SISTEMA DE NAVEGACIÓN AUTÓNOMA EN ROBOT MÓVIL TIPO ORUGA PARA APOYO EN TAREAS DE SIEMBRA EN CAMPOS CAFICULTORES

LUIS HERMES ORTEGA QUIÑONEZ
NICOLAS SILVA GARCIA

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
FACULTAD DE INGENIERIAS
ING. MECATRÓNICA
BUCARAMANGA
2021

SISTEMA DE NAVEGACIÓN AUTÓNOMA EN ROBOT MÓVIL TIPO ORUGA PARA APOYO EN TAREAS DE SIEMBRA EN CAMPOS CAFICULTORES

DESARROLLADO POR:
LUIS HERMES ORTEGA QUIÑONEZ
NICOLAS SILVA GARCIA

Trabajo de grado propuesto para sintetizar los procesos de investigación, clase, conceptos y hallazgos efectuados durante los últimos años para acceder al título de Ingeniería Mecatrónica.

DIRECTOR:
OSCAR EDUARDO RUEDA SÁNCHEZ, M. Sc
CO DIRECTOR:
SEBASTIAN ROA PRADA, Ph. D

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
FACULTAD DE INGENIERIAS
ING. MECATRÓNICA
BUCARAMANGA
2021

Nota de aceptación:

Proyecto de grado titulado “SISTEMA DE NAVEGACIÓN AUTÓNOMA EN ROBOT MÓVIL TIPO ORUGA PARA APOYO EN TAREAS DE SIEMBRA EN CAMPOS CAFICULTORES”, presentado por los estudiantes Luis Hermes Ortega Quiñonez y Nicolas Silva García para optar por el título de ingeniero mecatrónico.

OBSERVACIONES:

_____

_____

_____

_____

_____

Oscar Eduardo Rueda Sánchez M.Sc.

FIRMA DEL DIRECTOR

Hernando González Acevedo M.Sc.

FIRMA DEL EVALUADOR 1

FIRMA DEL EVALUADOR 2

Miguel Arlenzo Duran Sarmiento M.Sc.

DEDICATORIA

Al llegar a este punto de la carrera da un poco de nostalgia, mirar atrás, recordar todo los vivido en los últimos años es reconfortante, horas de estudio, alegrías, tristezas, sacrificios, etc. Decir que llegue hasta acá solo sería mentir, por tal razón le quiero hacer una mención especial a mi familia, mi padre Luis Hermes, mi madre Cielo Janeth, mi hermana Valentina y cada miembro que ha aportado su grano de arena a esta carrera.

Luis Hermes Ortega Quiñonez

Estando en este punto tan determinante de mi carrera profesional, y haciendo memoria de tantos momentos alegres, vivencias, experiencias y conocimiento en todos los ámbitos personales sin olvidar la innumerable cantidad de esfuerzos, sacrificios, noches enteras de estudio e interminables jornadas de trabajos y trasnochos acompañadas no siempre de los mejores resultados y con notas a veces tan desalentadoras. No hay mejor manera y espacio para traer a mención todas esas personas que han aportado de alguna u otra manera el estar hoy alcanzando este logro personal tan importante en mi vida, dando un elogio especial a mi padre Enrique Silva, mi madre Sandra García, mis hermanos Sebastián y Mariana, y Anny que no han desistido conmigo ni dejado de apoyarme en cada etapa de este trayecto.

Nicolás Silva García

AGRADECIMIENTOS

En un proceso tan largo uno se encuentra con una infinidad de personas en el proceso, por tal razón, es fundamental recordar y agradecer. Primero a Dios por permitirme vivir este proceso con buena salud, mis padres que me acompañaron y apoyaron con todos los inconvenientes presentados, pasando por los docentes que tuvieron paciencia y se esforzaron por brindarme los conceptos técnicos de la mejor manera; así también el personal de la universidad (Mercadeo, Becas, Créditos, Facultad) que siempre trabajaron para brindarme las mejores soluciones. Gracias a todos.

Luis Hermes Ortega Quiñonez

En este importante y largo trayecto es importante dar el reconocimiento principal a Dios primero y antes que nada por la vida, por la salud, por la familia que ha puesto a mi lado para apoyarme a cumplir cada uno de mis sueños y en segunda instancia a cada una de esas personas que se me ha presentado en este proceso y de las que he aprendido a apreciar el esfuerzo ajeno, el tiempo de las personas, la lealtad y compañerismo de nuestros colegas de carrera y el importante papel que desempeñan dentro en nuestra carrera formativa el gran plantel docente y educativo en todas las áreas. Gracias a cada uno de ellos estoy cumpliendo con esta etapa primordial para mi vida.

Nicolás Silva García

CONTENIDO

INDICE DE TABLAS

LISTA DE FIGURAS

1. INTRODUCCIÓN

La agricultura de precisión es un área que está evolucionando los cultivos del país, su objetivo es medir, controlar y ejecutar procesos con la menor intervención humana posible, con el fin de mejorar los procesos productivos en la plantación, cosecha y post cosecha. Esto fue un factor fundamental para la selección de tema en este trabajo de grado. Para las condiciones locales se encontró en los campos caficultores una serie de problemáticas que se pueden solucionar bajo la concepción de la agricultura de precisión.

Por tal razón este trabajo de grado apunta a una solución con el objetivo de diseñar sistema de navegación autónomo a robot móvil tipo oruga instrumentado, capaz de reconocer su ubicación, planear trayectorias y transmitir información a interfaz de mando, para apoyar tareas de siembra en campos caficultores. Este sistema le brindara al robot un nivel de autonomía que ahorrara tiempos y esfuerzos al proceso del café.

En el desarrollo del proyecto fueron usadas múltiples metodologías para encontrar una solución a las necesidades del usuario y a los criterios de diseño fundamentados en las generalidades del cultivo de café. Durante el desarrollo del presente trabajo, se utilizaron las metodologías APTE, FAST y QFD para encontrar los parámetros preliminares y así obtener diferentes opciones conceptuales. Una vez obtenidos los criterios conceptuales en la etapa de diseño, se hizo uso de la metodología Design Thinking con la cual se obtuvo el resultado definitivo.

2. PROBLEMÁTICA

Para iniciar cualquier proyecto de investigación es fundamental entender el entorno en el que vivimos para de esta manera encontrar los problemas o debilidades en alguna área o parte de cualquier proceso. Para este proyecto de grado se consultó en la red y directamente en el cafetal El Roble para concluir que:

2.1 PLANTEAMIENTO

El sector cafetero es una de las actividades económicas más destacadas y representativas del país, por lo que constituye un área de gran influencia socioeconómica para el campo colombiano. Actualmente, este sector se está viendo afectado por la disminución de la oferta de mano de obra para el cuidado, cosecha y pos cosecha de los campos cafeteros; Al mismo tiempo, se está presentando un incremento considerable en el área cultivada y en la producción de café, lo cual trae consigo un incremento de la demanda de mano de obra.

La agricultura de precisión es un área de la robótica móvil en auge debido a las soluciones que pueden brindar hacia un sector fundamental en la economía Nacional. Esto fue un factor que predominó en la selección del tema a desarrollar en este proyecto de grado, a la vez que se encontró en la caficultura colombiana un sector con una serie de problemas como lo son la escasez de mano de obra, una suma innumerable de pérdidas en la puesta a punto en el proceso de preparación para recolecta o producción. De igual manera en el tiempo que se está llevando a cabo estas tareas y una vez finalizadas debido al no aprovechamiento a tope de la mano de obra involucrada en el campo ya sea por condiciones climáticas, error humano o diferentes factores ergonómicos que alteran la productividad máxima del empleado. Estos son puntos claves a considerar para el desarrollo de este proyecto.

2.2 JUSTIFICACIÓN

En este proyecto se desarrollará, un sistema inteligente de robot móvil tipo oruga de carga, con capacidad de ubicación, trayectorias y alarmas por medio de la telemetría para que sea un sistema totalmente autónomo con la capacidad de desarrollar diversas tareas entre el humano recolector y el centro de recolecta. Además, este proyecto “Sistema de navegación autónoma en robot móvil tipo oruga para transporte de suplementos y apoyo en tareas de siembra en campos caficultores” cuenta con la asesoría de los docentes Sebastián Roa y Oscar Rueda y también apoyará la investigación y desarrollo al proyecto “Vehículo terrestre con navegación autónoma para apoyo en tareas de cultivo de café” participante en la X convocatoria bienal de la Universidad Autónoma de Bucaramanga.

El vehículo desarrollado espera ser la solución de algunos asuntos problemáticos y como lo son la exposición de la mano de obra al constante transporte de carga, los fuertes pesticidas y las largas horas de trabajo en la cosecha. Con lo anterior, se alcanzará un rendimiento eficaz y pertinente dentro de estas actividades. A causa de las condiciones necesarias para el cultivo del café colombiano, este se encuentra generalmente en pendientes, lo que implica que la maquinaria grande y pesada tendría dificultades para realizar sus funciones. Por lo tanto, se hace necesario la concepción de equipos que se adecúen a estos terrenos escarpados y húmedos.

3 OBJETIVOS

3.1 OBJETIVO GENERAL

Diseñar sistema de navegación autónoma a robot móvil tipo oruga instrumentado, capaz de reconocer su ubicación, planear trayectorias y transmitir información a interfaz de mando, para apoyar tareas de siembra en campos caficultores.

3.2 OBJETIVOS ESPECÍFICOS

- Determinar el modelo cinemático que rige el robot móvil, para caracterizar la información necesaria en la navegación autónoma.
- Desarrollar sistema de localización del robot móvil, para conocer su ubicación dentro de un cultivo de café en tiempo real.
- Desarrollar un algoritmo en base a la planeación de trayectorias para el campo de café establecido.
- Diseñar interfaz gráfica Hombre-Máquina para la operación continua y manipulación manual del robot móvil
- Realizar puesta a punto de la estructura física, instrumentación, mecanismos de transmisión, movimiento y tracción del robot móvil tipo oruga.
- Validar sistema de localización instantánea, algoritmo de trayectoria, planta física, interfaz gráfica humano-robot, lógica aplicada y funcionamiento propio del robot móvil.

4 METODOLOGÍA

Al considerar las características de la metodología en mecatrónica, se dispone:

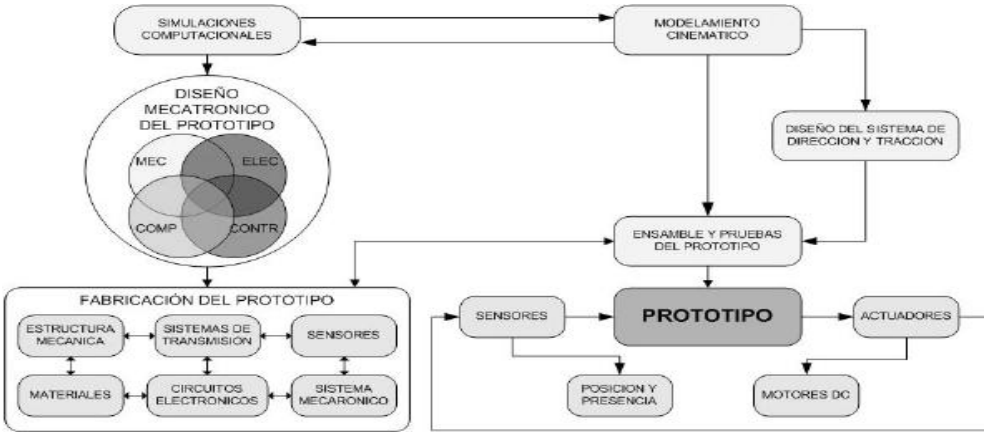


Figura 1: Metodología de diseño mecatrónica para la construcción de prototipos.

Los requerimientos solicitados a nivel general para un proyecto de grado, se basa en:

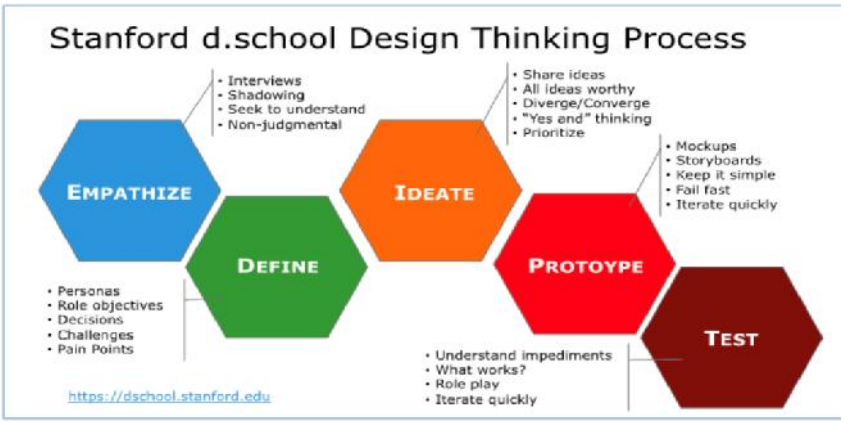


Figura 2: Metodología de Design Thinking

Se empleará la anterior metodología en la que se llevará constante avance debido a un cronograma de actividades previamente estipulado para cumplir uno a uno los objetivos específicos del proyecto.

1. Determinar el modelo cinemático que rige el robot móvil, caracterizar la información necesaria para la navegación autónoma.

1.1 Inferir las tareas específicas a desarrollar por el robot móvil para conocer dentro de que limites va a estar en movimiento para determinar los sensores y actuadores necesarios (EMPATHIZE).
1.2 Leer los textos de trabajos de grado y estados del arte referente a las tareas por realizar y poder establecer el óptimo modelo para nuestro robot móvil(DEFINE).

2. Desarrollar sistema de localización del robot móvil para conocer su ubicación dentro de un cultivo de café en tiempo real.
2.1 Visitar campo caficultor, tomar los datos necesarios para construir mapa de referencia al robot móvil(EMPATHIZE).
2.2 Clasificar las diferentes técnicas y componentes disponibles en localización, para cualquier dispositivo no tripulado(EMPATHIZE).
2.3 Determinar mediante balance de calidad y conveniencia el mejor sistema de ubicación para el robot móvil en cuestión(DEFINE).
2.4 Desarrollar sistema para transmitir información del robot móvil al puesto de mando para asegurar los avances del dispositivo(IDEATE).

3. Desarrollar un algoritmo en base a la planeación de trayectorias para el campo de café establecido.
3.1 Almacenar la información recolectada del mapa caficultor en un sistema de procesamiento(DEFINE).
3.2 Identificar los objetos que le rodean y las trayectorias disponibles dentro del campo caficultor establecido(IDEATE).

4. Diseñar interfaz gráfica Hombre-Máquina para la operación continua y manipulación manual del robot móvil.
4.1 Seleccionar los datos y características más importantes para su constante evaluación y toma de decisiones(DEFINE).
4.2 Desarrollar la interfaz Hombre-Máquina con la cual manipulará y evaluará el robot móvil(IDEATE).

4.3 Validar él envió (circuito) y recepción (interfaz) de la información para comprobar el sistema de comunicación a implementar(TEST).

5. Realizar puesta a punto de la estructura física, instrumentación, mecanismos de transmisión, movimiento y tracción del robot móvil tipo oruga.
5.1 Validar experimentalmente el robot móvil tipo oruga (Coffee Rover) haciendo un reconocimiento de la funcionalidad del prototipo ya diseñado y evaluando las partes a mejorar o rediseñar(IDEATE).
5.2 Adaptar los nuevos dispositivos al diseño del robot móvil para validar su correcta implementación y para su nueva puesta en marcha(PROTOTYPE).

6. Validar sistema de localización instantánea, algoritmo de trayectoria, planta física, interfaz gráfica humano-robot, lógica aplicada y funcionamiento propio del robot móvil.
6.1 Evaluar el recorrido del robot móvil de esta manera comprobar que el robot móvil está en óptimas condiciones en este campo de trabajo(TEST).
6.2 Evaluar el funcionamiento en general, cumpliendo con lo esperado y así validar el algoritmo, sistema de localización, transmisión de datos y comprobar las óptimas condiciones en este terreno de trabajo(TEST).
6.3 Realizar cambios y modificaciones en robot móvil de acuerdo con los resultados obtenidos, para una veraz retroalimentación del proyecto en su totalidad(TEST)

Tabla 1: Cronograma de actividades

OBJETIVOS	ACTIVIDADES	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	METOGOLOGÍA
Modelo cinemático que rige el robot móvil.	Inferir las tareas específicas desarrolladas por el robot																	EMPATHIZE
	Establecer modelo																	DEFINE
Sistema de localización del robot móvil	Visitar Campo Caficultor																	EMPATHIZE
	Clasificar tecnicas y componentes disponibles																	EMPATHIZE
	Determinar mejor sistema de ubicación																	DEFINE
	Sistema de transmisión de datos para el robot																	IDEATE
Algoritmo en base a la planeación de trayectorias en el robot móvil	Almacenar el mapa caficultor en sist. de procesamiento																	DEFINE
	Identificar objetos y trayectorias disponibles en el terreno																	IDEATE
	Pruebas de campo para los recorridos establecidos																	TEST

OBJETIVOS	ACTIVIDADES	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
Interfaz grafica Hombre-Maquina	Seleccionar datos y características más importantes																	DEFINE
	Diseñar interfaz de mando																	IDEATE
	Validar la transmición de la información																	TEST
Realizar puesta a punto de la estructura física del robot móvil	Evaluar experimentalmente su funcionamiento mecánico																	IDEATE
	Adaptar los nuevos dispositivos al diseño físico																	PROTOTYPE
Validación del sistema y su funcionamiento en general	Validar el recorrico (trayectoria) del robot móvil																	TEST
	Evaluar el funcionamiento de todo lo implementado																	TEST
	Realizar cambios y modificaciones pertinentes																	TEST

5 ESTADO DEL ARTE

1) REFERENCIA BLIOGRAFICA: http://www.interempresas.net/Horticola/Articulos/151745-El-uso-de-robots-en-tareas-agricolas.html

Los robots son comúnmente utilizados en la manufactura de productos de alimentación, pero aún son poco frecuentes en el campo abierto o en los invernaderos.

