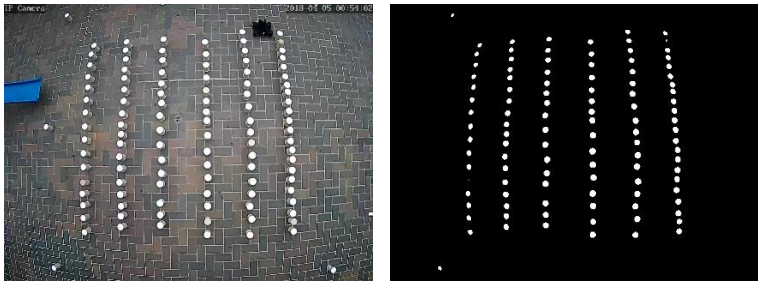
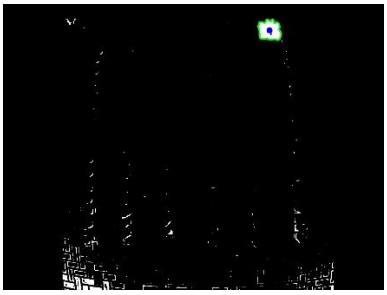


Figura 122. Ubicación de robot 4.



```
('x en pixeles ', 895)
('y en pixeles ', 125)
('x en centimetros', 382)
('y en centimetros', 435)
```

Tabla 18. Exactitud posición 4 del robot.

	Toma 1	Toma 2	Toma 3	Toma 4	Toma 5	Error
X (cm)	382	383	383	384	383	± 4 (cm)
Y (cm)	435	435	435	434	434	± 3 (cm)

7.3.1.5 Prueba 5. Posición del robot medido por el flexómetro:

- Ancho (X) = 30 cm.

- Profundo (Y)= 35 cm.

Figura 123. Imagen 5 tomada.

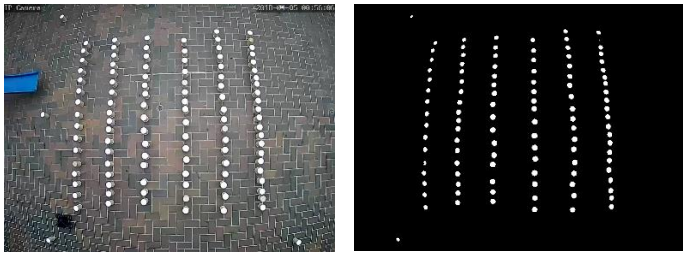


Figura 124. Ubicación de robot 5.

```
('x en pixeles ', 235)
('y en pixeles ', 880)
('x en centimetros', 39)
('y en centimetros', 42)
```

Tabla 19. Exactitud posición 5 del robot.

	Toma 1	Toma 2	Toma 3	Toma 4	Toma 5	% error
X (cm)	39	37	38	38	36	± 4 (cm)
Y (cm)	42	43	42	42	43	± 4 (cm)

7.4 PRUEBAS DEL SISTEMA DE NAVEGACION AUTONOMA

7.4.1 Generalidades y funcionamiento del código de control.

- Se procede a conectar todos los dispositivos (cámara, Raspberry Pi2 y PC) a una red Wi-Fi.
- Se procede a ubicar y a calibrar la cámara acorde a las condiciones y los parámetros establecidos anteriormente.

- Por medio del programa PuTTY[33] se procede a ingresar a la interfaz Raspbian de la Raspberry Pi2 desde un computador portátil, con el fin de activar el funcionamiento del robot móvil.

Figura 125. Programa PuTTY.