Por tal razón se diseñó e implementó un sistema de navegación que se trata medir o actuar sobre localizaciones precisas, ya que la primera necesidad a ser cubierta por un robot que se mueve en un entorno agrícola es la de navegación. La tecnología a utilizar en el caso de agricultura de interiores o exteriores es significativamente diferente. Así, en el caso de robots para interiores, puede contemplarse desde el uso de robots que se desplazan por rieles, facilitando enormemente su localización; hasta la inclusión de marcas o balizas en el entorno

(ópticas, RFID, etc.), que pueden simplificar la tarea de localización del robot. Por otro lado, en los robots de interior no se considera el uso de los sistemas de navegación por satélite, como el conocido GPS, o el futuro sistema Galileo, sistemas que forman parte destacable, pero normalmente insuficiente, del sistema de navegación de los robots agrícolas de exteriores.

En ambos casos la navegación del robot suele verse apoyada por sistemas de medida inercial (IMU) y por sistemas de telemetría que, con los correspondientes algoritmos, consiguen precisiones suficientes para el posicionamiento del robot. Toda esta tecnología está implantada en otros sectores y en fuerte proceso de mejora y crecimiento.

Sistemas de Visión por Computador**:** La visión por computador 2D y 3D, tanto en espectro visible como en otros espectros, constituye un pilar fundamental en el uso de robots en agricultura, tanto para las tareas de captura de información que, adecuadamente procesada, permite la toma de decisiones, como para ser usado como instrumento de ayuda al posicionamiento relativo entre el robot y el entorno sobre el que debe actuar. Las oclusiones entre los objetos sobre los que actuar, como por ejemplo frutos dentro de la masa foliar, la falta de homogeneidad y por ello difícil clasificación de los objetos presentes en la imagen y la variabilidad de las condiciones de iluminación, suponen una notable dificultad que limita aún las posibilidades de los robots tanto en exteriores como en interiores.

Plataformas**:** Mientras que en el caso de robots de interiores las plataformas que transportan y posicionan los sistemas de medida o actuación son similares a las utilizadas en otras aplicaciones (transporte en almacén, vigilancia etc.) en el caso de robots de exteriores, las características del terreno suponen un factor a considerar. Muchos de los sistemas robóticos para ser aplicados en agricultura de exteriores se han basado en la ‘robotización’ de determinada maquinaria agrícola, incorporando los sensores necesarios para su navegación autónoma (los ya citados

GPS, IMU, telemetría, etc.). Sin embargo, la tendencia actual apunta al desarrollo de ‘pequeños’ robots agrícolas, más flexibles en cuanto a su cometido, con mejor capacidad de maniobra, robustez colectiva ante averías (el conjunto puede seguir operando aún si uno de ellos se detiene) y con un menor impacto en la compactación de terreno. Así mismo el acceso a cultivos de ladera o diferentes a praderas, es abordable con pequeños robots mejor que con máquina de medio o gran tamaño.

Sistemas de Manipulación: La manipulación, generalmente asociada al cosechado, tanto en invernadero, como en frutales, supone uno de los retos más complejos por resolver, pues hay que combinar un agarre estable sobre objetos de geometría variable, con una adecuada velocidad que haga eficiente la automatización del proceso.

2)REFERENCIA BIBLIOGRAFICA: DISEÑO Y CONSTRUCCIÓN DE UN ROBOT MÓVIL COMO PLATAFORMA PARA EL APOYO A LAS LABORES EN LOS CULTIVOS DE CAFÉ [Trabajo de grado].
En la búsqueda del estado del arte se evidencio las siguientes tareas puntuales en la cuales se ha venido trabajando en agricultura de precisión:

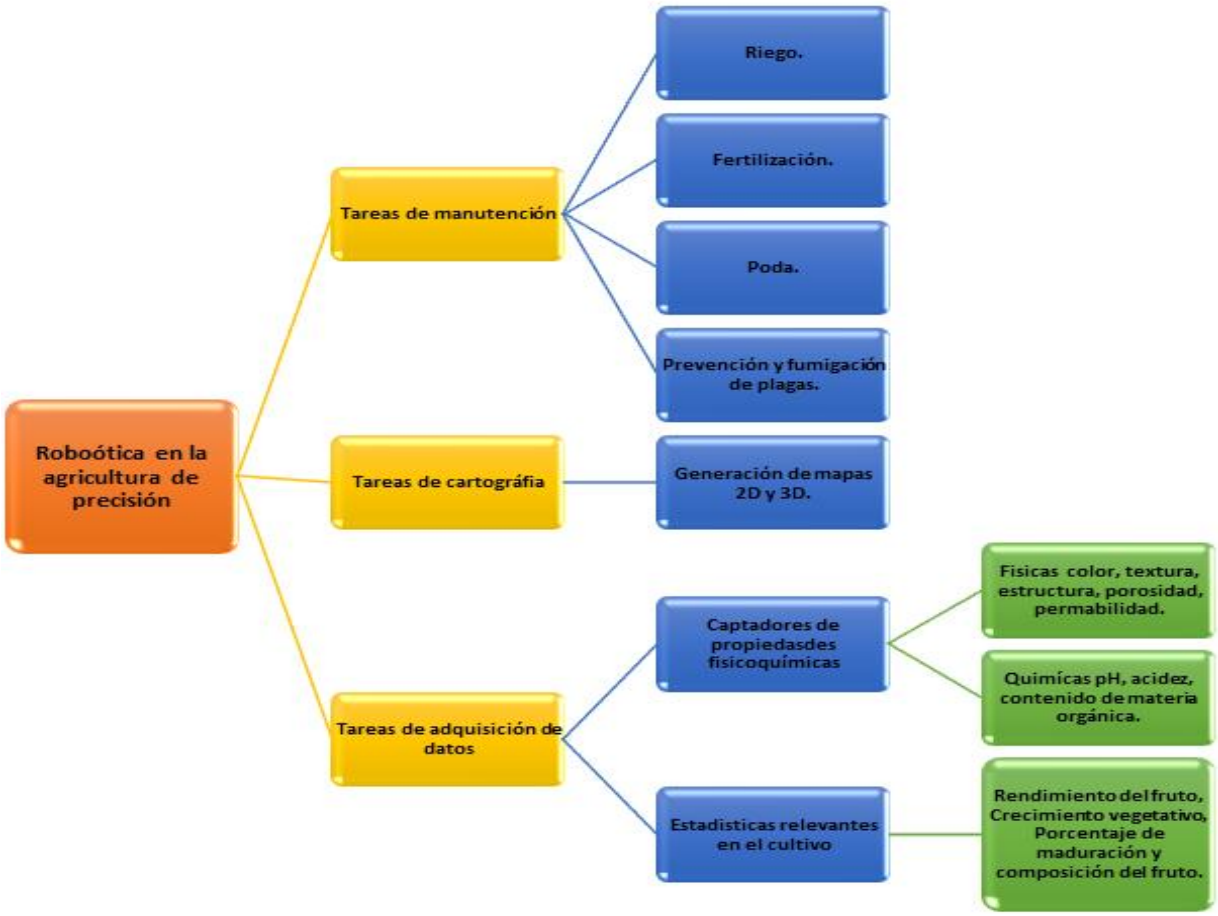


Figura 3: Tareas desarrolladas en agricultura de precisión

A esta clasificación se llegó al determinar los patrones en común de los robots ya desarrollados por distintas entidades y para diferentes niveles de implementación en el área de la agricultura de precisión.

Entre los cuales se hace referencia a los más relevantes:

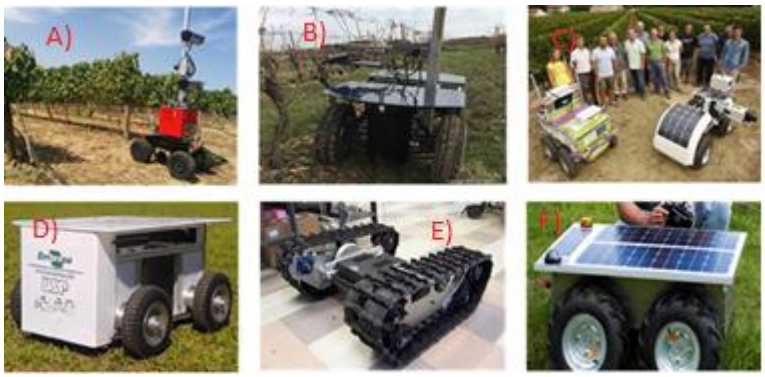


Figura 4: a) Vinbot, b) Myce, c) VineRobot I y II, d) Mirã Rover, e) Segway tracked robot, f) Agri-rover.

Un ejemplo inicial de estos es el Segway Tracked Robot, patrocinado por ‘‘Conjunto SPAWAR & UCSD’’ en el invierno del 2012. Dicho equipo, se centró en plantear la cuestión de cómo es posible hacer vehículo robótico orugas relativamente barato con un único marco de los controles. El Segway PGR 50 cuesta $ 7.210 dólares y se puede dar fácilmente comandos desde una interfaz de bus serie CAN. Idealmente, este tipo de robot podría ser utilizado como un vehículo para una variedad de diferentes aplicaciones posibles. El segway tracked robot, presenta una serie de objetivos de alta prioridad y encontramos (movilidad, robustez, versatilidad y carga). La movilidad consiste en recorrer una multitud de terrenos, y subir pendientes del 100%. La robustez plantea que sea fiable en ambientes y condiciones climáticas hostiles.

Un par de años después, en el 2014 se realizó un proyecto que contó con 8 socios (5 PYMES y 3 organizaciones de IDT) de 4 países europeos (Francia, Alemania, Italia y España). El proyecto VineRobot incluuyó el diseño, desarrollo e implementación de un robot de uso agrícola, como un vehículo terrestre no tripulado (UGV), equipado con inteligencia artificial y tecnologías de detección no invasivas. Este robot puede realizar mediciones de parámetros como: el rendimiento de la uva, el crecimiento vegetativo, el estado de agua y la composición del fruto. Otra función que cumple este rover es procesar y enviar a los vinicultores los datos recolectados; Los usuarios finales reciben los datos en tiempo real en aplicaciones desarrolladas

para tabletas, computadoras o cualquier otro dispositivo inteligente, para buscar la mejora de la gestión vitivinícola y la calidad de la uva.

En Barcelona, en ese mismo año, 2014, se lanzó Vinbot, es un robot móvil autónomo todo terreno (Summit XL) en consorcio con Robotnik para optimizar la gestión del rendimiento y la calidad del vino. VinBot está dotado con un conjunto de sensores capaces de capturar y analizar imágenes de viñedos y datos en 3D mediante el uso de aplicaciones de cloud computing. Su finalidad es determinar el rendimiento de los viñedos y compartir esta información con los viticultores. Vinbot evaluará con precisión el rendimiento de las uvas y los fito-datos relevantes a través de un conjunto de sensores, rastreando el estado y la ubicación de los activos, generando mapas, capturando puntos de muestreo, y compartiendo esa información de una manera rápida, flexible, autónoma y fácil de usar.
En el año 2015 en Brasil, se creó Mirã Rover. fue desarrollado como un proyecto de cooperación entre LabRom (EESC - USP) y Embrapa Instrumentação. Utilizó una tecnología incorporada para analizar elementos esenciales de los nutrientes del suelo para aplicaciones de agricultura de precisión.

3) REFERENCIA BIBLIOGRAFICA: Sistema de navegación, ubicación y modelado del entorno para un robot móvil. Jose Arcos. http://tierra.aslab.upm.es/documents/PFC/PFC_JAArcos.pdf
El origen de las dificultades en la localización y la construcción de mapas deriva de la existencia de ruido en las medidas de los sensores y en las limitaciones en el rango de las mismas. Un poco más en profundidad, los principales factores que impiden que el proceso sea más sencillo son los siguientes:
Las observaciones se obtienen con respecto al sistema de referencia propio del robot, cuya posición viene afectada de un cierto grado de incertidumbre inherente a la odometría. Así, la imprecisión en las observaciones se añade a la ya existente en la posición del robot y se complica extremadamente la minimización de los errores.

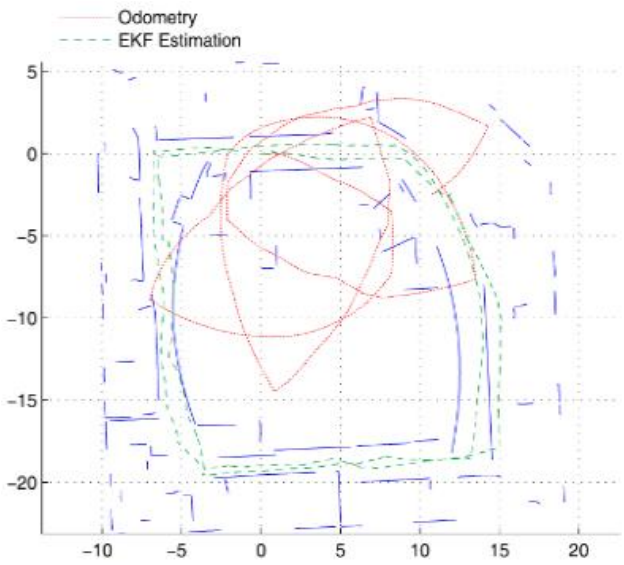


Figura 5: Trayectorias del robot por sensores.

En rojo se muestra la trayectoria del robot según la detectan los sensores odométricos. En verde, la trayectoria real del robot.

El SLAM (simultaneous localization and mapping) es una técnica en la cual un robot o vehículo autónomo opera en un entorno a priori desconocido, utilizando únicamente sus sensores de abordo, mientras construye un mapa de su entorno, el cual utiliza al mismo tiempo para localizarse. Los sensores tienen un gran impacto en los algoritmos usados en SLAM. Enfoques recientes se están centrando en el uso de cámaras como sensor principal, ya que generan mucha información y están bien adaptadas para su aplicación en sistemas embebidos: son ligeras, baratas y ahorran energía.

A continuación, se explicarán los tres algoritmos básicos existentes en la actualidad para resolver el problema de SLAM, de los que deriva la gran mayoría de algoritmos aplicados a día de hoy.
El primero, conocido como Filtro Extendido de Kalman (de siglas EKF eninglés) es el primero históricamente, pero su uso ha disminuido debido a sus propiedades computacionales limitantes. El algoritmo utilizado en este proyecto deriva del EKF-

SLAM, corrigiendo algunos de sus problemas.
El segundo, basado en representaciones gráficas, aplica satisfactoriamente métodos de optimización no lineal (sparse nonlinear optimization methods) para resolver el problema de SLAM, y se ha convertido en el algoritmo más usado para la resolución del problema de SLAM completo.
El tercer y último método usa técnicas de filtrado estadístico no paramétrico, conocidas como filtros de partículas. Es un método muy usado para hacer SLAM online y aporta una nueva solución al problema de la asociación de datos en SLAM.

4) REFERENCIA BIBLIOGRAFICA: ALGORITMOS DE PLANIFICACION DE ROBOTS USANDO FAST MARCHING. Universidad Carlos III de Madrid. https://core.ac.uk/download/pdf/30044327.pdf

Se considera una ruta óptima aquella que realiza un trazado suave y que evita los obstáculos desconocidos (dinámicos o no). Según Latombe (1991), los métodos de planificación de rutas para robots móviles se dividen en cinco tipos:

Roadmap: Este método consiste en representar en forma de red el entorno y luego aplicar algoritmos de búsqueda gráfica para encontrar la ruta.

Exact cell Decomposition: Consiste en dividir elentorno en celdas no superpuestas que son llevadas a un gráfico.

Approximate Cell Decomposition: A diferencia del método anterior las celdas tienen un tamaño predefinido y no cubren el espacio libre en su totalidad.

Potential Fields (Campos de potencial): Este método fue desarrollado por Khatib (1986). Se basa en la creación de un campo de potencial artificial en el cual la meta o destino es un polo de atracción y los obstáculos polos de repulsión. El robot sigue el gradiente de este potencial hacia su mínimo. La ventaja de este método es la simplicidad y la facilidad de tratamiento de obstáculos tanto fijos como móviles. La desventaja es la existencia de mínimos locales y de oscilaciones bajo ciertas configuraciones de obstáculos (zonas estrechas).

Navigation Functions: Son consideradas un caso especial del método Potential Fields.

Estos métodos se basan en repetir el proceso de percepción-acción en una tasa muy elevada (frecuencia). El proceso consta de dos pasos: primero se lee la información de los sensores y seguidamente se genera una orden de movimiento para evitar la colisión mientras el robot continúa su camino hacia su destino sin alterar la ruta global planificada.

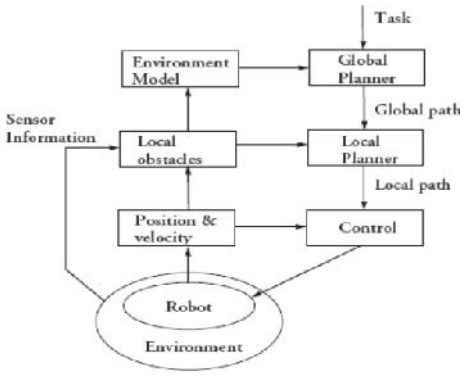


Figura 6: Método para planeación de trayectorias

6 MARCO TEÓRICO

En este capítulo se registra toda la información encontrada en línea, librerías y revistas científicas con el fin de dar solución a cada uno de los objetivos planteados.

6.1 GENERALIDADES DEL CULTIVO DE CAFÉ

Para entender mejor el proceso se analizó en primera parte todo lo relacionado con el café. Cenicafé ha explorado diferentes alternativas, sin embargo, su implementación e impacto han sido limitados, debido a:

- Características propias e idiosincráticas de la caficultura colombiana.
- La mayoría de estos desarrollos requieren equipos e implementos complementarios.
- Esto implica una inversión adicional por parte del caficultor.

6.1.1 CAFÉ

El café es una bebida muy popular que se obtiene de la semilla de un arbusto denominado Cafeto. Esta semilla es molida y tostada para obtener el polvo que se utiliza para la elaboración de la bebida.

El café es la segunda mercancía comercializada en el mundo, tras el petróleo. Se estima en 125 millones el número de personas que vive del cultivo del café, incluyendo 25 millones de pequeños productores. Cada año se beben 400.000 millones de tazas de café. Por tanto, en juego hay muchos intereses económicos y sociales extremadamente importantes.

Colombia es el primer exportador de café suave y tiene representación mundial a través de las tiendas Juan Valdez. Además, tiene una de las más grandes variedades de cafés especiales, ya que, dependiendo de la región de cultivo, el sabor, color y aroma del café varían; muchos de estos cafés se consideran Premium dentro del mercado mundial y pueden ser bastante costosos.

6.1.2 ARBOL

El cafeto, también denominado con el término coffea, es un arbusto silvestre perteneciente a la familia de plantas Rubiáceas; originarias del sur del continente asiático y de África sub tropical. Hoy en día, para la obtención de su fruto, algunas especies han sido domesticadas y son cultivadas por caficultores a lo largo de la región tropical del planeta.

Su raíz principal crece verticalmente, llegando hasta una profundidad de 50 centímetros en suelos aptos. A partir de ella, crecen raíces gruesas horizontales que sirven de soporte a raíces absorbentes más finas o delgadas. La raíz es muy importante, pues además de dar soporte y estabilidad a la planta, la provee del agua y los nutrientes necesarios para su desarrollo y crecimiento, tomándolos del suelo.

Figura 7: Planta de café

Sus hojas; con forma elíptica, son verdes durante todo el año. Son las responsables de la fotosíntesis, transpiración y respiración de la planta; por ello es importante que las hojas del cafeto sean abundantes y sanas. Con la aplicación de fertilizantes, la realización de podas, control de malezas y la regulación de la luz en el cafetal se consigue aumentar en crecimiento de ramas y hojas.

6.1.3 DENSIDAD DE LA SIEMBRA

La capacidad de producción de la tierra cultivada en café depende en buena parte del número de árboles que en ella se siembren y éste, a su vez, está en relación directa con la distancia de siembra utilizada. Para determinarla se deben tener en cuenta los siguientes aspectos:

- La disposición del cultivo: en hileras sencillas o dobles a través de la pendiente, utilizando una distancia entre surcos mayor a la empleada entre árboles, en bloques o parcelas de 11 surcos, 1 metro entre surcos y 40 árboles por surco a igual distancia, dispuesto en cuadro o triángulo, recomendado para terrenos con pendientes menores al 5% y en curvas a nivel.
- El sistema de producción según su luminosidad: sol, sombra o semi-sombra.
- La variedad a sembrar: porte bajo (Caturra, Colombia, Castillo) o alto (Borbón, Típica y Tabi).

Figura 8: Disposición del cultivo en hacienda el Roble

Algunas generalidades en las distancias entre plantas y considerando los cultivos en sombra se observan en la tabla a continuación:

Tabla 2:Distancias de siembra del sombrío y del café

Factor A. Distancias de siembra del sombrío	Factor B. Distancias de siembra del café
A_1. 6,0 x 6,0 m: 278 plantas/ha	B_1. 1,00 x 1,00 m (10.000 plantas/ha)
A_2. 9,0 x 9,0 m: 123 plantas/ha	B_2. 1,42 x 1,42 m (5.000 plantas/ha)
A_3. 12,0 x 12,0 m: 70 plantas/ha	B_3. 2,00 x 2,00 m (2.500 plantas/ha)

6.1.4 COSECHA

El café se cosecha una vez al año cuando la mayoría de las cerezas están maduras. Los productores utilizan diversos métodos de recolección, pero en el cultivo de café se utilizan principalmente dos métodos de recolección:

- **Picking:** Un proceso totalmente manual, en el que las cerezas maduras se seleccionan y recogen una a una, lo que exige a los recolectores recorrer el cultivo varias veces, pero produce una cosecha de alta calidad más homogénea.

Este método es el empleado actualmente en la hacienda el Roble, sin ninguna duda genera la mejor calidad en el producto, pero origina grandes inconvenientes para los trabajadores. Como ya se habló los tiempos de cosecha son muy importantes y cuando inicia la temporada el personal no puede perder tiempo, debe recolectar en bolsas el café durante jornadas de 12 horas o hasta que la luz del sol se los permita.

Además. se debe considerar que cada bolsa empleada se llena cada 10 metros aproximadamente y cada una de ellas queda con un peso de 22 kg. Cada vez que se termina una línea del surco, el cosechador debe devolverse con dos bolsas y desplazarlas hasta el camino por donde circula el tractor recolector, aproximadamente de 50 a 300 metros por recorrido, un esfuerzo realmente alto sin tener en cuenta las condiciones climáticas y otros factores a los cuales se expone el personal.

- **Stripping:** Un proceso, que puede ser manual o mecanizado, en el que los frutos se retiran todos de una vez cuando la mayoría de los granos estánmaduros. A menudo requiere una comprobación posterior para eliminar impurezas y cerezas inmaduras o ya fermentadas.

El siguiente paso es extraer el grano de la baya utilizando un método de procesamiento seco o húmedo.

El procesamiento húmedo produce un café clasificado como lavado o suave, de la calidad más apreciada y con una apariencia uniforme, sin defectos. Este método de

procesamiento requiere el uso de cerezas con maduración y consistencia uniforme. Una máquina pela los granos, que se colocan después en agua para que fermenten y se elimine la pulpaLos granos se lavan entonces y se dejan secar al sol o en una secadora. Por último, la descascaradora elimina las dos membranas protectoras restantes.

Con el procesamiento seco se produce un tipo de café menos homogéneo, conocido como natural. El fruto recién recogido se extiende al aire y al sol durante dos o tres semanas. La exposición solar seguida de acción mecánica elimina la piel, la pulpa y las membranas protectoras.

6.2 ROBÓTICA EN LA AGRICULTURA DE PRECISIÓN

La denominada Agricultura de Precisión contribuye específicamente a optimizar el uso de los recursos e insumos. Su filosofía se sintetiza en utilizar el recurso adecuado en el lugar y momento preciso, evitando de este modo aplicar, por ejemplo, un riego o producto fitosanitario en una dosis excesiva o en un lugar innecesario. Para ello utiliza una metodología similar a la utilizada frecuentemente por los seres vivos y que, a su vez, inspira el funcionamiento de los sistemas automáticos de control industrial: Medir, Procesar la información y Actuar, reiterando este proceso de manera continua, para así conseguir que los resultados se acerquen a los objetivos.

La eficacia de este modo de funcionamiento precisa conocer con detalle espacial y temporal las necesidades de los cultivos y poder actuar con la misma resolución espacio temporal de manera adecuada. Estas funciones, que a pequeña escala pueden ser realizadas de manera manual, precisan de sistemas automáticos para poder ser proyectadas a las escalas de producción que la demanda precisa. Es aquí donde los robots agrícolas, entendidos como maquinaria con capacidad de desenvolverse en el entorno agrícola para capturar información o actuar sobre el terreno de manera autónoma o semiautónoma, encuentran su justificación.

La idea de utilizar tecnología robótica en la agricultura es relativamente nueva. Las oportunidades de aumentar la productividad utilizando sistemas robotizados son inmensas, robots comerciales están poco a poco apareciendo en el campo de la agricultura de precisión, estos están dejando de ser exclusivamente un elemento abstracto de investigación académica sin aplicaciones reales a ser productos que realmente colaboran ejecutando actividades puntuales en los cultivos.

En los años noventa surgen el robot móvil. Una definición correcta de *robot móvil* plantea la capacidad de movimiento sobre *entornos no estructurados*, de los que se posee un conocimiento incierto, mediante la interpretación de la información suministrada a través de sus sensores y del estado actual del vehículo.

6.2.1 CLASIFICACIÓN DE ROBOTS

La subdivisión de los robots, con base en su arquitectura, se hace en los siguientes grupos:

- **Poliarticulados**

En este grupo están los robots de muy diversa forma y configuración cuya característica común es la de ser básicamente sedentarios (aunque excepcionalmente pueden ser guiados para efectuar desplazamientos limitados) y estar estructurados para mover sus elementos terminales en un determinado espacio de trabajo según uno o más sistemas de coordenadas y con un número limitado de grados de libertad. En este grupo se encuentran los manipuladores, los Robots industriales y los Robots cartesianos.

Figura 9: Robot poli articulado

- **Móviles**

Son robots con gran capacidad de desplazamiento, basados en carros o plataformas y dotados de un sistema locomotor de tipo rodante. Siguen su camino por telemando o guiándose por la información recibida de su entorno a través de sus sensores. Estos robots aseguran el transporte de piezas de un punto a otro de una cadena de fabricación.

Figura 10: Robot móvil

- **Androides**

Son robots que intentan reproducir total o parcialmente la forma y el comportamiento cinemática del ser humano. Actualmente los androides son todavía dispositivos muy poco evolucionados y sin utilidad práctica, y destinados, fundamentalmente, al estudio y experimentación. Uno de los aspectos más complejos de estos Robots, y sobre el que se centra la mayoría de los trabajos, es el de la locomoción bípeda. En este caso, el principal problema es controlar dinámica y coordinadamente en el tiempo real el proceso y mantener simultáneamente el equilibrio del Robot.

Figura 11: Robots androides.

- **Zoomórficos**

constituyen una clase caracterizada principalmente por sus sistemas de locomoción que imitan a los diversos seres vivos. A pesar de la disparidad morfológica de sus

posibles sistemas de locomoción es conveniente agrupar a los robots zoomórficos en dos categorías principales: caminadores y no caminadores. El grupo de los robots zoomórficos no caminadores está muy poco evolucionado. Los experimentos efectuados en Japón basados en segmentos cilíndricos biselados acoplados axialmente entre sí y dotados de un movimiento relativo de rotación son un ejemplo de estos últimos. Los robots zoomórficos caminadores multípedos son muy numerosos y están siendo experimentados en diversos laboratorios con vistas al desarrollo posterior de verdaderos vehículos terrenos, piloteando o autónomos, capaces de evolucionar en superficies muy accidentadas. Las aplicaciones de estos Robots serán interesantes en el campo de la exploración espacial y en el estudio de los volcanes.

Figura 12: Robot zoomórfico

- **Híbridos**

corresponden a aquellos de difícil clasificación cuya estructura se sitúa en combinación con alguna de las anteriores ya expuestas, bien sea por conjunción o por yuxtaposición. Por ejemplo, un dispositivo segmentado articulado y con ruedas, es al mismo tiempo uno de los atributos de los Robots móviles y de los Robots zoomórficos.

Figura 13: Robot Híbrido

6.2.2 APLICACIONES AGRÍCOLAS DEL ROBOT MÓVIL

Los robots en general tienen una infinidad de aplicaciones, todo depende de las necesidades locales encontradas por el equipo de diseño. Para dar una perspectiva de esto se citarán algunos ejemplos encontrados alrededor de mundo que brindan soluciones a la agricultura.

- **The FarmRobot**

El robot, o, los robots (ya que trabajan en grupo), con aspecto de insecto, permiten analizar una parcela para determinar donde establecer la plantación y la separación óptima para el cultivo.

Figura 14: The FarmRobot

- **Agrobot**

Permite una recolección automatizada de un fruto tan delicado como la fresa. Un robot con 20 brazos que analizan, al pasar por el cultivo, cuales fresas están en su punto óptimo de maduración, recolectándolas, dejando las demás que terminen de madurar en la planta.

Figura 15: Agrobot

- **The Lettuce Bot**

Este robot permite ahorra en herbicidas en el cultivo de lechugas. En una pasada sobre el campo, analiza las plantas que crecen sobre él, y realiza pulverizaciones localizadas solo en las malas hierbas.

Figura 16: The Lettuce Bot

6.3 MECÁNICA DEL ROBOT

6.3.1 CINEMÁTICA

Se entiende por cinemática el estudio del movimiento sin considerar las fuerzas que lo producen. Por tanto, se trata de estudiar tanto las propiedades geométricas como las temporales del movimiento. En estos casos se consideran, además del problema puramente geométrico involucrado en el posicionamiento estático, las variaciones en el tiempo de las posiciones y orientaciones; es decir las velocidades.

En los modelos geométricos y cinemáticos se involucra esencialmente el estudio de las relaciones existentes entre el espacio de las variables articulares y el espacio de trabajo, o espacio operacional, que suele ser el espacio cartesiano. Para el proyecto de grado desarrollado se tomaron en cuenta algunas condiciones.

Hipótesis básicas

a. El robot se mueve sobre una superficie regularmente plana.
b. Los ejes de guiado son perpendiculares al suelo.
c. Se supone que las ruedas y orugas se mueven con rodadura pura; es decir, el desplazamiento es despreciable en el periodo de control.
d. El robot no tiene partes flexibles.

e. Durante un periodo de tiempo suficientemente pequeño en el que se mantiene constante la consigna de dirección, el vehículo se moverá de un punto al siguiente a lo largo de un arco de circunferencia.

f. El robot se comporta como un sólido rígido, de forma que, si existen partes móviles, estas se situarán en la posición adecuada mediante el sistema inteligente.

- **Configuración Diferencial:** En este caso, las variables de control son las velocidades de las ruedas laterales. Sean wi y wd , las velocidades de giro de las ruedas izquierda y derecha, respectivamente. Si el radio de la rueda es c, las velocidades lineales correspondientes son Vi = wi * c y Vd = wd * c. En este caso, la velocidad lineal y la velocidad angular correspondientes vienen dadas por el modelo:

$$v = \frac{v_d + v_i}{2} = \frac{(\omega_d + \omega_i)c}{2}$$

$$\omega = \frac{v_d - v_i}{b} = \frac{(\omega_d - \omega_i)c}{b}$$

$$\omega_i = \frac{v - \left(b/2\right)\omega}{c} \qquad \omega_d = \frac{v + \left(b/2\right)\omega}{c}$$

Jacobiano modelo diferencial

$$\begin{bmatrix} x' \\ y' \\ \varphi' \end{bmatrix} = \begin{bmatrix} -(c\sin\varphi)/2 & -(c\sin\varphi)/2 \\ (c\cos\varphi)/2 & (c\cos\varphi)/2 \\ -c/b & c/b \end{bmatrix} \begin{bmatrix} \omega_i \\ \omega_d \end{bmatrix}$$

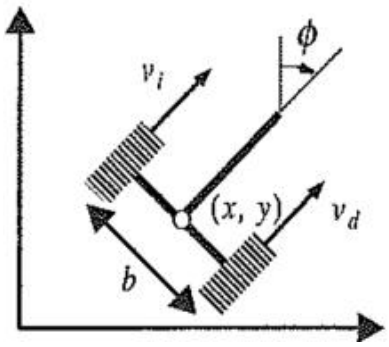


Figura 17: Ubicación cartesiana de la configuración diferencial

6.3.2 CINÉTICA:

El problema de la obtención del modelo dinámico de un robot es, por lo tanto, uno de los aspectos más complejos de la robótica, lo que ha llevado a ser obviado en numerosas ocasiones. Sin embargo, el modelo dinámico es imprescindible para conseguir los siguientes fines:

1. Simulación del movimiento del robot.

2. Diseño y evaluación de la estructura mecánica del robot.

3. Dimensionamiento de los actuadores.

4. Diseño y evaluación del control dinámico el robot.

- **Modelo de un robot mediante la formulación de Newton–Euler:**

La formulación de Newton-Euler parte del equilibrio de fuerzas y pares:

$$\sum_i^n \vec{F}_i + \sum_i^n \sum_j^n \vec{f}_{ij} = \sum_i^n m_i {}^N\vec{a}^i$$

$$\sum \vec{M}^{S/0} = \vec{r}_{os^*} \times m_s {}^N\vec{a}_{s^*} + \left[\bar{I}^{S/s^*} \cdot {}^N\vec{\alpha}^S + {}^N\vec{\omega}^S \times \left(\bar{I}^{S/s^*} \cdot {}^N\vec{\omega}^S\right)\right]$$

Un adecuado desarrollo de estas ecuaciones conduce a una formulación recursiva en la que se obtiene la posición, velocidad y aceleración del eslabón i referidos a las bases del robot a partir de los correspondiente del eslabón i-1 y el movimiento relativo de la articulación i. De este modo, partiendo del eslabón 1 se llega al eslabón n. Con estos datos se procede a obtener las fuerzas y pares actuantes sobre el eslabón i referidos a la base del robot a partir de los correspondientes al eslabón i+1, recorriéndose de esta forma todos los eslabones desde el eslabón n al eslabón 1

- **FUERZA MÁXIMA DE EMPUJE**

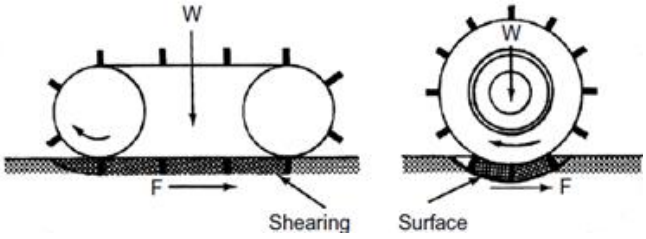


Figura 18: DCL En la oruga o banda del robot.

$$F_{max} = \tau A = (c + \sigma \tan\phi)A = cA + W \tan\phi$$

A: *Area proyectada del vehiculo* W: *Peso del vehiculo*

Los factores que afectan el rendimiento del vehiculo son:

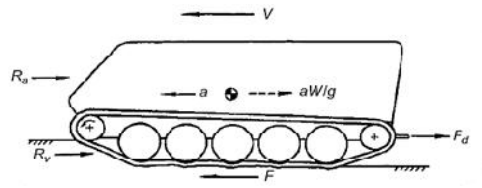


Figura 19: DCL aplicado en el robot tipo oruga.

$$F - R_a + R_v + F_d \pm R_g \quad R_g = W \sin \theta_s$$

F: Fuerza de empuje.

R_a: Resistencia Aerodinámica “Velocidades >48 Km/hr”.

R_v: Resistencia al movimiento.

R_ob: Resistencia debido a los obstáculos.

R_in: Resistencia debido a perdidas entre el Sprocket/Track.

R_t: Resistencia debido a la interacción con el terreno.

F_d: Fuerza de tracción por una carga externa remolcada.

R_g: Grado de resistencia.

W: Peso del vehículo.

θ_s: Angulo de pendiente.