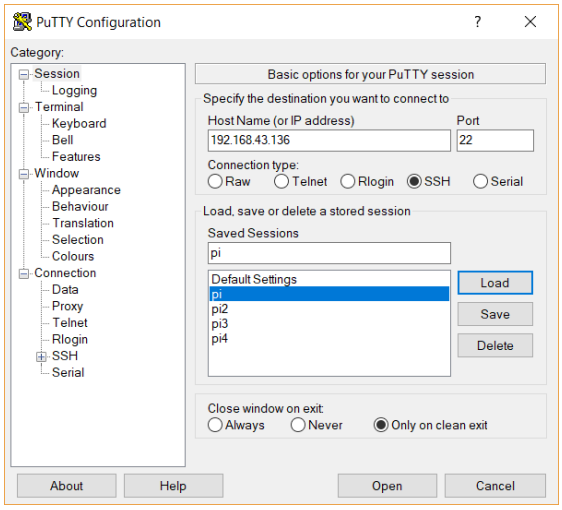
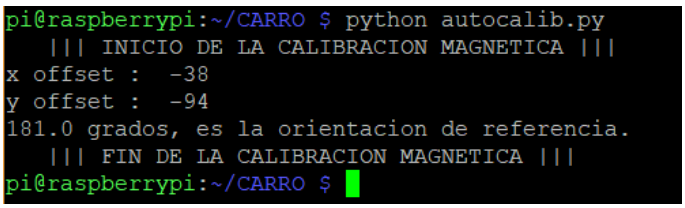


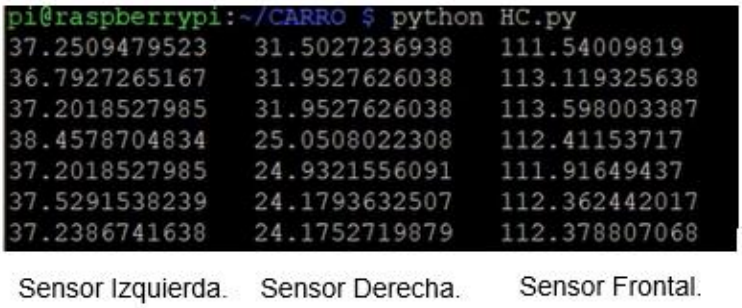
Figura 126. Calibración del robot.



- Se procede a calibrar el magnetómetro de la plataforma con el fin de obtener los valores de compensación del campo magnético terrestre en este entorno, necesario para la ejecución de giros.

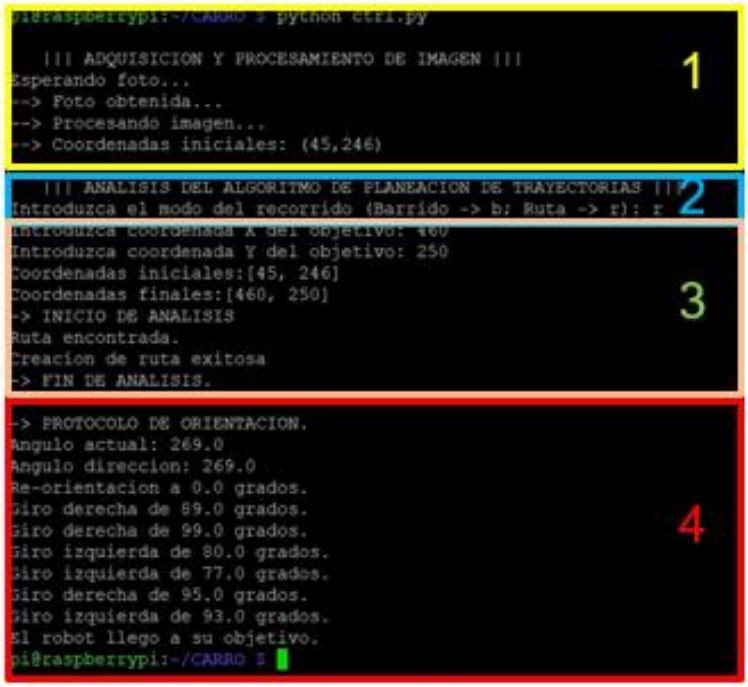
[33] PuTTY: Port unique terminal type, permite conectarse a servidores remotos iniciando una sesión en ellos que nos permite ejecutar comando.

Figura 127. Lecturas realizadas por los sensores de distancia.



- En una interfaz de PuTTY adicional se activa el código principal de los sensores de distancia, con el fin de realizar un proceso de medición de distancia en simultaneidad con la ejecución del movimiento.

Figura 128. Ejemplo de la demostración de los procesos del robot.



- Recuadro 1: Con el fin de iniciar el debido funcionamiento del robot, se procede a activar el código principal del sistema de localización en donde se espera el ingreso de la imagen proporcionada por la cámara. Dichas imágenes son

enviadas cada 15 segundos a la memoria de la Raspberry, de esta forma el programa siempre tomara la última imagen almacenada ignorado las anteriores con el fin de evitar el procesamiento de una imagen incorrecta.

- Recuadro 2: Una vez determinado el terreno y la ubicación del robot, se procede a preguntar al usuario que tipo de recorrido que desea ejecutar por medio del sistema de navegación.

- Recuadro 3: Si es seleccionado modo ruta, el programa pedirá al usuario la posición final (x, y) del robot en centímetros. Si es seleccionado modo barrido, se generará una trayectoria que recorrerá todas las hileras por medio del uso de nodos maestros.

- Recuadro 4: Ya determinada la ruta a seguir, el sistema de locomoción realiza un recuento de los avances y giros realizados hasta llegar al lugar destinado.

- Si durante el proceso de avance del robot, se detecta un obstáculo por el sensor frontal, el robot se detendrá y generará una alerta visual en el computador, posteriormente esperará 10 segundos para que el obstáculo sea removido. Si el obstáculo permanece una vez que pasen los 10 segundos, se asume que es de carácter estático, por lo que el robot pedirá de una imagen con la nueva información del terreno con el fin de poder generar una nueva ruta respecto a su posición actual hasta la posición final ya ingresada.

7.4.2 Pruebas en modo ruta Para las pruebas en el modo ruta de la plataforma robótica se propone una ruta entre las coordenadas [40,250] y [460,250]. Se realizará un análisis de los diferentes sistemas y el rol que cumplen para el desplazamiento autónomo del robot.

En la primera prueba, se realizará la ruta trivial entre las dos coordenadas para proponer un marco de referencia para las demás pruebas. En la segunda prueba se introducirán obstáculos, los cuales forzarán al algoritmo a crear una trayectoria no trivial. En la prueba final el robot creara la ruta trivial de la primera prueba, pero será detenido a mitad de la ejecución con un obstáculo dinámico, el cual forzara la creación de una nueva trayectoria a partir de la posición en la que se encuentra.

7.4.2.1 Sin obstáculos El tiempo de ejecución de la trayectoria fue de 35 segundos. El tiempo requerido para generar la trayectoria a partir de la imagen fue dé 42 segundos. El tiempo total para la ejecución del código fue de 94 segundos.

Figura 129 Trayectorias en modo ruta sin obstáculos.

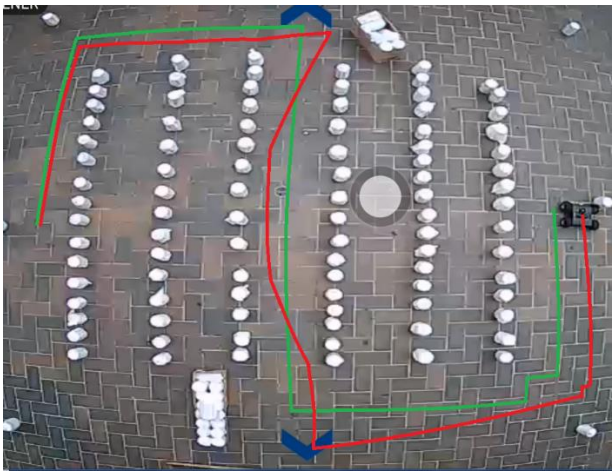
La línea verde indica la trayectoria generada por el algoritmo de navegación, la cual tiene una longitud de 8.4 metros. La línea roja muestra la trayectoria ejecutada por el robot a partir de la trayectoria generada por el algoritmo.

El recorrido es ejecutado apropiadamente hasta llegar a la esquina superior derecha en donde se ejecuta el giro de 90 grados con dirección hacia la parte inferior. Es entonces donde se puede observar un giro incorrecto de aproximadamente 110 grados lo que causa que el robot se dirija hacia la hilera, pero esta dirección es corregida con ayuda del sistema perceptivo a través de los sensores de distancia,

lo que evita que colisione. Esta corrección hace que el robot se aleje del objetivo por algunos centímetros.

7.4.2.2 Con obstáculos conocidos El tiempo de ejecución de la trayectoria fue de 52 segundos. El tiempo requerido para generar la trayectoria a partir de la imagen fue dé 45 segundos. El tiempo total para la ejecución del código fue de 109 segundos.

Figura 130 Trayectorias en modo ruta con obstáculo conocido.