También se debe tener en cuenta la distribución de presión bajo la oruga, como se observa a continuación:

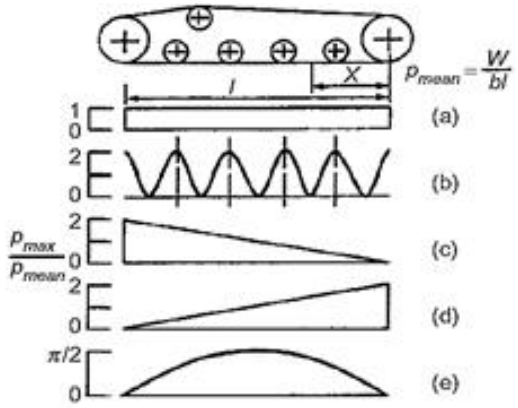


Figura 20: Distribución de la presión dentro de la oruga o banda.

$$F = b\int_0^l \left(c + \frac{w}{bl}\tan\phi\right)\left(1 - e^{-ix/k}\right)dx = (Ac + W\tan\phi)\left(1 - \frac{k}{il}\left(1 - e^{-il/k}\right)\right)$$

l: *Longitud de la oruga*

k: *parametro de deformacion de corte*

6.4 PARTES DEL ROBOT

Para referenciar de manera óptima los sensores y demás dispositivos que hacen parte de un robot, es fundamental fragmentar o dividir todo el robot por áreas que permitan estudiar individualmente cada una de sus funciones y escoger de mejor manera.

6.4.1 SISTEMA INTELIGENTE

Un sistema inteligente es aquel sistema capaz de resolver problemas complejos y multidisciplinares de una forma automática dando soporte a las decisiones de un experto. Las aplicaciones pueden ser numerosas y de una gran variedad, desde un sistema inteligente como soporte de decisión en telemedicina hasta un sistema inteligente para el tratamiento de datos o imágenes.

Necesidad o problema que resuelve: El desarrollo tecnológico alcanzado en nuestros días, unido al consecuente abaratamiento de los recursos, ha propiciado que cualquier entidad sea capaz de generar una cantidad ingente de datos. Esta facilidad para generar y almacenar información ha provocado la necesidad de

desarrollar y perfeccionar sistemas inteligentes que automaticen el análisis de la información y que ofrezcan instrumentos que faciliten la toma de decisiones.

Sistemas inteligentes en la teleoperación: Las diversas arquitecturas de control en teleoperación se diferencian básicamente por la información que se intercambia entre el maestro y el esclavo y por el tipo de sensorización que se requiere. En función de la información intercambiada entre el maestro y el esclavo pueden clasificarse en las siguientes categorías:

- **Esquema Posición-Posición:** la posición del esclavo se determina a partir de la del maestro y viceversa. No hay necesidad de sensores de fuerza.

- **Esquema Fuerza-Posición:** la posición del esclavo se determina por seguimiento del robot maestro y las fuerzas que aparecen sobre el esclavo se miden y se generan en el maestro mediante sus motores. Sólo se requiere medida de fuerzas en el esclavo.

- **Esquema Fuerza-Fuerza:** las trayectorias del maestro y el esclavo se determinan a partir de las lecturas de fuerza de ambos. También existe un control local de posición en ambos robots.

- **Esquema de Cuatro Canales:** en este caso hay intercambio de información tanto en posición como en fuerza. El análisis teórico de esta solución refleja que es capaz de proporciona transparencia infinita.

Las arquitecturas de teleoperación permiten diferentes grados de telepresencia que para el operador son fácilmente comparables y evaluables, afectando principalmente la estabilidad y la transparencia en el sistema.

- **Estimadores basados en la percepción del entorno para la trayectoria.**

Los estimadores basados en la percepción del entorno emplean sensores que suministran información sobre éste a partir de la cual se infiere la localización del robot móvil mediante comparación de esta información con otros datos o modelo conocido del entorno.

Los sensores empleados en este tipo de estimadores pueden clasificarse en seis grupos:

Estimación mediante marcas o balizas: En general las marcas (también conocidas como mojones o balizas especiales) son características del entorno de operación que un robot puede reconocer desde sus entradas sensoriales. Aunque puede entenderse que este proceso conlleva la percepción del entorno, la posición no se estima a partir del análisis o interpretación del entorno percibido, sino que es determinado de una forma más o menos directa en base al principio de triangulación, bien a partir de medidas de distancias, de ángulos o combinaciones de los dos.

Marcas naturales: Son aquellos objetos o características propios del ambiente y que tienen una función distinta a la de facilitar la navegación del robot. El principal problema en el posicionamiento mediante marcas naturales es poder detectar y extraer características distintivas del entorno de trabajo a partir de la forma en que se haya estructurado éste. El sistema sensorial por excelencia es la *visión computarizada*. La mayoría de los sistemas de visión empleados en la navegación mediante marcas naturales tratan de identificar segmentos verticales de longitud apreciable como son los marcos de una puerta, la intersección de paredes u objetos característicos como las fuentes de luz en el cielo raso.

Marcas artificiales: Éstas son formas geométricas (rectángulos, líneas, círculos, etc.) que, además, pueden incluir información adicional (por ejemplo, en forma de código de barras), tienen una posición fija conocida en relación a la cual el robot móvil puede estimar su posición y su único propósito es facilitar la navegación de éste.

Posicionamiento basado en mapas del entorno: El posicionamiento basado en mapas del entorno (también conocida como “map matching”) es una técnica en la cual el robot emplea sus sensores para crear un mapa de su entorno local. Este mapa local es luego comparado con un mapa global previamente almacenado en memoria. Si alguna correspondencia entre los mapas es encontrada (comparación de mapas) el robot puede computar su posición y orientación real en el ambiente.

Construcción de mapas: Un problema relacionado con la construcción del mapa es la exploración autónoma. En orden a elaborar un mapa, el robot debe explorar su entorno para mapear áreas no registradas. Habitualmente se supone que el robot comienza su exploración sin tener conocimiento del ambiente. Luego se sigue una determinada estrategia de movimiento dirigida a maximizar la cantidad de área mapeada en la menor cantidad de tiempo. Dicha estrategia de movimiento se llama estrategia de exploración y depende fuertemente del tipo de sensor empleado.

Técnicas de comparación de datos: Uno de los aspectos más importantes de la navegación basada en mapas es la comparación de datos, es decir, el procedimiento por el cual se establece la correspondencia entre un mapa local actual y el mapa global almacenado en memoria.

6.5 SISTEMA DE LOCALIZACIÓN

DEFINICIÓN

Los métodos desarrollados para la resolución de este problema se pueden considerar agrupados según dos tendencias:

1. Detección de marcas (naturales, artificiales) presentes en el entorno
2. Correspondencia entre la información suministrada por los sensores y un mapa a priori del entorno.

Las características a tener en cuenta para diseñar el sistema sensorial vendrán dadas por:

- Tamaño

- Consumo
- Simplicidad
- Redundancia
- Capacidad de operar en tiempo real
- Capacidad para detectar todo tipo de objetos en el entorno
- Resolución
- Exactitud
- Máxima y mínima distancia efectiva
- Campo de visión

Tipo de sensores que pueden ser utilizados, dentro de los más comunes y con mejores resultados:

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

Sistemas de navegación inercial

6.5.1 SISTEMA DE POSICIONAMIENTO GLOBAL:

La existencia de los sistemas de posicionamiento global (GPS) data del año 1978 cuando se puso en órbita el primer satélite de la serie Navstar. Su característica principal es situar un móvil en tiempo real en cualquier punto de la tierra con una precisión de pocos metros, en coordenadas y con control de la velocidad [Eng96]. El sistema de posicionamiento global está constituido por tres sectores: sector espacial, sector de control y sector de utilidades. El sector espacial está constituido por los diferentes satélites empleados, situados a una determinada altitud sobre la Tierra, y con un cierto periodo y plano orbital. La precisión del sistema está basada en la calidad de los relojes (osciladores) de muy alta estabilidad, colocados a bordo de los satélites GPS, lo que permite al sistema ser muy preciso en la escala de tiempos.

El sector de control está relacionado con el rastreo y seguimiento de los satélites a través de estaciones terrestres.

El sector de utilidades encuadra a los diferentes receptores existentes en el mercado, cuya función principal es recibir las señales emitidas por los satélites y emplearlas para el cálculo de la posición o para la determinación precisa del tiempo. Para realizar estas funciones todos los receptores GPS deben estar dotados de antenas integradas o externas para la captación de las señales emitidas por los satélites. El GPS se basa en el cálculo de las distancias desde una estación receptora a varios satélites, actuando como puntos de referencia fijos en el espacio. La medición de las distancias se realiza a través de los retardos de las señales recibidas, y calculando después la distancia a partir de ese tiempo. Esto sería posible si tanto los satélites como los receptores usasen la misma base de tiempos (sincronismo emisor-receptor), lo que exigiría que los receptores poseyeran un reloj muy estable. La solución es usar un nuevo satélite para conocer el retardo entre los relojes de los satélites y el receptor.

Como consecuencia de la necesidad de utilizar al menos cuatro satélites para determinar la posición del vehículo, se desarrollaron de forma paralela los sistemas DGPS, basados en el sistema diferencial, precisando únicamente de una estación en tierra que emita las correcciones proporcionadas por los satélites. Estos sistemas son capaces de ofrecer una exactitud con un margen de error de centímetros. El principio en el que se basan es simple. Partiendo de la hipótesis de que el error sobre una posición es el mismo para dos receptores situados en zonas no muy alejadas, es suficiente que uno de los dos esté ubicado en un punto perfectamente conocido para así determinar el error que conlleva el sistema GPS. Este error, entonces, permite corregir el punto calculado por el segundo receptor.

6.6 SISTEMA DE TRANSMISIÓN DE DATOS

DEFINICIÓN

El número de dispositivos conectados sigue aumentando (y se espera que alcance los 125.000 millones para 2030), por lo que las tecnologías inalámbricas maduras que los soportan también siguen recibiendo una gran atención en todo el mundo. NB-IOT (Narrowband IOT), LoRa y Sigfox son las puntas de lanza de todas las tecnologías de red de área amplia (LPWAN) de bajo consumo.

Muchas veces los ingenieros no conocen las ventajas e inconvenientes de estas 3 tecnologías y a menudo se confrontan entre sí; pero eso no es necesariamente una imagen precisa del ecosistema de conectividad. Cada una de estas tecnologías desempeñará un papel importante en el espacio de la IoT dependiendo del caso de uso, por lo que es fundamental comprender las características y diferencias de cada una.

- **SIGFOX:**

Sigfox es la compañía que despertó al mundo al potencial de los dispositivos IoT para usar conexiones de muy bajo ancho de banda. Sigfox es la más básica de las tres tecnologías, con las diferencias clave:

  - Sigfox tiene los módulos de radio de menor costo (<€5, comparado con ~€10 para LoRa y €12 para NB-IOT).
  - Sigfox es sólo un enlace ascendente. Aunque es posible un enlace descendente limitado, tiene un presupuesto de enlace diferente y es muy restringido.
  - Sigfox es un reproductor de red y tecnología de extremo a extremo.

Sigfox, sin embargo, parece tener ciertos problemas. Hasta ahora no ha desplegado redes significativas en los Estados Unidos y está luchando como compañía. Su modelo de negocio es un tanto ambicioso, ya que la compañía ya ha gastado cientos de millones de dólares para desplegar una red a la que la gente está pagando centavos para acceder. Sigfox piensa que mantener el costo de la aplicación a un nivel bajo es la manera de llevar a la gente a su mercado, pero esta línea de pensamiento ha hecho que sea difícil obtener suficientes ingresos. Al final, parece que Sigfox está teniendo dificultades en sus esfuerzos en establecerse como operador de una red celular alternativa para dispositivos de IoT.

- **LORA**

LoRa es una tecnología de modulación no celular para LoRaWAN. (Al igual que BPSK o QPSK es la modulación de NB-IoT.) Esos dos términos -LoRa y LoRaWAN-

no son intercambiables: LoRaWAN es el protocolo estándar para las comunicaciones WAN y LoRa se utiliza como tecnología de red de área extendida. LoRa representa una buena red radioeléctrica para las soluciones de IoT y tiene mejores presupuestos de enlace que otras tecnologías de radiocomunicaciones comparables. Pero fuera de unos pocos mercados en Europa, si desea conectarse a las redes LoRaWAN -o utilizar LoRa en absoluto- necesita desplegar su propio gateway de red.

Esto puede parecer un inconveniente, pero en realidad hace de LoRa una buena alternativa a la WiFi para los dispositivos de bajo consumo que necesitan conectarse en un edificio, como una fábrica o un hospital. Por ejemplo, es difícil encontrar un departamento de TI que apruebe la instalación de un dispositivo de terceros en su propia red debido a problemas de seguridad. La configuración de su propio gateway crea una red completamente separada y segura. De las tres tecnologías discutidas en este artículo, es la única que puede ser utilizada como tecnología "hágalo usted mismo"; cualquier empresa puede construir y utilizar su propio dispositivo conectado dondequiera que coloque la puerta de enlace.

6.7 ALGORITMO DE RAYECTORIAS

Existen diferentes formas de conseguir una trayectoria de un punto inicial a uno final a través de diferentes obstáculos. Lo que se busca es modelar matemáticamente lo que se quiere lograr en el terreno, para ello tomaremos de referencia algunos algoritmos encontrados.

- **A Estrella**

Antes de explicar el algoritmo de A* conviene explicar un poco que es un algoritmo voraz. Así pues, un algoritmo voraz (también conocido como ávido, devorador o goloso) es aquel que, para resolver un determinado problema, sigue una heurística consistente en elegir la opción óptima en cada paso local con la esperanza de llegar a una solución general óptima

- **Diagrama Voronoi**

Los Diagramas de Voronoi son uno de los métodos de interpolación más simples, basados en la distancia euclidiana, especialmente apropiada cuando los datos son

cualitativos. Se crean al unir los puntos entre sí, trazando las mediatrices de los segmentos de unión.

- **RRT y RRT***

Un Rapidly exploring Random Tree (RTT) es un algoritmo diseñado para buscar eficientemente espacios no convexos de alta dimensión mediante la construcción aleatoria de un árbol de relleno de espacio. El árbol se construye de forma incremental a partir de muestras extraídas al azar del espacio de búsqueda y está intrínsicamente sesgada para crecer hacia grandes áreas no buscadas del problema.

Rapidly exploring Random Tree (RTT)

Un RRT crece en forma de árbol arraigado en la configuración inicial usando muestras aleatorias del espacio de búsqueda. A medida que se dibuja cada muestra, se intenta una conexión entre ella y el estado más cercano en el árbol. Si la conexión es factible (pasa completamente a través del espacio libre y obedece cualquier restricción), esto resulta en la adición del nuevo estado al árbol. Con un muestreo uniforme del espacio de búsqueda, la probabilidad de expandir un estado existente es proporcional al tamaño de su región de Voronoi. Como las regiones Voronoi más grandes pertenecen a los estados en la frontera de la búsqueda, esto significa que el árbol se expande preferentemente hacia grandes áreas no buscadas.

Rapidly exploring Random Tree Star (RRT*)

RRT * es un algoritmo basado en RRT, que se centra en mejorar la calidad de las soluciones. Mientras RRT simplemente conecta un aleatorio con el nodo más cercano en el árbol, RRT * busca los nodos en una esfera de volumen con un radio específico para encontrar el nodo que hace el menor costo para llegar a la muestra aleatoria.

7 ANTECEDENTES

Desarrollos en la agricultura de precisión, temática de gran importancia en el presente proyecto propuesto, ha tenido antecedentes en otros trabajos y proyectos de grado de la UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA (UNAB) en la programa de ingeniería mecatrónica como es el caso del proyecto culminado y nombrado **"DISEÑO Y CONSTRUCCIÓN DE UN ROBOT MÓVIL COMO PLATAFORMA PARA EL APOYO A LAS LABORES EN LOS CULTIVOS DE CAFÉ."** realizado por el estudiante de ingeniera mecatrónica BRAJAN NICOLÁS RUIZ ROMERO.

El objetivo general de este proyecto fue: Desarrollar un robot móvil funcional capaz de transportar herramientas para el cuidado y cosecha del café en ambientes no estructurados. El énfasis de este proyecto fueron los tipos de ruedas y materiales para la construcción del robot. Pretendía ser funcional en campos pocos convencionales como los encontrados en la caficultura.

Para la validación del prototipo y comparación con los resultados esperados propuestos, se hicieron diferentes pruebas de validación. Estas pruebas fueron realizadas en un ambiente controlado, tanto en la incidencia lumínica como en la interacción del suelo con el sistema de tracción del prototipo móvil, esto debido a los distintos problemas que se pudieran presentar al momento de la adquisición de datos y el desplazamiento de la plataforma móvil en un entorno agrícola.

8 ANÁLISIS DE LA INFORMACIÓN RECOLECTADA.

Esta fase nos ayuda a divergen entre las diferentes soluciones encontradas en el estado de arte por medio de la metodología APTE, y así proporcionar una solución.

Metodología APTE

Es una metodología francesa que tiene como objetivo obtener una solución que se adapte mejor a las necesidades del usuario y logre un aumento en la calidad.

Los principios básicos de la metodología APTE son:

- Definir problemas como metas para lograr.
- Crear un marco común entre los miembros del equipo del proyecto.
- Ser objetivo para evitar comparaciones entre soluciones.
- Fomentar la creatividad para encontrar diferentes soluciones.

Para cumplir con estos principios, hay tres estrategias de análisis en la metodología APTE, que son el diagrama de análisis de necesidad, el diagrama de análisis funcional de necesidad y el diagrama de diagnóstico de costo funcional. Para este caso nos basaremos en el primer diagrama.