La trayectoria generada por el algoritmo de navegación tiene una longitud de 12.8 metros. La línea roja muestra la trayectoria ejecutada por el robot a partir de la trayectoria generada por el algoritmo.

El sistema de localización logro definir los obstáculos con exactitud y esto se puede observar en la trayectoria generada por el algoritmo de navegación, sin embargo, se puede observar una irregularidad en el último tramo de la trayectoria, en donde se realiza un movimiento “escalón” el cual es causado por el cono en la esquina inferior.

En este recorrido se puede apreciar el error acumulativo de los encoders, los cuales hacen que el robot recorra una longitud mayor a la propuesta por el algoritmo. El magnetómetro realizo mediciones incorrectas las cuales se observan en el tercer giro. Los sensores de distancia corrigen la orientación del robot antes de que colisione. El robot llega al objetivo sin colisionar con una diferencia de 10 centímetros en el eje y a la trayectoria ideal.

7.4.2.3 Con obstáculos desconocidos El tiempo de ejecución de la primera trayectoria fue de 22 segundos y el de la segunda 28 segundos. El tiempo requerido para generar la primera trayectoria a partir de la imagen fue dé 38.5 segundos y el de la segunda trayectoria de 40 segundos. El tiempo total para la ejecución del código fue de aproximadamente 240 segundos.

El obstáculo dinámico fue posicionado en la parte superior de la quinta hilera.

Figura 131 Trayectoria con obstáculo desconocido.

La coordenada inicial para la segunda trayectoria fue [323,458].

La línea verde indica la trayectoria inicial generada por el algoritmo de navegación, la cual tiene una longitud de 4.9 metros. La línea azul indica la trayectoria generada tras encontrar el obstáculo y tiene una longitud de 7.7 metros. La línea roja muestra la trayectoria ejecutada por el robot a partir de las trayectorias generada por el algoritmo.

Los sensores de distancia lograron avisar a la plataforma con suficiente tiempo para que detuviera su movimiento sin colisionar, posteriormente determino que el obstáculo era de carácter estático por lo que llamo nuevamente el sistema de localización para determinar su posición y además actualizar los cambios en el entorno. Con el fin de evitar una lectura del estado del obstáculo errónea, el robot gira 180 grados.

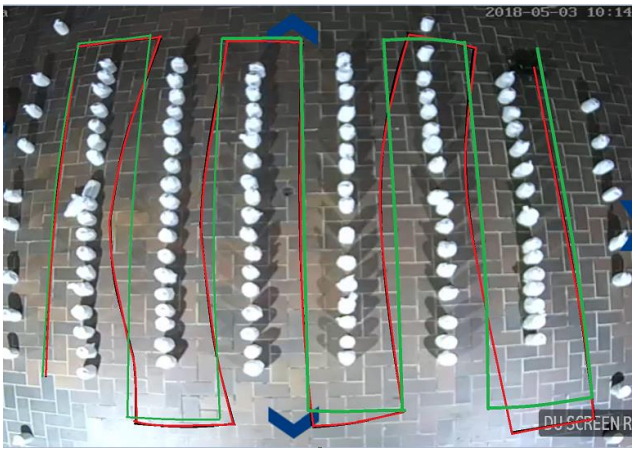
La coordenada suministrada por el sistema de localización en la segunda ejecución logro determinar con gran precisión su ubicación lo que facilito la creación de una trayectoria, sin embargo, el giro realizado para la orientación con la nueva trayectoria fue incompleto lo que ocasiono una dirección contra los obstáculos la cual fue corregida exitosamente por los sensores de distancia.

Debido a estas falencias en el sistema perceptivo, la plataforma llego a 15 centímetros de distancia del objetivo.

7.4.3 Pruebas en modo barrido Para las pruebas en el modo barrido de la plataforma robótica se propone una ruta a lo largo de todas las hileras de cultivo, desde la esquina inferior izquierda hasta la superior derecha. Se realizará un análisis de los diferentes sistemas y el rol que cumplen para el desplazamiento autónomo del robot.

7.4.3.1 Sin obstáculos El tiempo de ejecución de la trayectoria fue de 187 segundos. El tiempo requerido para generar la trayectoria a partir de la imagen fue dé 190 segundos. El tiempo total para la ejecución del código fue de 410 segundos.