8.1 DIAGRAMA DE ÁNALISIS DE NECESIDAD

El primer diagrama es el diagrama de análisis de necesidad. Consiste en las siguientes partes:

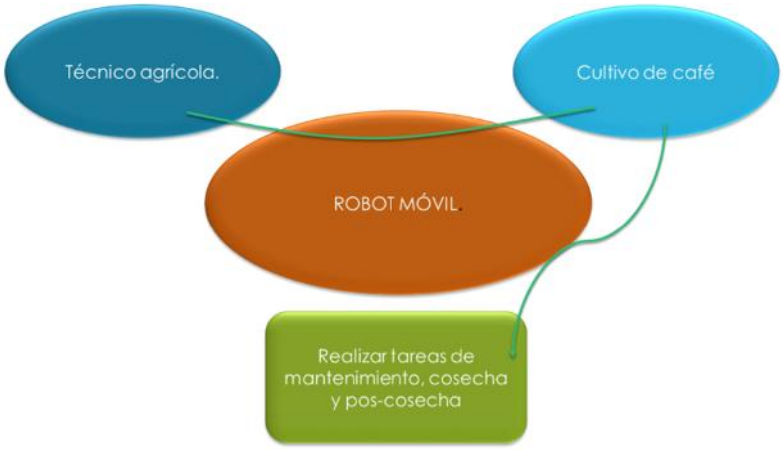


Figura 21: Diagrama de análisis de la necesidad.

9 ALTERNATIVAS DE SOLUCIÓN

Para conceptualizar las alternativas es importante encontrar los criterios de evaluación para las diferentes soluciones que posteriormente serán mostradas, esto con la intención de no sesgar una posterior selección.

9.1 DIAGRAMA DE ANALISIS FUNCIONAL DE NECESIDAD

Para la búsqueda de estos criterios se hizo uso del segundo diagrama de la metodología APTE, llamado diagrama de análisis funcional de necesidad, el cual tiene los siguientes elementos:

- Elementos externos
- Producto
- Funciones de servicio divididas en:
    - Función principal (FP)
    - Funciones de restricción (FC)

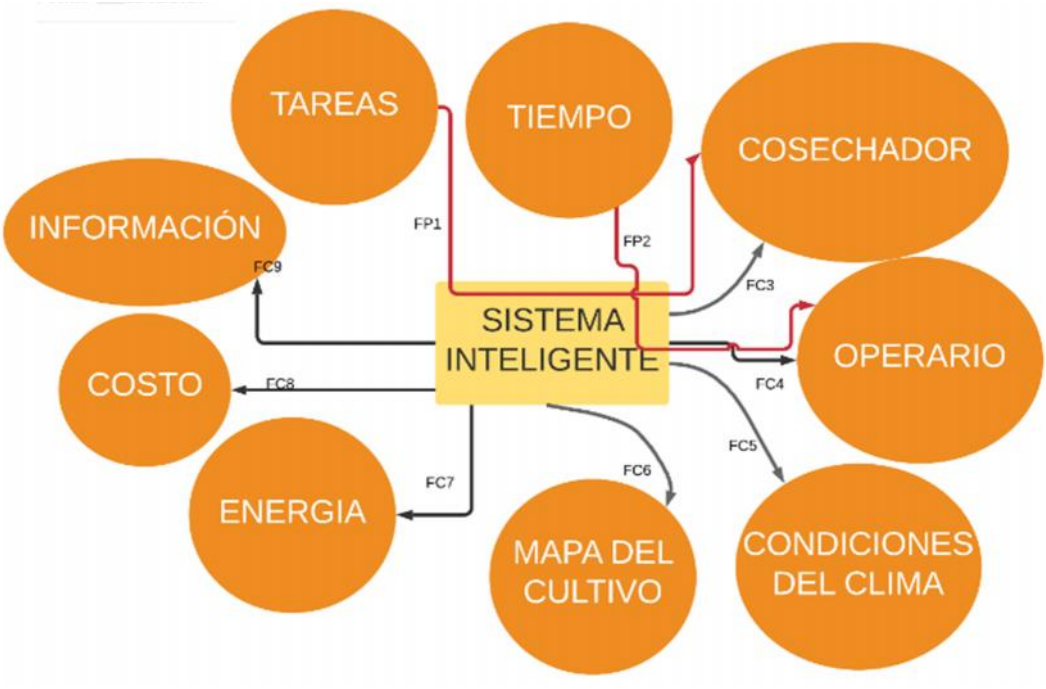


Figura 22: Diagrama de la metodología APTE

Teniendo en cuenta los siguientes elementos de medio externo basados en la convocatoria de Federación Nacional de Cafeteros de Colombia y visitas técnicas realizadas a la Haciendo el Roble situada en la Mesa de los Santos-Santander, se elaboró el diagrama de análisis funcional de la necesidad:

FUNCIONES PRINCIPALES
FP1: Ayudar en las tareas de siembra y cosecha.
FP2: Ahorrar tiempo en los procesos.

FUNCIONES RESCTRITIVAS:
FC3: Requerir del cosechador para cargar los insumos.
FC4: Manipular y guiar al robot por medio de un operario.
FC5: Considerar las condiciones de clima para evitar daños
FC6: Conocer el terreno donde se va a desplazar el dispositivo.
FC7: Implementar al sistema una fuente de energía permanente.
FC8: Determinar los costos porque son limitados.
FC9: Procesar la información de sensores para la toma de decisiones.

Las funciones principales y las funciones de restricción se convierten en criterios de evaluación que son la entrada de una herramienta de evaluación.

Las funciones de principales nos permiten también realizar un acercamiento a las posibles soluciones técnicas, con ayuda de la metodología de diseño FAST, que es el paso siguiente antes de realizar una evaluación de las diferentes alternativas.

9.2 METODOLOGIA FUNCTION ANALYSYS SYSTEM TECHNIC (FAST)

Las funciones principales encontradas de la metodología APTE son descompuestas en funciones técnicas para llegar a soluciones técnicas, Empezamos con una

función y preguntamos COMO esa función es realizada hasta llegar a una solución técnica específica.

La importancia de la metodología FAST es que gráficamente muestra las dependencias funcionales y crea un proceso para estudiar vínculos entre funciones al tiempo que explora opciones para crear sistemas mejorados.

FUNCIÓN PRINCIPAL DE TAREAS DE SIEMBRA Y COSECHA

Esta función es principalmente afectada por los resultados obtenidos de la segunda función principal propuesta, esto se debe al interfaz de mando y a las órdenes que este puede ejecutar.

Después trabaja normalmente la metodología mencionada.

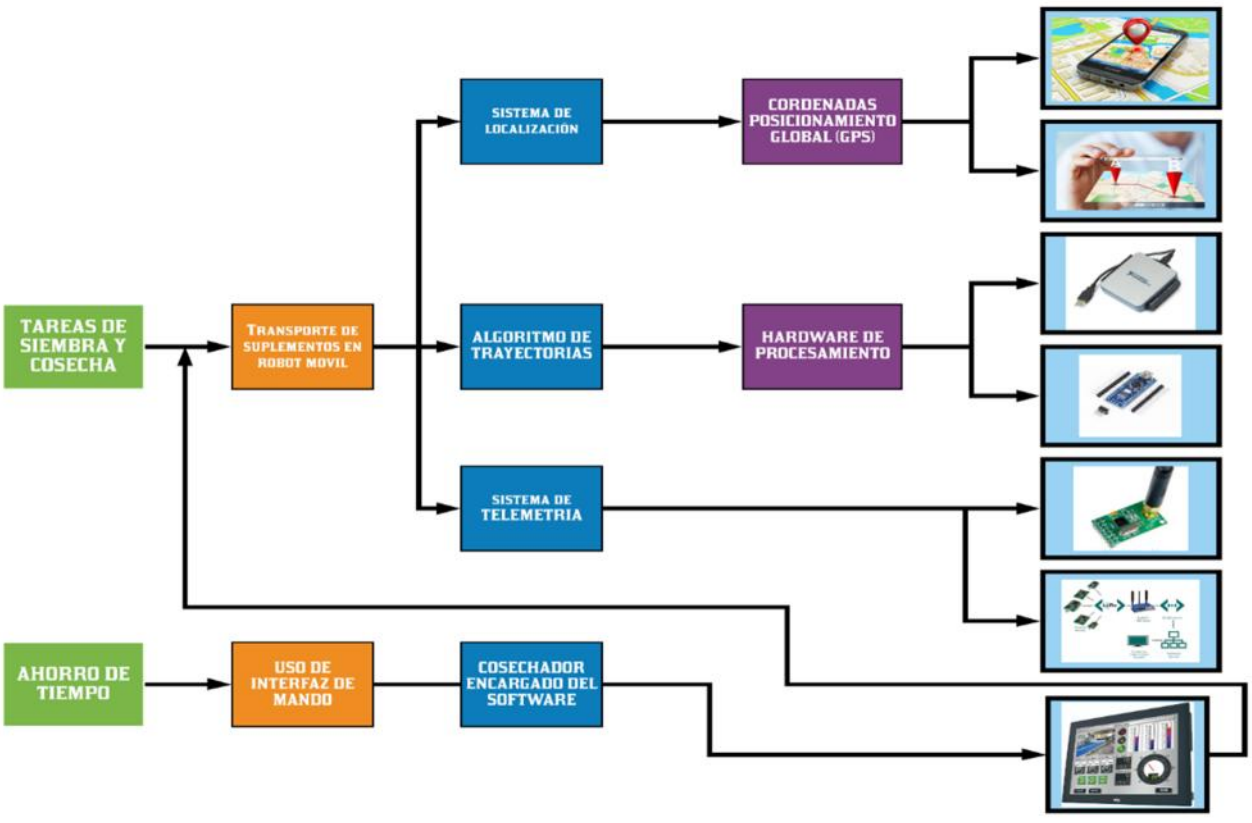


Figura 23: Diagrama FAST

10 EVALUACIÓN DE ALTERNATIVAS

Con los criterios de evaluación y con las posibles soluciones técnicas se procedió a realizar la evaluación a través de la matriz de calidad QFD.

10.1 METODOLOGÍA QFD

Quality Function Deployment (QFD) es una metodología japonesa utilizada en la gestión de calidad para seleccionar una alternativa u opción que mejor se adapte a las necesidades o demandas del usuario.

QFD se implementa siguiendo esta secuencia de pasos:

1. Definición de objetivos: identificación de los objetivos del cliente y los parámetros de evaluación.
2. Qué es: Caracterización del producto o servicio deseado por el cliente o usuario del producto.
3. Cómo: Identificación de los medios para cumplir con lo que es.

Estos pasos se pueden aplicar a lo largo de las diferentes fases secuenciales del ciclo de desarrollo del producto: diseño, fabricación, mantenimiento, etc.

La metodología QFD se utilizó para seleccionar la morfología más apropiada para el problema de diseño. Esto se hace aplicando una casa de calidad que correlaciona las necesidades o requisitos con los aspectos específicos de cada morfología.

ASPECTOS DE LOCALIZACIÓN:

Tabla 3: QFD En el aspecto de localización

Peso	Función	Sistemas de localización		
		Sensor GPS	Crear red local	Teleoperación
5	FP1: Ayudar en las tareas de siembra y cosecha	9	9	9
5	FP2: Ahorrar tiempo en los procesos	9	3	1
5	FC3: Requerir del cosechador para cargar los insumos			
3	FC4: Manipular y guiar al robot por medio de un operario	9	3	1
1	FC5: Considerar las condiciones del clima para evitar daños	3	3	9
4	FC6: Conocer el terreno donde se va a desplazar el dispositivo	1	3	9
5	FC7: Implementar al sitema una fuente de energia permanente			
3	FC8: Determnar los costos porque son limitados	9	1	1
5	FC9: Procesar la informacion de sensores para toma de decisiones	3	9	1
TOTAL		76	72	56

ASPECTOS DE HARDWARE DE PROCESAMIENTO

Tabla 4: QFD En el aspecto de hardware de procesamiento

Peso	Función	Hardware de procesamiento		
		Arduino	DAQ National Instruments	Raspberry
5	FP1: Ayudar en las tareas de siembra y cosecha	9	3	9
5	FP2: Ahorrar tiempo en los procesos	3	3	9
5	FC3: Requerir del cosechador para cargar los insumos			
3	FC4: Manipular y guiar al robot por medio de un operario			
1	FC5: Considerar las condiciones del clima para evitar daños			
4	FC6: Conocer el terreno donde se va a desplazar el dispositivo			
5	FC7: Implementar al sitema una fuente de energia permanente	3	3	3
3	FC8: Determnar los costos porque son limitados	9	3	1
5	FC9: Procesar la informacion de sensores para toma de decisiones	9	3	9
TOTAL		87	39	63

ASPECTO DE TELEMETRÍA

Tabla 5: QFD En el aspecto del sistema de telemetría.

Peso	Función	Sistema de telemetria	
		Radiofrecuencia RF	LORA
5	FP1: Ayudar en las tareas de siembra y cosecha	3	9
5	FP2: Ahorrar tiempo en los procesos	3	3
5	FC3: Requerir del cosechador para cargar los insumos	9	9
3	FC4: Manipular y guiar al robot por medio de un operario	1	9
1	FC5: Considerar las condiciones del clima para evitar daños	1	3
4	FC6: Conocer el terreno donde se va a desplazar el dispositivo		
5	FC7: Implementar al sitema una fuente de energia permanente	3	3
3	FC8: Determnar los costos porque son limitados	9	1
5	FC9: Procesar la informacion de sensores para toma de decisiones	1	9
TOTAL		96	138

ASPECTO DE INTEFAZ DE MANDO

Tabla 6: QFD En aspecto de interfaz de mando

Peso	Función	Uso de interfaz de mando	
		Hardware de control de mando	Dispositivos paralelos de uso
5	FP1: Ayudar en las tareas de siembra y cosecha	9	9
5	FP2: Ahorrar tiempo en los procesos	3	9
5	FC3: Requerir del cosechador para cargar los insumos	1	9
3	FC4: Manipular y guiar al robot por medio de un operario	9	1
1	FC5: Considerar las condiciones del clima para evitar daños		
4	FC6: Conocer el terreno donde se va a desplazar el dispositivo	9	9
5	FC7: Implementar al sitema una fuente de energia permanente	3	3
3	FC8: Determnar los costos porque son limitados	1	3
5	FC9: Procesar la informacion de sensores para toma de decisiones	3	3
TOTAL		101	123

11 MODELO MATEMÁTICO DEL ROVER

Se decidió trabajar con la configuración diferencial para aterrizar el modelo del robot móvil tipo oruga a trabajar. Dentro del modelo no se puede considerar el factor deslizamiento porque es una condición que se no se puede predecir. Los valores del robot están registrados en los códigos en Anexos.

Velocidad Lineal:

$$V = \frac{V_d + V_i}{2} = \frac{w_d R + w_i R}{2} = \frac{R}{2}(w_d + w_i)$$

$$V = \frac{V_d + V_i}{2} = \frac{w_d R + w_i R}{2} = \frac{R}{2}(w_d + w_i)$$

Velocidad Angular:

$$w = \frac{R}{L}(w_d - w_i) = \dot{\varphi}$$

CINEMÁTICA DIRECTA:

Matriz de cinemática directa

$$\left[V_x = \dot{x} = V.cos(\phi)\ V_y = \dot{y} = V.sin(\phi)\ w = \dot{\varphi} = \frac{R}{L}(w_d - w_i) \right]$$

Aplicando la inversa obtendremos modelo cinemática inversa

$$\left[\dot{x}\ \dot{y}\ \dot{\varphi} \right] = \left[\frac{R.cos(\phi)}{2} \quad \frac{R.cos(\phi)}{2} \quad \frac{R.sin(\phi)}{2} \quad \frac{R.sin(\phi)}{2} \quad \frac{R}{L} \quad -\frac{R}{L} \right] \left[w_d\ w_i \right]$$

Donde:

V_d = *Velocidad lineal rueda derecha*

V_i = *Velocidad lineal rueda izquierda*

w_d = *Velocidad angular rueda derecha*

w_i = *Velocidad angular rueda izquierda*

R = *Radio de las ruedas guias*

ϕ = *Ángulo de dirección*

12 VISITA AL CAMPO CAFICULTOR

Al visitar el campo caficultor en la Hacienda el Roble ubicada en la mesa de los santos, se tuvo la oportunidad de cruzar palabras con algunos cosechadores y contarles el proyecto a desarrollar. La respuesta fue positiva y encontraron muy productivo trabajar con el robot tipo oruga. Es una Hacienda de aproximadamente 450 hectáreas donde es agotador el traslado a pie de un punto a otro. Por tal razón la Hacienda ha sido dividida en pequeñas parcelas gracias a un camino construido para el paso del tractor recolector. Aunque las parcelas no son uniformes se puede garantizar que tienen un área entre 40.000 y 150.000 metros cuadrados. Los cosechadores manifiestan que en el proceso de recolecta de café; ellos de forma manual llenan un saco con granos equivalente a 20kg de peso y los dejan separados cada 15 m aproximadamente, por esta razón ellos deben dejar el saco de café en el punto donde termina el llenado, para seguir con la recolecta; al final, ellos deben devolverse y cargar estos sacos hasta la carretera por donde pasa el tractor recolector.

Ante esta problemática se les comentó a los cosechadores que el robot tendría la capacidad de pasar entre las plantas (por los senderos establecidos) hasta los sacos llenos de café a la espera que un operario cargue el saco encima del robot y este lo desplace hasta el punto por donde pasa el tractor recolector. Esta ayuda simplifica el trabajo y evita que los cosechadores hagan más de una vez el recorrido y no sufran ningún tipo de lección muscular debido al desplazamiento de esos pesados granos de café.

Además de esta tarea el robot podría ayudar en diferentes etapas del cultivo, por ejemplo; en la siembra, se necesitan miles de semillas o hinchazón de yemas, las cuales pueden ser cargadas por el robot. También puede transportar herramientas, (palas, podadora, etc) cuando no es temporada de siembra ni cosecha.

Figura 24: Condiciones del terreno.

12.1 PRUEBAS REALIZADAS EN LA PRIMERA VISITA

La visita fue aprovechada para aparte de conocer la ubicación o punto de partida del robot, la locación, terreno y necesidades del proceso; también se realizó una prueba de comunicación base para entender el comportamiento de las señales bajo esas condiciones.

Figura 25: Coordenada del punto de partida del robot.

12.1.1 DISPOSITIVOS EMPLEADOS

- **NRF 24L01**

Figura 26: NRF 24L01

El nRF24L01 es un transceptor de un solo chip de 2.4GHz con un motor de protocolo de banda base incorporado (Enhanced ShockBurst™), diseñado para aplicaciones inalámbricas de ultra baja potencia. El nRF24L01 está diseñado para funcionar en la banda de frecuencia ISM mundial a 2.400 - 2.4835 GHz. Se necesita una MCU (microcontrolador) y muy pocos componentes pasivos externos para diseñar un sistema de radio con el nRF24L01. El nRF24L01 se configura y opera a través de una interfaz periférica en serie (SPI). A través de esta interfaz, el mapa de registro está disponible. El mapa de registro contiene todos los registros de configuración en el nRF24L01 y es accesible en todos los modos de operación del chip.

Tabla 7: Pines de funcionamiento del NRF 24L01

Pin	Name	Pin function	Description
1	CE	Digital Input	Chip Enable Activates RX or TX mode
2	CSN	Digital Input	SPI Chip Select
3	SCK	Digital Input	SPI Clock
4	MOSI	Digital Input	SPI Slave Data Input
5	MISO	Digital Output	SPI Slave Data Output, with tri-state option
6	IRQ	Digital Output	Maskable interrupt pin. Active low
7	VDD	Power	Power Supply (+1.9V - +3.6V DC)
8	VSS	Power	Ground (0V)
9	XC2	Analog Output	Crystal Pin 2
10	XC1	Analog Input	Crystal Pin 1
11	VDD_PA	Power Output	Power Supply Output(+1.8V) for the internal nRF24L01 Power Amplifier. Must be connected to ANT1 and ANT2 as shown in Figure 30.
12	ANT1	RF	Antenna interface 1
13	ANT2	RF	Antenna interface 2
14	VSS	Power	Ground (0V)
15	VDD	Power	Power Supply (+1.9V - +3.6V DC)
16	IREF	Analog Input	Reference current. Connect a 22kΩ resistor to ground. See Figure 30.
17	VSS	Power	Ground (0V)
18	VDD	Power	Power Supply (+1.9V - +3.6V DC)
19	DVDD	Power Output	Internal digital supply output for de-coupling purposes. See Figure 30.
20	VSS	Power	Ground (0V)

- **ARDUINO NANO**

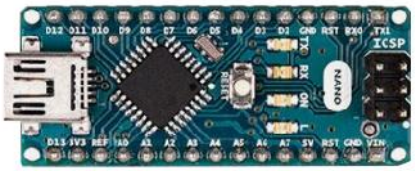


Figura 27: Arduino Nano.

Arduino Nano es una placa de desarrollo de tamaño compacto, completa y compatible con protoboards, basada en el microcontrolador ATmega328P. Tiene 14 pines de entrada/salida digital (de los cuales 6 pueden ser usando con PWM), 6

entradas analógicas, un cristal de 16Mhz, conexión Mini-USB, terminales para conexión ICSP y un botón de reseteo.

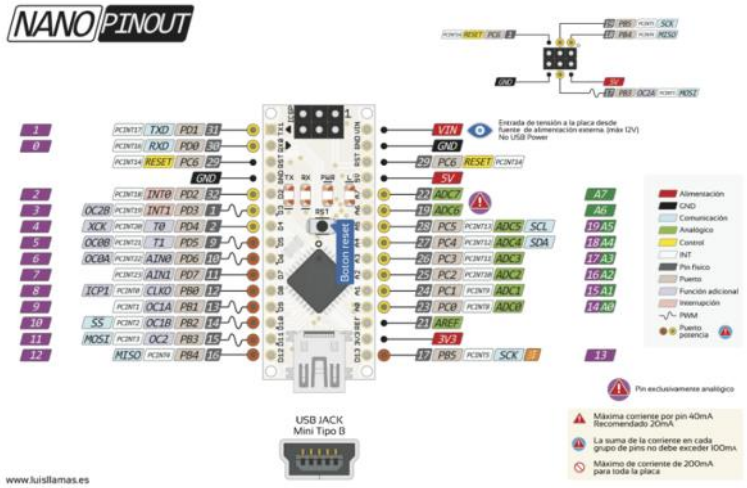


Figura 28: Pines de uso en arduino nano

- **HC 05(Módulo Bluetooth):**

Figura 29: Módulo HC 05

La microcomputadora MC68HC05X16 (MCU) es miembro de la familia MC68HC05 de microcomputadoras de chip único de bajo costo de Motorola. Esta MCU de 8 bits contiene un módulo de red de área de controlador a bordo (MCAN), completo con circuitos de interfaz, que comprende controladores de salida, comparadores de entrada y un generador VDD / 2. Además, el dispositivo contiene un oscilador en chip, CPU, RAM, ROM, EEPROM, convertidor A / D, salidas moduladas por longitud de pulso, E / S, interfaz de comunicaciones en serie, sistema de temporizador programable y watchdog.

- **DESARROLLO:**

Para la implementación de la tarjeta se trabajó el siguiente circuito:

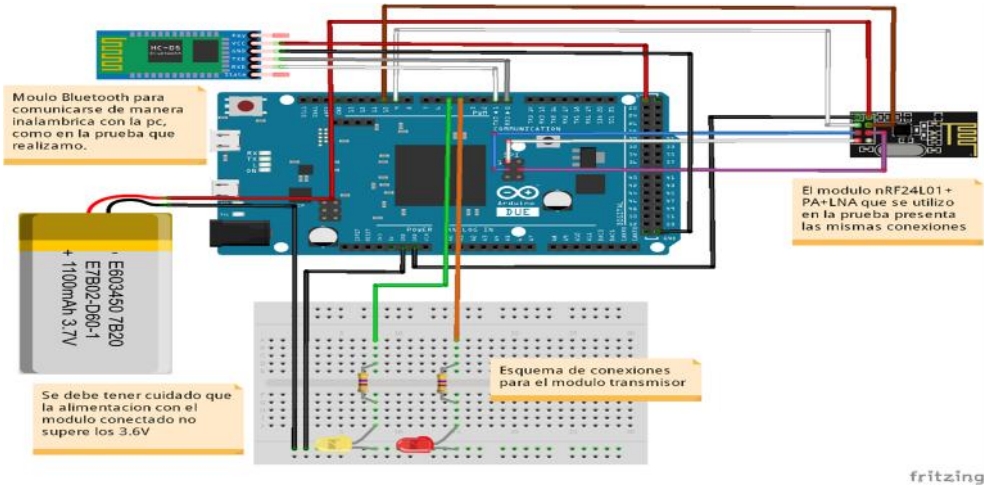


Figura 30: Circuito utilizado.

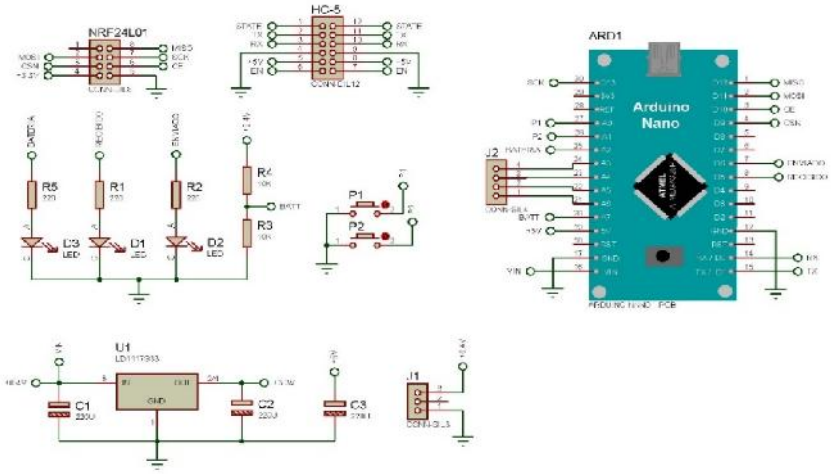


Figura 31: Diseño del circuito empleado en la tarjeta.

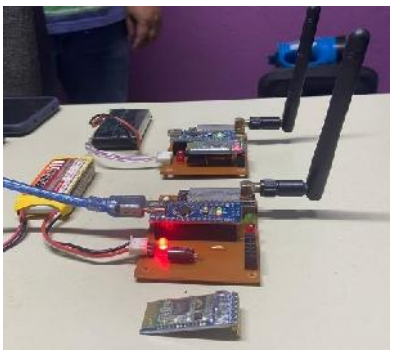


Figura 32: Modelo en tarjeta implementado.

12.1.2 RESULTADOS DE LA PRUEBA

Teniendo en cuenta que la prueba de envío y recepción de datos se realizó en cuatro puntos diferentes; los parámetros a considerar son:

- Punto maestro: 6°51’41”N 73°2’46”O
- Potencia máxima a: 0 Db
- Canal de comunicación: 100

Tabla 8: Resultados de la cantidad de datos perdidos en los cuatro puntos durante 10 pruebas

PUNTO 1	6°51'39"N 73°2'43"O
Altitud	1600 m
11	
3	
6	
1	
7	
33	
26	
37	
33	
30	
9,41620	Datos perdidos

PUNTO 2	6°51'37"N 73°2'44"O
	1640 m
15	
11	
3	
1	
7	
5	
3	
4	
4	
4	
3,09902285	Datos perdidos

PUNTO 3	
Altitud	
12	
24	
15	
17	
28	
27	
21	
31	
31	
23	
22,1232	Datos perdidos

PUNTO 4	6°51'58"N 73°2'52"O
Altitud	1630 m
78	
97	
101	
99	
80	
87	
89,8611508	Datos perdidos

Se puede concluir que el dispositivo empleado en las pruebas, trabajo satisfactoriamente dentro del mapa, pero tuvo problemas cuando el dispositivo sale de la línea de vista, además se encontró una elevación en el terreno correspondiente al punto de control o “administración” además el lugar cuenta con líneas de galpón de aproximadamente 100 m de largo con tejas de zinc, las cuales interfieren en las señales de tipo RF.

12.2 TAREAS A DESARROLLAR

Al tener la oportunidad de visitar el campo caficultor y de hablar con los cosechadores e ingenieros a cargo, se pudo determinar las tareas a desarrollar por el robot.

Teniendo en cuenta el tamaño de los surcos, los tiempos de trabajo, las herramientas empleadas y el terreno se puede concluir que:

- El robot va ser funcional en varias etapas del proceso del café y no solo en la etapa de cosecha.
- En la cosecha: El robot acompañara al cosechador mientras esté llena el saco con granos de café, su función inicia cuando el campesino termina el llenado y carga al robot con dicho producto. El robot inicia su operación haciendo el desplazamiento del producto hasta la carretera por donde transita el tractor recolector. Se espera que mientras el robot realiza el recorrido el cosechador pueda seguir seleccionando los granos de café; este proceso se repite las veces que sean necesarias o requeridas por el operador. Gracias a esto se reduce tiempo del proceso y se evita la exposición al desgaste físico que requiere esta tarea.
- En siembra: El robot seguirá las indicaciones del operador para desplazarse por la hacienda hasta los puntos de nuevas plantaciones, lo que se espera es que el robot movilice las semillas de café, con el fin de evitar al personal humano realizar recorridos de ida y vuelta con este producto, ahorrando tiempo en el proceso.
- Mantenimiento: Cuando no se está en época de siembra o cosecha es normal ver al personal humano en diferentes sectores de la hacienda generando el adecuado mantenimiento al cultivo, en algunas zonas el corte de maleza, en otras partes la fumigación con diferentes químicos dependiendo de cuál sea el caso. Por tal motivo lo que se pretende es evitar que el obrero se desplace hasta la casona para cambiar las herramientas de uso; por medio del robot pueda adquirir las herramientas de la nueva tarea y al mismo tiempo depositar las herramientas de la tarea terminada, ahorrando desgaste en el desplazamiento que debería hacerse para dicho cambio.

En general el robot tendría un uso diario en la hacienda de café, facilitando el proceso, ahorrando tiempo y desgaste físico que conlleva todos los procesos ejecutados para obtener el mejor café.

12.3 MAPA CAFICULTOR

Gracias a la visita realizada se pudieron obtener datos de ubicación por medio del GPS de los celulares y la aplicación de google maps. La intención era hacer un mapa detallado con los caminos a recorrer, pero a causa del confinamiento obligatorio no fue posible hacer más visitas a la hacienda.

Se establecieron punto de referencia por ser entradas o finales de surco o perímetros del área a trabajar.

Tabla 9: Puntos de referencia en el campo caficultor

PUNTO	N	W
1	6°51’43.58627"	73°2’53.09347"
2	6°51’41.99287"	73°2’56.85401"
3	6°51’42.34806"	73°2’53.26540"
4	6°51’37.29241"	73°2’52.78960"
5	6°51’35.96021"	73°2’50.45570"
6	6°51’40.81088"	73°2’45.70912"

Desde una vista satelital se puede apreciar la magnitud de la hacienda y el área de trabajo, se tiene de referencia, la casona y los galpones que son los lugares más cercanos.

Figura 33: Vista satelital de los puntos de referencia.

Se hizo un análisis con el equipo de trabajo para seleccionar el área de trabajo, finalmente se decidió trabajar cerca a los galpones, en el área triangular abarcada en los puntos 4, 5 y 6.

13 COMPARACIÓN SISTEMA DE COMUNICACIÓN

Tres fuertes comparaciones se pueden establecer para definir entonces porque para nuestro caso específico seleccionaremos la tecnología LORA:

- **COMPARACION 1:**

Aunque ambas redes se posicionan de manera similar en el mercado de IoT, tienen diferencias tecnológicas y de marketing significativas. Si bien SigFox pretende convertirse en un operador global de IoT, la alianza LoRa quiere proporcionar una tecnología que permita a otras compañías habilitar un IoT global.Ni SigFox ni LoRa ofrecen comunicaciones encriptadas.

Tabla 10: Primera comparación de sistemas de comunicación.

SISTEMA	SIGFOX	LORA
Frecuencia de banda	868/902 MHz (ISM)	433/868/780/915 MHz (ISM)
Rango urbano	3-10 km	2-5 km
Rango rural	30-50 km	15-20 km
Tamaño de paquetes	12 bytes	Definido por el usuario
Dispositivo por puntos de acceso	1 M	100k
Estado	En despliegue	Especificaciones lanzadas desde Enero 2015
Topología	Estrella	Estrella

- **COMPARACIÓN 2:**

Tabla 11: Sistema Sigfox lora

SISTEMA	SIGFOX	LORA

Ancho de banda de señal	On UNB (Ultra narrowband) Mayor eficiencia espectral y puede mitigar el ruido mejor	On CSS (Chirp Spread Spectrum)
Covertura	Menos que 17 km	Menos que 14km
Usos	Es muy práctico para transmisiones frecuentes y ofrece una mayor duración de la batería	Usa más ancho de banda
Modelo de negocio	Es un operador de red, por lo que se espera a que se implementen y pagar una tarifa de suscripción	Puede desplegarse sólo para cubrir su área local.
Seguridad	Es mejor para prevenir la repetición y los ataques de usuarios en el medio	Más debil en comparacion con sigfox pero se está mejorando constantemente

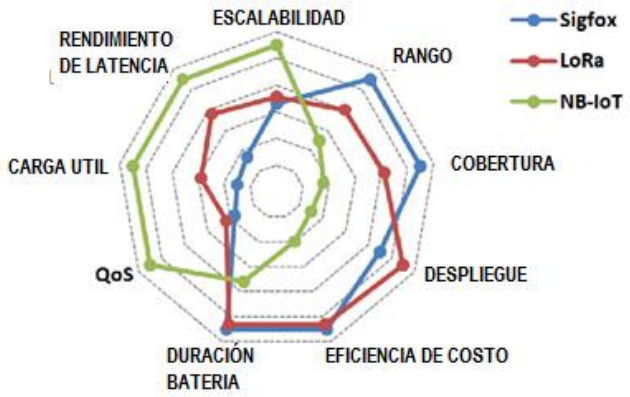


Figura 34: Ventajas a grandes rasgos en la comparación de las tecnologías evaluadas

- **CONCLUSIÓN**

Podemos entonces concluir que la tecnología LORA nos puede ofrecer:

A) Uso gratuito, en donde podemos configurar nuestra propia red.
B) Enviar tantos mensajes como se necesite, a velocidades de hasta 50 kbps
C) Alcance muy largo (hasta 40km en áreas rurales con antenas direccionales)
D) Baja potencia, dependiendo de la amplificación de su radio.
E) Selección de la frecuencia legal que está permitida usar.

A pesar de que debemos suministrar nuestra propia red y Gateway, los módulos permitan una fácil y desplegable códigos de programación para su fácil adaptación. Y nos debemos mover dentro de los módulos disponibles por Semtech y de bajas patentes.

14 COMPARACIÓN ALGORITMO DE TRAYECTORIAS

Se reviso la literatura en muchos artículos para determinar los 8 algoritmos de trayectorias más empleados, todos están registrados en anexos de este documento.

Comparación entre RRT y RRT*

El algoritmo Rapidly exploring Random Tree (RRT) se esfuerza por buscar el espacio para un camino factible mediante el “crecimiento” de un árbol de relleno de espacio.

Este árbol está arraigado en el punto inicial; en cada expansión incremental del crecimiento del árbol, se genera una semilla aleatoria, se inspeccionan nodos del árbol ya existente; basándose en ciertas reglas, se seleccionará un nodo como el siguiente que se va a expandir (nodo padre). El árbol comienza el crecimiento (navegación del vehículo) desde el padre, hasta que por ciertas reglas detienen el crecimiento del árbol, y este punto de parada (nodo hijo) se agrega al árbol.

Una ventaja de RRT es que, en cada expansión del árbol, las restricciones no holonómicas (aquellas que dependen de la velocidad, que sean integrable y no se puedan obtener derivando una restricción holónoma) podrían ser tomadas en cuenta en la navegación, por lo tanto, todos los nodos agregados en el árbol están disponibles para la dinámica del vehículo. La trayectoria de vuelta de RRT es una innata y factible.

La desventaja, sin embargo, debido a la aleatoriedad en la generación de semillas, la trayectoria de RRT podría ser un zigzag. Para superar este problema, se puede sesgar la generación de las semillas, o un ajuste del RRT original denominado como RRT*.