Figura 132. Trayectoria en modo barrido sin obstáculos.



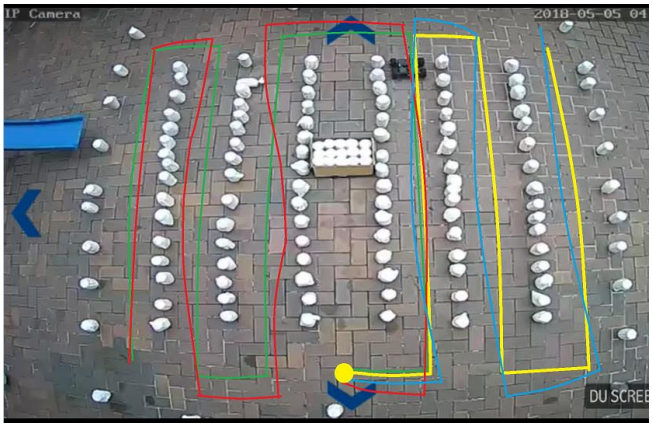
La línea verde indica la trayectoria generada por el algoritmo de navegación, la cual tiene una longitud de 33.6 metro. La línea roja muestra la trayectoria ejecutada por el robot a partir de la trayectoria generada por el algoritmo.

La ruta realizada por el robot presenta un alargamiento a medida que avanza por el campo, provocando una diferencia de 10 centímetros en el eje Y, esto se debe al error acumulativo provocado por los encoders al momento de tener el robot que retomar la ruta por medio de los surcos de maíz. Este error es compensado gracias a los giros bruscos de corrección que se pueden apreciar en las curvas 2, 4, 7, 10 y 11, los cuales retoman la ruta establecida por el algoritmo de navegación. Se puede observar en el primer surco la caída de un tubo, esto se debe al acercamiento del robot provocado por el giro de corrección, aunque gracias a los sensores de distancian evitaron la colisión del robot provocando la retoma de la ruta.

7.4.3.2 Con obstáculos conocidos El tiempo de ejecución de la trayectoria fue de 210 segundos. El tiempo requerido para generar la trayectoria a partir de la imagen

fue dé 215 segundos. El tiempo total para la ejecución del código fue de 390 segundos.

Figura 133. Trayectoria en modo barrido con obstáculo conocido.



La trayectoria generada por el algoritmo de navegación se dividió en dos colores para una mejor visualización, la línea verde representa la trayectoria hasta el nodo maestro donde se encuentra la hilera obstaculizada, y la línea amarilla representa el giro de 180 grados necesario para terminar de atravesar los demás nodos maestros. La trayectoria generada por el robot se representa de igual forma con dos colores, la línea roja muestra la trayectoria ejecutada por el robot desde su punto de rigen hasta el nodo maestro y la línea azul representa el trayecto del robot desde el nodo maestro hasta terminar la trayectoria generada por el algoritmo.

El sistema de localización logro definir correctamente el obstáculo ubicado en la cuarta hilera del mapa como se puede observar con la trayectoria realizada por el algoritmo de navegación. El error acumulativo producido por los encoders se sigue presentando como se puede apreciar en el alargamiento de las salidas de los surcos, pero al momento de realizar el giro de 180 grados, el robot lo realiza correctamente y retoma la ruta estipulada. A pesar de los errores el robot no colisiono con los surcos y termino su trayectoria con un desface de 15 centímetros en el eje Y.

8. CONCLUSIONES

- Por medio de este proyecto se desarrolló una herramienta que contribuirá con la investigación científica en el área de la robótica a través de la creación de una plataforma con dispositivos que le ayudaran en el desplazamiento autónomo tales como un sistema de localización de alta precisión, un sistema de percepción que lo orientara y le dará información de su entorno y unos algoritmos de planeación de trayectorias que facilitara el cumplimiento de su objetivo; estos sistemas le proporcionaran la suficiente autonomía para ejecutar una ruta satisfactoria en un mapa agrícola de maíz emulado con dimensiones establecidas.

- Se realizaron los modelos matemáticos para la planeación de trayectorias de los algoritmos de Dijkstra y Campos potenciales, mediante la recreación del entorno de trabajo. Los algoritmos fueron adecuados para ser empleados en un entorno agrícola emulado en el software Matlab con el fin de realizar un paralelo entre las rutas obtenidas en entornos estáticos con y sin obstáculos, de igual manera se evaluaron los tiempos de ejecución, porcentaje de uso de CPU y memoria utilizada en diferentes equipos.