Aparte de conectar al hijo con el nodo padre, RRT* también inspecciona cada uno de los nodos que se encuentran dentro de un vecindario del niño recién agregado. Si un nodo puede rastrear hasta la raíz del árbol a través del hijo a una distancia más corta que la conexión actual, el padre de este nodo se cambiará a hijo. El RRT*

básicamente intenta suavizar las ramas de los árboles en cada paso, por lo tanto, el zigzag desaparecerá. Matemáticamente se ha demostrado que con el número de nodos alcanzado el infinito, la trayectoria de vuelta desde RRT* será la más corta.

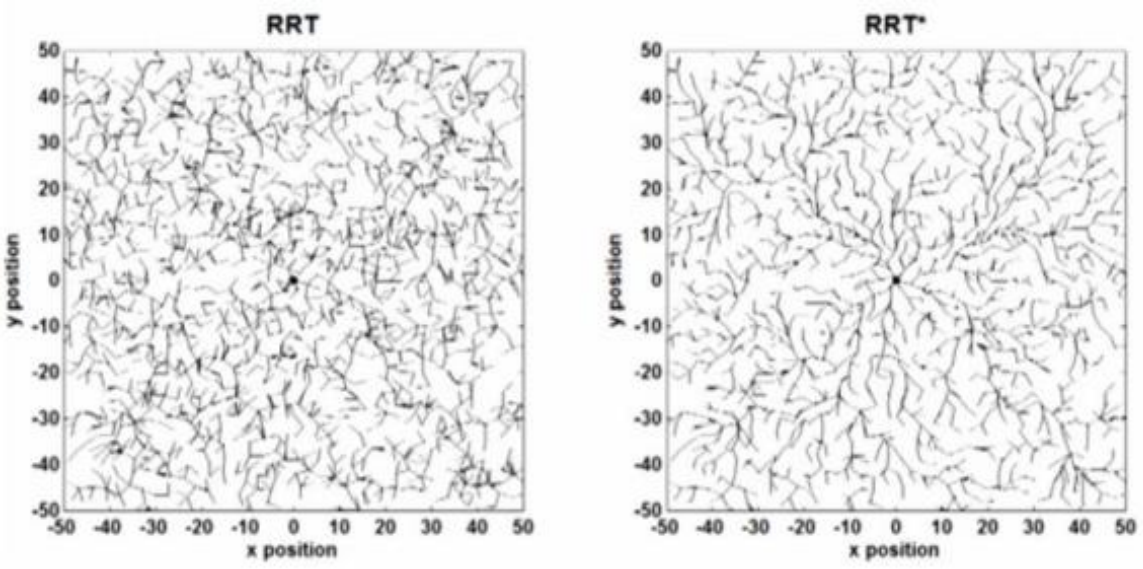
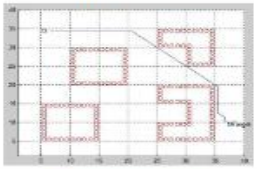
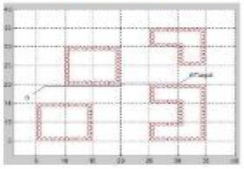
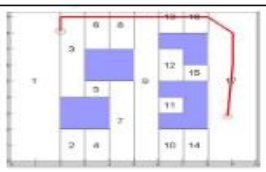


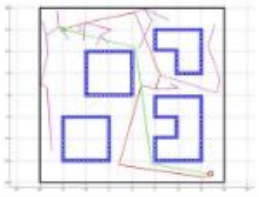
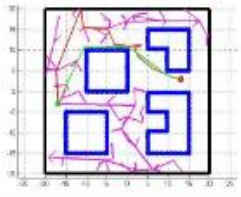
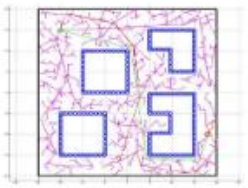
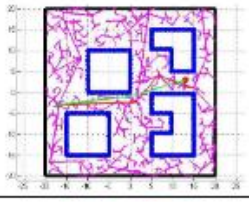
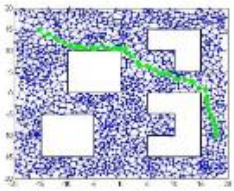
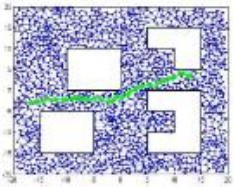
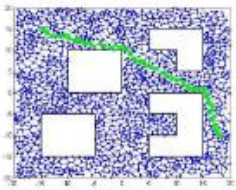
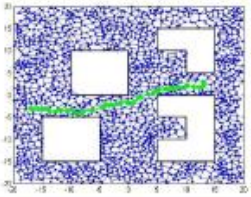
Figura 35: Prueba con 2000 nodos.

La diferencia se nota bastante. Se observa como el RRT es más caótico que el RRT*.

Se puede leer más acerca de todos los posibles algoritmos de trayectorias que esta registrado en anexos de este documento.

Tabla 12: Comparación de algoritmos

Nombre	MISMO MAPA, 4 OBSTACULO Y PUNTOS FIJOS	CON PUNTO DE PARTIDA Y OBJETIVO VARIANTES	TIEMPO	INTERACCIONE	CONCLUSIONES
ALGORITMO A*			1. 0,03s -- 2. 0,1s	x	Tanto el A*, el Voronoi como el RRT (dependiendo de la exactitud que se busque en el algoritmo) el camino que toman es el más corto. Sin embargo hay peros. El A* tiene un problema y es que si el mapa es muy grande y tiene muchos obstáculos el cáculo computacional que necesita es enorme, entonces es muy buen método para zonas pequeñas y con pocos obstáculos. El más fácil de implementar es el A*.
DESCOMPOSICIÓN DE CELDAS			1. 0,13s -- 2. 0,13s	x	El algoritmo de menor costo es el de descomposición de celdas puesto que la ruta que sigue es siempre siguiendo los nodos que menos coste tienen.

ALGORITMO RRT (A.5)			12,04s	120	
ALGORITMO RRT (A.1)			28,37s	502	RRT tiene un problema similar pero en este caso se puede solucionar utilizando el RRT*, el cuál el coste es menor para mapas más grandes. A pesar de esto, este algoritmo es más eficiente que en los otros casos. Tomando todo esto en consideración podemos afirmar que la mejor opción en general es el RRT, puesto que es el mas eficiente en la mayoría de los casos, aunque como he dicho antes, depende del caso concreto en el que se trabaje es mejor
ALGORITMO VORONI (SIMULACIÓN 1)			X	X	
ALGORITMO VORONI (SIMULACIÓN 2)			1. 1,64s -- 2. 1,49s	X	El Voronoi depende del mapa, el cuál, al igual que los anteriores, si la región en la que el vehículo tiene que moverse es muy grande necesita hacer muchos cálculos entre vértices el cual puede ser demasiado grande. Estos problemas dependen mucho del caso en el que se utilicen o como de preciso es el algoritmo

Teniendo en cuenta los resultados y las características del proyecto se puede concluir que la mejor trayectoria para aplicar es la generada por el algoritmo RRT*

15 ESTADO DEL ROBOT AL INICIO

Como punto de partida en este proyecto, se necesita contar con un robot al cual aplicarle el sistema inteligente a desarrollar, pero en este caso se cuenta con parte de dicho robot.

Actualmente el robot está incompleto porque no cuenta con sistema electrónico ni de potencia y se le debe hacer algunas modificaciones a la parte mecánica.

Figura 36: Evidencia 1 del estado del robot al ser recibidos.

Figura 37: Evidencia 2 del estado del robot al ser recibido

15.1 REPARACIONES REALIZADAS:

Durante el desarrollo de este proyecto se tuvo que realizar una reparación de la parte mecánica del robot, con el fin de mejorar su funcionamiento y permitir la puesta en marcha del mismo.

Por tal razón se identificó los problemas y se encontró que:

- Se compró y maquinó el material de los discos para la construcción de los 4 piñones guías traseros, 4 piñones guías medianos y 2 piñones pequeños.
- Se completaron 14 separadores de disco pequeño y mediano para evitar el juego o desajuste de los elementos.
- Se implementaron 24 separadores a los amortiguadores con el fin de fijarlos en la posición central del eje.
- Se enderezaron 3 láminas del sistema de tracción con el fin de alargar su vida útil.
- Se compraron 10m de Varilla roscada para completar los huecos en las esquinas del sistema de tracción, con el fin de evitar dobleces en las puntas.
- Se le añadió silicona liquida con el fin de evitar filtraciones de agua que puedan afectar su funcionamiento.
- Toda la instalación electrónica y de potencia, ya que el robot no contaba con ningún elemento electrónico que lo hiciera andar o funcionar.

A continuación, se visualiza algunas figuras con las reparaciones realizadas y mencionadas con anterioridad.

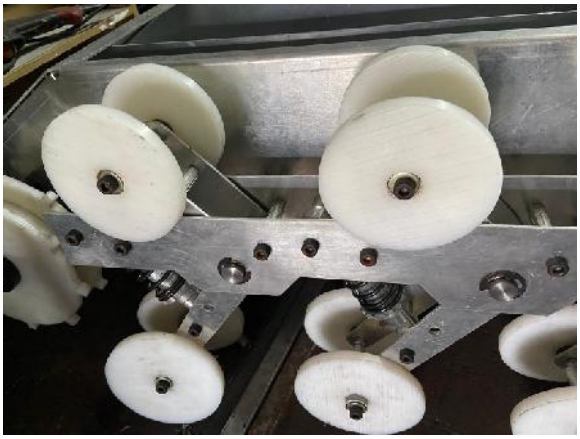


Figura 38: Discos guías nuevos

Figura 39: Ajustes centrados de amortiguadores.

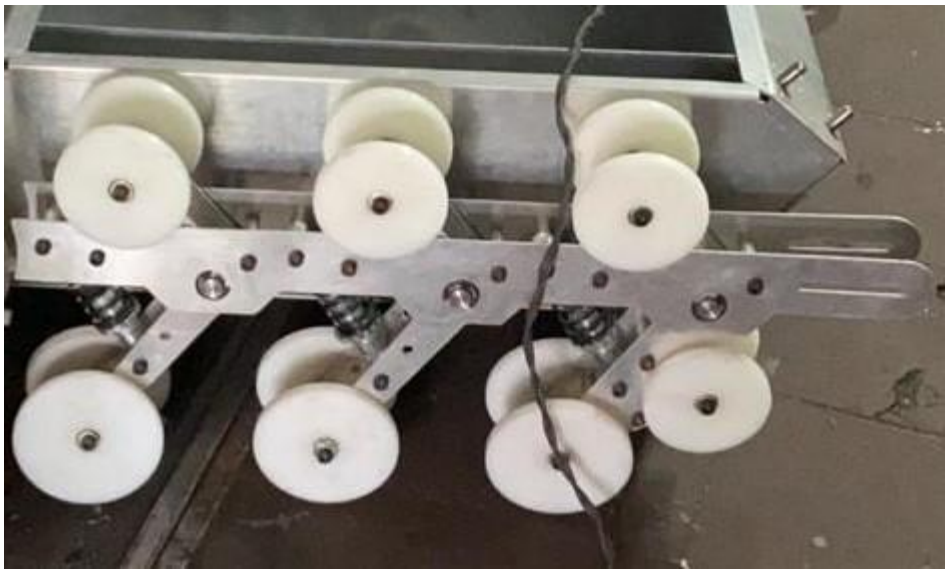


Figura 40: Ajuste y alineación de placas.

16 TÉCNICAS Y COMPONENTES PARA LA IMPLEMENTACIÓN

Para la implementación de la investigación realizada al entorno real, se decidió construir un robot a escala, que permitiera probar y corregir fallas, evitando sobre costos en las pruebas finales.

Su objetivo se basa en la óptima toma de datos, para posterior análisis, se quiere dejar todo construido, listo y probado en el robot a escala, con el fin de asegurar los métodos escogidos en este proyecto de grado, además de facilitar la implementación en el robot real y sus respectivas pruebas en el cafetal.

16.1 ROBOT A ESCALA:

Teniendo en cuenta el robot tipo oruga que va a funcionar en el cafetal, se construyó un robot más pequeño, pero con diseño y configuraciones muy similares. Por medio de la impresión 3D se pudo obtener el chasis y oruga del robot.

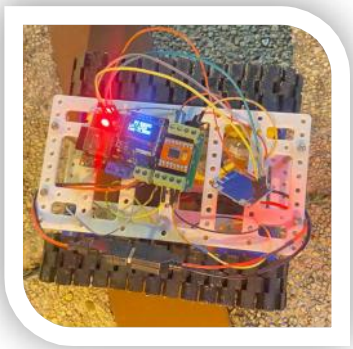


FIGURA 41: Robot a escala

La figura 41 ilustra la vista superior del robot empleado para realizar las simulaciones y pruebas que permitan cumplir con todos los parámetros establecidos en los objetivos específicos.

Para la unidad electrónica se utilizó la plataforma KiCAD la cual facilito la construcción del circuito electrónico que rige en el robot e implemento los diferentes sensores y dispositivos que permiten el óptimo funcionamiento.

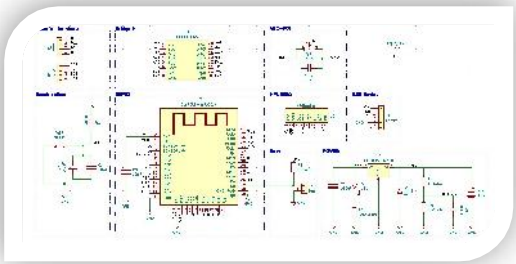


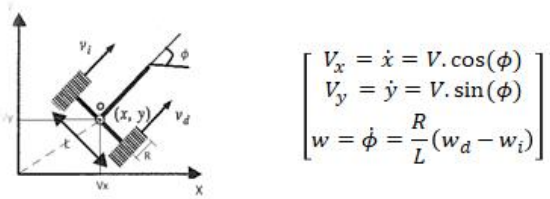
Figura 42: Circuito electrónico del robot a escala.

Por medio del micro SP32 se estableció la base del circuito y se construyó la tarjeta que permita manipular el robot. Se debe tener en cuenta que la única diferencia entre el robot a escala y el robot real, es el cambio del puente H, porque utilizan motores diferentes, pero el principio de todo lo demás es prácticamente el mismo.

Figura 43: Tarjeta construida para robot a escala

16.2 IMPLEMENTACIÓN DEL MODELO MATEMATICO:

Cuando se estaba construyendo el robot, se decidió en primer lugar hacer las pruebas del modelo matemático para entender el funcionamiento del mismo desde las ecuaciones. Por tal razón se decidió utilizar el modelo diferencial (Figura 44).



$$\begin{bmatrix} V_x = \dot{x} = V.\cos(\phi) \\ V_y = \dot{y} = V.\sin(\phi) \\ w = \dot{\phi} = \frac{R}{L}(w_d - w_i) \end{bmatrix}$$

Figura 44: Modelo diferencial aplicado al robot

Por medio del programa Matlab se realizó la simulación que permite analizar los resultados cuando se ejerce movimiento lineal en el robot.

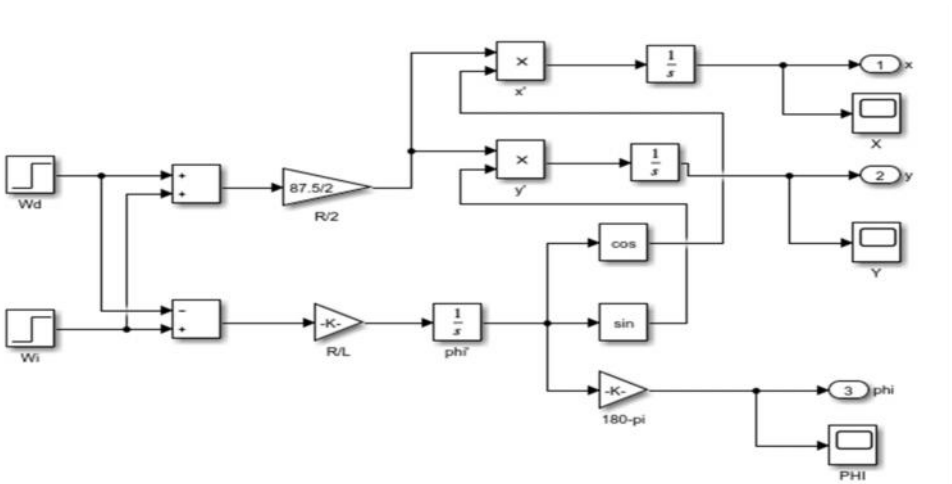


Figura 45: Simulación del modelo matemático.

Como se puede observar en la Figura 44 se construyó en la herramienta Simulink la ecuación matricial que permite entender los movimientos que sufrirá el robot ante dos entradas. Se debe recordar que se deben tener los valores de las variables R (Radio de piñón guía) y L (distancia entre ejes) los cuales son; 87,5 mm y 250 mm respectivamente.

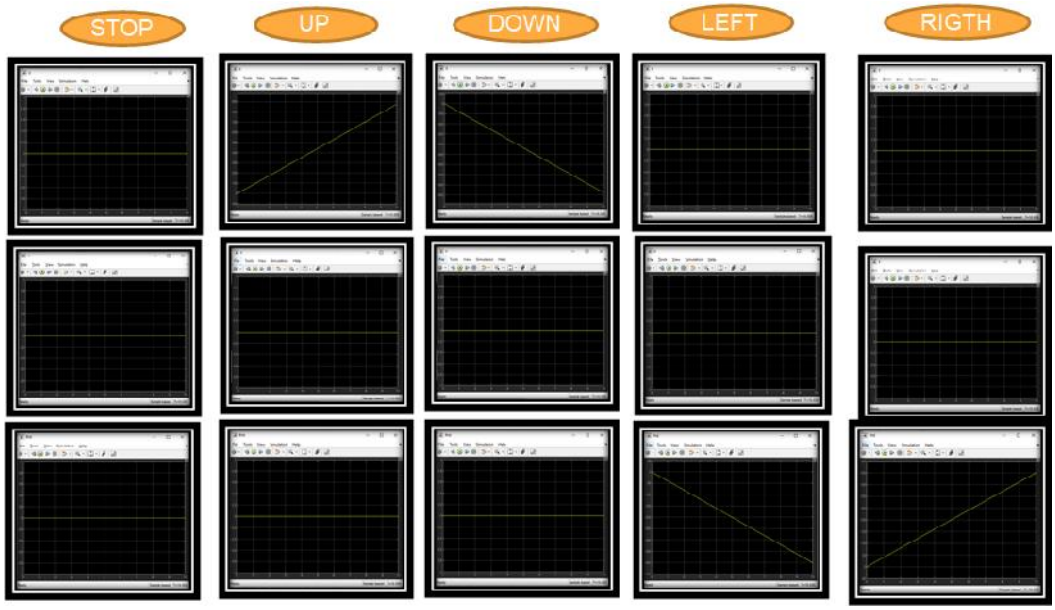


Figura 46: Resultados obtenidos del modelo matemático

Los resultados obtenidos son satisfactorios por que se asemejan a los esperados, y permiten entender la velocidad en “X”, velocidad en “Y” y en ángulo de rotación que lo rige.