- Se seleccionó el algoritmo de Campos Potenciales como modelo deliberativo principal para la creación de un sistema de navegación autónomo. A diferencia del algoritmo de Campos Potenciales, el algoritmo Dijkstra requirió una reducción de resolución, a pesar de esto, presento tiempos de ejecución hasta 30 veces mayores a los de Campos Potenciales, lo que lo hace poco viable para ser implementado en la plataforma robótica.

- Se diseñó y construyó un sistema de percepción que le facilita la corrección de la ruta, la detección de obstáculos y además apoya el sistema de localización relativo del robot. Para esto se utilizaron sensores de distancia HC-SR05 y el magnetómetro HMC-5883L que le permite a la plataforma robótica obtener información del entorno inmediato y además un marco de referencia global.

- Se diseñó y construyó un sistema de localización absoluto que determina la ubicación del robot con una resolución de centímetros. Esto se logró por medio del uso de una cámara WiFi que le transmite imágenes en tiempo real del entorno a la plataforma robótica, los cuales son procesados por la placa computadora del robot para determinar las condiciones del entorno y la coordenada del robot.

- El uso de la placa computadora Raspberry Pi 2 como sistema de control, facilito el uso del robot y el desarrollo de las aplicaciones necesarias para la ejecución del algoritmo de navegación. A través de esta se desarrollaron las funciones del sistema autónomo tales como, procesamiento de imagen, creación de trayectorias, corrección de ruta, protocolo de giro, y ejecución de movimiento. Además, el uso de un módulo WiFi facilito el control y la adquisición de datos en tiempo real para realizar los respectivos estudios.

- En la creación de las funciones de control se utilizó el lenguaje de programación Python en su versión 2.7 y se usaron las librerías: OpenCV para el procesamiento de imagen, Numpy y MatPlotLib para la creación y el manejo de matrices y RPi GPIO para el control de los pines en la tarjeta Raspberry Pi; las cuales resultaron de gran ayuda a la hora de escribir cada una de las funciones de la plataforma robótica.

- La implementación del algoritmo seleccionado en la plataforma robótica mostro tiempos satisfactorios con respecto a la planeación de trayectorias en entornos

con y sin obstáculos. Con tiempos de planeación de hasta 3 minutos en modo barrido, hasta tiempos de 40 segundos en modo ruta. Además, la ejecución de la trayectoria planeada resulto exitosa en la mayoría de los casos con distancias no mayores a 20 centímetros. Con la ayuda del sistema de percepción se obtuvo la información necesaria para la corrección autónoma y oportuna de la trayectoria y además se comprobó la habilidad del sistema para planear una trayectoria nueva tras ser detenida por un obstáculo desconocido inicialmente por el sistema de localización

- La implementación del sistema de localización por procesamiento de imagen, proporciono la información requerida sobre el entorno a trabajar, los obstáculos y la ubicación de la plataforma robótica con una desviación en la medida de ±5 centímetros en el eje X y de ±4 centímetros en el Y.

- Se ha podido valorar la utilidad de implementar en el algoritmo de navegación los nodos maestros. Gracias a esta división se ha tenido una mayor capacidad de detección y reducción de fallos, y además facilita enormemente la reutilización de estos nodos para retomar la ruta ya guardada ante obstáculos imprevistos al momento de planear una trayectoria nueva.

- El modelo de movimiento Skid Steering para robots móviles de tracción diferencial, tuvo un comportamiento adecuado al momento de realizar los giros en las esquinas suministradas por el sistema de navegación al momento de realizar los recorridos modo ruta. Aunque en los recorridos modo barrido presento inconvenientes, estos se deben a los errores acumulativos por la corrección de ruta dentro de las hileras.

9. RECOMENDACIONES

- El sistema de navegación realizado en este proyecto fue enfocado en un cultivo con un marco de plantación rectangular de medidas 60cmX20cm el cual puede ser adecuado a otras dimensiones. Se sugiere implementar los algoritmos de navegación aquí expuestos en otros marcos de plantación tales como tresbolillo.

- El sistema de percepción satisfizo los requerimientos para este proyecto, sin embargo, para la fase 2 de este proyecto se recomienda el uso de sensores más sensibles y con menos posibilidad de error ya que sensores como el encoder acumulan error con la distancia recorrida, lo que puede ser contraproducente para entornos de grandes dimensiones.

- El sistema de localización utilizado en este proyecto logro determinar la posición del robot con excelente precisión, pero la implementación de un sistema con cámara aérea para cultivos reales y de grandes dimensiones puede ser poco rentable o ineficiente, por lo que se recomienda el uso de sistemas de diferente resolución tal como un módulo GPS.

BIBLIOGRAFÍA

BETANCUR, Laura. Los drones son los nuevos ‘ojos’ de los científicos de la agricultura: Dron de ocho hélices capta imágenes de alta calidad de los cultivos del Valle del Cauca. El Tiempo [en línea] (2016) http://www.eltiempo.com/vida/ciencia/drones-usados-en-la-agricultura-39867> [citado en 14 de diciembre de 2016].

BUSTOS, Nery, GODÍNEZ, Julio. Navegación de un robot móvil aplicando campos potenciales y reconocimiento de objetos usando on Android. Ciudad de México, 2016. p.34.

CORDERO, Martín. Control de un robot móvil de ruedas mediante campos potenciales artificiales y procesamiento digital de imágenes en la evasión de obstáculos. México: Instituto Politécnico Nacional. Facultad de Ingeniería. 2011. p. 63

EL MERCURIO. Teledetección, una herramienta para hacer más eficiente las tareas agrícolas. [en línea]. <http://www.elmercurio.com/Campo/Noticias/Noticias/2014/04/23/Teledeteccion-una-herramienta-para-hacer-mas-eficiente-las-tareas-agricolas.aspx> [citado en 6 de marzo de 2015].

HAMAD, A. H., & IBRAHIM, F. B. Path Planning of Mobile Robot Based on Modification of Vector Field Histogram using Neuro-Fuzzy Algorithm. 2010. p 129-138.

HAWKINSON, J.. Guidelines for creation, selection and registration of an Autonomous System. [Base de datos en línea]. Marzo de 1996. Network Working Group. [citado el 10 enero 2018]. Disponible en https://tools.ietf.org/pdf/rfc1930.pdf.

HEMAV. Drones para agricultura-teledetección agrícola [en línea]. [citado el 22 de mayo de 2015].<https://hemav.com/drones-para-agricultura-teledeteccion-agricola/>

HONEYWELL. 3-Axis Digital Compass IC HMC5883L. [En línea] [Citado en 8 febrero de 2018] disponible en: < http://www.hobbytronics.co.uk/datasheets/HMC5883L-FDS.pdf >

INVENSENSE. ITG-3200 Product specification. [En línea] [Citado en 8 febrero de 2018] disponible en: < http://www.hobbytronics.co.uk/datasheets/PS-ITG-3200-00-01.4.pdf >

KONDO, N., M. MONTA, T. FUJIJURA, Y. SHIBANO Y K. MOHRI. Robot recolector de tomates. EN: Proceedings of the Asian Control Conference Conferenca 1-4. 1994. Japon.

LANNING, Daniel; HARRELL, Gregory; WANG, Jin. Dijkstra's algorithm and google maps. [Base de datos en línea]. Marzo 28 de 2014. Revista Communications of the ACM, Articulo No. 30. [citado el 10 enero 2018]. Disponible en https://dl.acm.org/citation.cfm?doid=2638404.2638494 .

LÓPEZ ORTEGA, Jorge. Reconstrucción de mapas utilizando escaneo láser para la navegación de un robot móvil. México, 2016, 144h. Trabajo de maestría (Tecnología en Cómputo). Instituto Politécnico Nacional. Centro de Innovación y Desarrollo Tecnológico en Cómputo.

LYNXMOTION. Products Rovers A4WD1 [en línea]. <http://www.lynxmotion.com/p-603-aluminum-4wd1-rover-kit.aspx>

MANDOW, Anthony & MARTÍNEZ, Jorge & MORALES, J & BLANCO, Jose Luis & GARCIA, Alfonso & González-Jiménez, Javier. 2007. Experimental kinematics for wheeled skid-steer mobile robots. 1222 - 1227. 10.1109/IROS.2007.4399139.

MASSEGURIDAD. Conexión P2P en grabadores y cámaras IP [en línea]. [citado el 1 de abril de 2016]. <https://www.masseguridad.es/es/articulos/cctv/692-conexion-p2p-en-grabadores-y-camaras-ip.html>

MATHWORKS. Robotic System Toolbox. [Citado el 15 septiembre de 2017]. Disponible en: https://la.mathworks.com/help/pdf_doc/robotics/robotics_ref.pdf

MATLAB. Que es Mat Lab [en linea]. (2015). [Consultado: 10 de nomviembre de 2017]. Disponible en Internet: https://juancarlosusomatlab2015.weebly.com/definicion-matlab.html

MISA, Thomas J.. An interview with Edsger W. Dijkstra. [Base de datos en línea]. Agosto de 2010. Revista Communications of the ACM, Vol. 53 (No.8), 41-47. [citado el 10 enero 2018]. Disponible en https://dl.acm.org/citation.cfm?doid=1787234.1787249 .

MORENO, José, RODRÍGUEZ, Francisco and GONZALEZ, Richard. Sistema de visión de apoyo a la navegación de un robot móvil en invernaderos. Almería, 2014, 8h. Universidad de Almería.

MOY, J.. OSPF protocol analysis. [Base de datos en línea]. Julio de 1991. Network Working Group. [citado el 10 enero 2018]. Disponible en https://tools.ietf.org/pdf/rfc1245.pdf.

POLOLU. Sharp GP2Y0A21YK0F Analog distance sensor. [En línea] [Citado en 13 enero de 2018] disponible en: <https://www.pololu.com/product/136>

PuTTY: Port unique terminal type, permite conectarse a servidores remotos iniciando una sesión en ellos que nos permite ejecutar comando.

REALE, Federico, PATIÑO, Charles and ARRIETA, Fernando. Estudio del estado del arte navegación en robots agrícolas. Montevideo, 2014, 55h. Trabajo de grado (Ingeniería en Computación). Universidad de la República. Facultad de Computación. Instituto de Computación.

RIBEIRO, Maria Isabel. Obstacle Avoidance. [En Linea] Instituto de Sistemas e Robótica, Instituto Superior Técnico. Lisboa, Portugal. Universidad de Lisboa. Noviembre de 2005. [citado el 10 de enero de 2018] Disponible en: http://users.isr.ist.utl.pt/~mir/

ROBERTS, Eric. Motion planning in robotics. [En línea]. courses. Standford, CA.: Universidad de Stanford. [citado el 26 de septiembre 2017] Disponible en https://cs.stanford.edu/people/eroberts/courses/soxco/projects/1998-99/robotics/basicmotion.html

RUIZ, Javier. Filtrado de imágenes y ecualización de histogramas. Santiago de Chile: 2015. p.4.

SIEGWART Roland, NOURBAKHSH ILLAH R, SCARAMUZZA Davide, Planning and navigating. En: Introduction to Autonomous Mobile Robots, Second edition, 2011. p.257

SIEGWART Roland, NOURBAKHSH ILLAH R, SCARAMUZZA Davide, Planning and navigating: Obstacle avoidance. En: Introduction to Autonomous Mobile Robots, Second edition, 2011. p.259

SKIENA, Steven. The Algorithm Design Manual. New York, USA: Springer, 2008. p.206.(Dijkstra's Algorithm). ISBN 978-1-84800-070-4

SPARKFUN. Ultrasonic ranging module HC-SR05. [En linea] [Citado en 13 enero de 2018] disponible en: < https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR05.pdf>

TESLABEM. Baterías LiPo - ¿Cómo usar y cuidar una batería LiPo? [en línea]. <http://blog.teslabem.com/como-usar-y-cuidar-las-baterias-lipo/> [citado en 3 de mayo de 2017].

VISION ROBOTICS CORPORATION. Robot recolector de naranjas (Producto). San Diego. 1999

ZHAO, Xin. System information class for windows. [Base de datos en línea]. Febrero de 2010. MathWorks. [citado el 15 septiembre de 2017)] Disponible en: https://la.mathworks.com/matlabcentral/fileexchange/26662-system-information-class-for-windows?s_tid=prof_contriblnk