16.3 SISTEMA DE GEOLOCALIZACIÓN IMPLEMENTADO

El robot pequeño por medio de la interfaz gráfica visualiza en todo momento las variables de latitud y longitud; suministrados por el sensor GPS incorporado al microcontrolador SP32 y se decidió trabajar la lógica de referencia por marcas naturales, las cuales permiten diseñar el mapa a transitar por ubicación de puntos en específico.

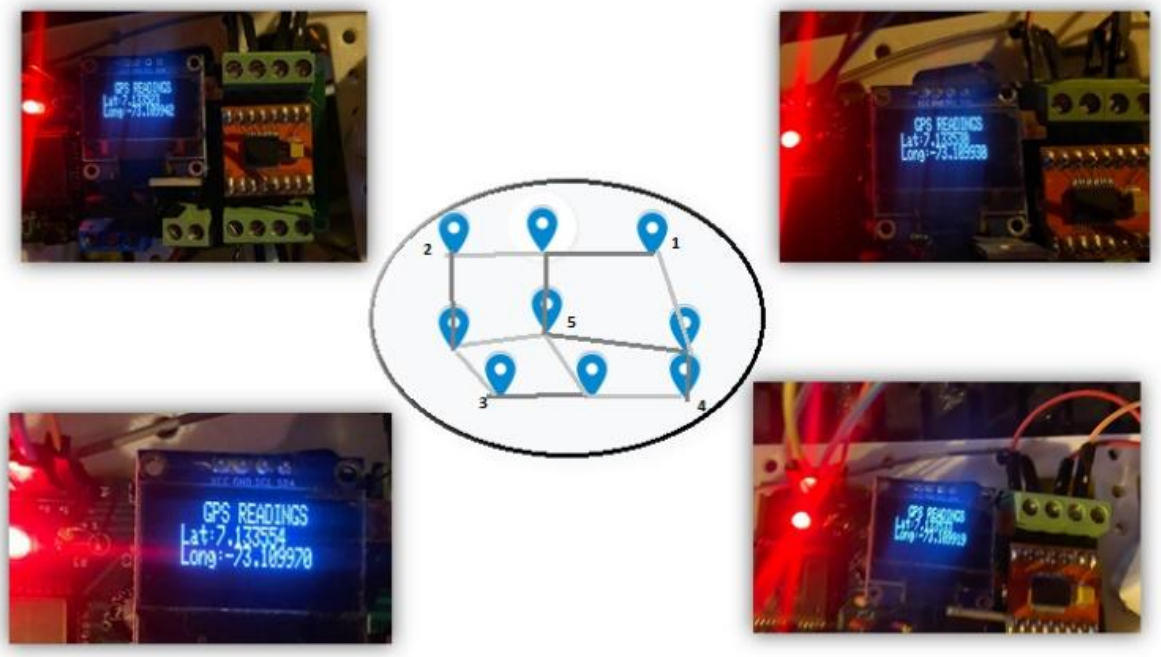


Figura 47: Simulación de geolocalización a escala.

La simulación permitió crear un mapa en un área común a la vivienda de residencia actual. Este paso es fundamental para indicar al robot los límites del mapa, los posibles caminos y los prohibidos; esto permite caracterizar el proceso de selección.

16.4 IMPLEMENTACIÓN DEL ALGORITMO DE TRAYECTORIAS.

En la robótica tradicional es normal que el usuario indique la trayectoria que desea ser transitada por el mecanismo móvil, pero en este caso, el sistema planteado a ser ejecutado en este proyecto de grado, requiere que por sí "solo" tenga la capacidad de entender su entorno y diseñar la trayectoria teniendo en cuenta solo el punto actual de su ubicación y el punto final a donde se quiere llegar.

Por tal motivo, en este punto se hicieron varias pruebas, se dividió el trabajo con el fin de atestar específicamente cada problemática. El primer paso es entender el entorno donde trabajara; recordando que son distancias largas (+1km) para comunicarse y trasladarse en un entorno complicado (cafetal).

Posteriormente se debe entender la geometría de la tierra, y caracterizarla por medio de figuras (triángulos esféricos) que nos permita trazar la distancia a recorrer y entender en ángulo indicado de ubicación.

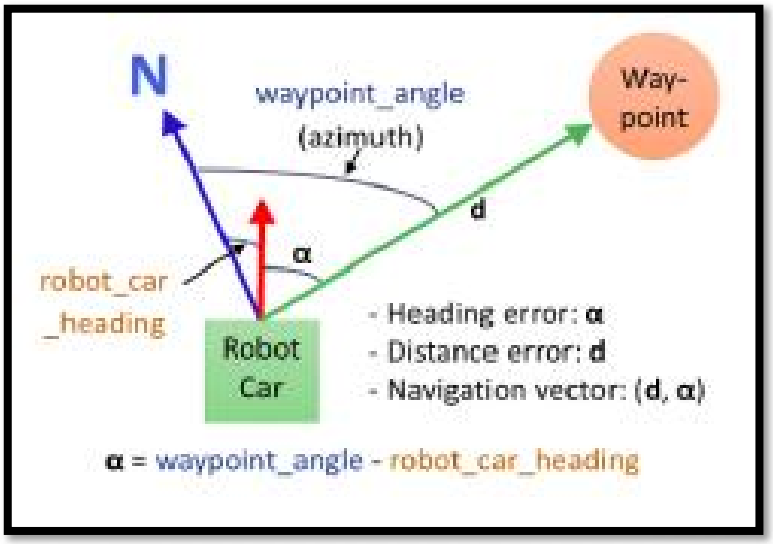


Figura 48: Formula para el ángulo azimut

En la figura 48 se ilustra la manera para hallar el ángulo entre el robot y el punto a donde se quiere llegar, en matemáticas este ángulo es llamado azimut y representa

el arco medido sobre el horizonte celeste que forman el punto cardinal Norte y la proyección vertical del astro sobre el horizonte del observador situado en alguna latitud. Por proyección vertical, entendemos el corte con el horizonte que tiene el círculo máximo que pasa por el cenit y el astro.

La altura y el acimut son coordenadas que dependen de la posición del observador. Es decir que, en un mismo momento, un astro es visto bajo diferentes coordenadas horizontales por diferentes observadores situados en puntos diferentes de la Tierra. Esto significa que dichas coordenadas son locales, con esta coordenada y teniendo cuenta el principio de Haversine o fórmula del semiverseno.

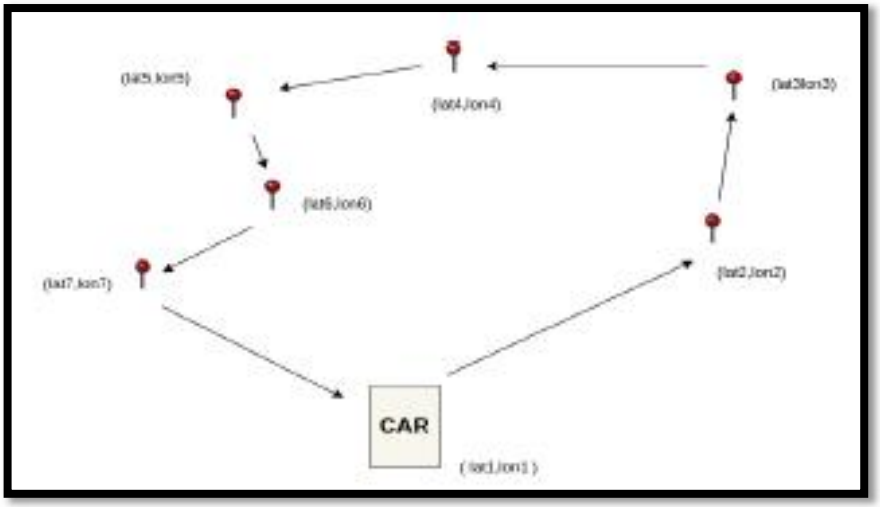


Figura 49: Construcción de mapa por medio de puntos o marcas.

Dicha fórmula permite la navegación astronómica, en cuanto al cálculo de la distancia de círculo máximo entre dos puntos de un globo sabiendo su longitud y su latitud.

Las fórmulas son introducidas en Matlab con sus respectivos valores en variables y esto le permitirá al sistema, evaluar y conocer el ángulo que el robot debe mantener y la distancia aproximada a su destino final.

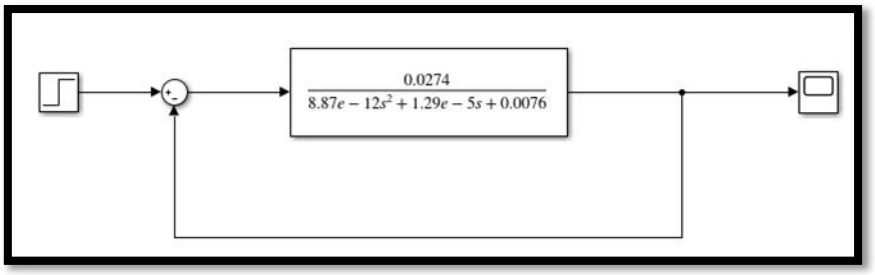


Figura 50: Función de transferencia del sensor IMU

Como se referenció anteriormente es fundamental que el robot entienda su entorno, por tal razón se implementó el sensor tipo IMU el cual tiene incorporado un giroscopio, acelerómetro y magnetómetro. Para implementar dicho dispositivo se debe encontrar la función de transferencia que lo rige, la Figura 50 muestra el lazo de control realizado a la función de transferencia resultante obtenida después de estabilizar el sensor y agitarlo bruscamente. Se analizo la gráfica obtenida, introduciendo los datos hallados en un simulink para su posterior implementación en la navegación propia del robot.

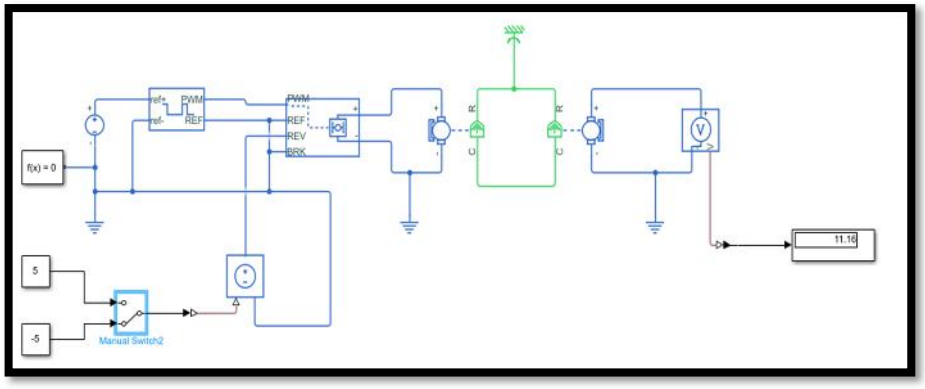


Figura 51: Simulación PWM en los motores.

El siguiente paso requiere el trabajo de los motores, poder enviar la información a realizar del software al hardware; en la figura 51 se ilustra la simulación realizada al entorno físico, enviando la información por medio del PWM (modulación por ancho de pulsos) dirigida a los motores, con el fin de poder realizar el movimiento o giro que el robot requiere para llegar al punto deseado.

Se debe tener en cuenta que, en el entorno de Simulink, su librería no permite manejar “puente h” para dos motores, por eso la simulación solo ilustra el movimiento aplicado a un solo motor, pero el principio del otro es exactamente el mismo.

Para finalizar, se diseñó un código que permite recrear mapas aleatorios para hacer la prueba de todo el algoritmo de trayectorias, teniendo en cuenta el seudocódigo RRT* analizado con anterioridad en este documento.

Cada punto de latitud y longitud fue caracterizado como un punto en coordenadas polares “X” “Y”. El principio de la simulación se deben introducir la ubicación actual del robot y el punto final a donde quiere que se desplace, por medio de las ecuaciones ya programadas, el sistema mide la distancia que hay entre el punto más cercano de sus cuatro lados y el punto a donde se quiere llegar, va seleccionando el más cercano. De esta manera el robot recrear el camino más cercano o posible para enviar estos resultados a los motores.

A continuación, se visualiza los resultados obtenidos en dos entornos diferentes:

En la figura 52 se ilustra la representación del algoritmo de trayectorias simulado en condiciones random para verificar y validar la entrada de datos y la resolución del algoritmo.

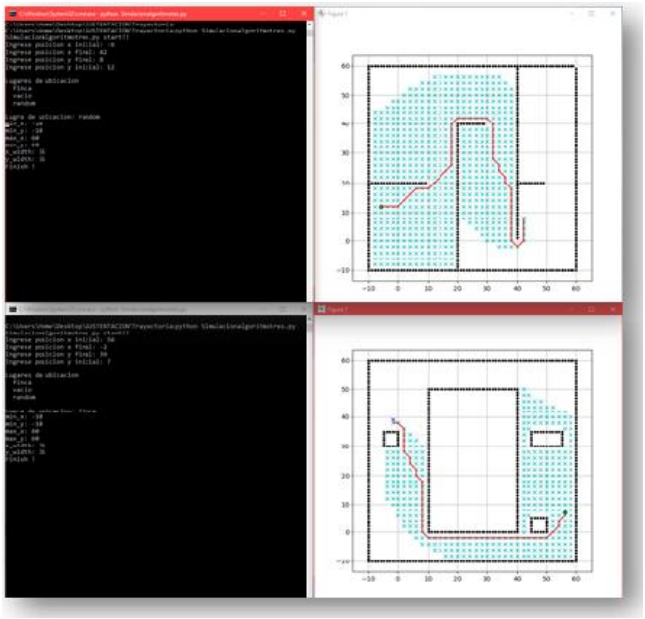
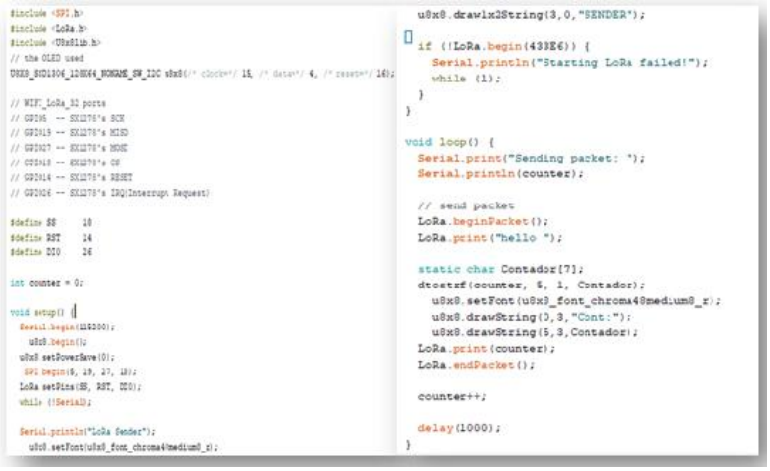


Figura 52: Simulación final del algoritmo de trayectorias

16.5 IMPLEMENTACIÓN DE SISTEMA DE COMUNICACIÓN E INTERFAZ GRAFICA



```
#include <SPI.h>
#include <LoRa.h>
#include <U8x8lib.h>
// the OLED used
U8X8_SSD1306_128X64_NONAME_SW_I2C u8x8(/* clock=*/ 15, /* data=*/ 4, /* reset=*/ 16);

// WIFI_LoRa_32 ports
// GPIO5  -- SX1278's SCK
// GPIO19 -- SX1278's MISO
// GPIO27 -- SX1278's MOSI
// GPIO18 -- SX1278's CS
// GPIO14 -- SX1278's RESET
// GPIO26 -- SX1278's IRQ(Interrupt Request)

#define SS      18
#define RST     14
#define DIO     26

int counter = 0;

void setup() {
  Serial.begin(115200);
    u8x8.begin();
  u8x8.setPowerSave(0);
   SPI.begin(5, 19, 27, 18);
  LoRa.setPins(SS, RST, DIO);
  while (!Serial);

  Serial.println("LoRa Sender");
    u8x8.setFont(u8x8_font_chroma48medium8_r);
    u8x8.drawlx2String(3,0,"SENDER");

  if (!LoRa.begin(433E6)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
}

void loop() {
  Serial.print("Sending packet: ");
  Serial.println(counter);

  // send packet
  LoRa.beginPacket();
  LoRa.print("hello ");

  static char Contador[7];
  dtostrf(counter, 6, 1, Contador);
    u8x8.setFont(u8x8_font_chroma48medium8_r);
    u8x8.drawString(0,3,"Cont:");
    u8x8.drawString(5,3,Contador);
  LoRa.print(counter);
  LoRa.endPacket();

  counter++;

  delay(1000);
}
```

Figura 53: Código emisor aplicado en robot a escala.

El sistema de comunicación es fundamental para tener un buen funcionamiento en el robot tipo oruga, se debe recordar que la comunicación entre interfaz y puesto de mando se va ser por medio de wifi, pero la comunicación o reenvío de datos al robot se va a hacer bajo los principio encontrados y analizados con anterioridad en el sistema tipo LORA.

La intercomunicación va de la mano con la interfaz gráfica ya que es donde el usuario inicia el funcionamiento e indica lo que se quiere realizar.

Para este caso, se realizó por medio del programa APPinventor, la programación y el diseño de la interfaz que permite manejar de forma manual y autónoma el robot.

En la siguiente imagen podrán visualizar la programación realizada y el prototipo de diseño que se planteó para hacer esta prueba.

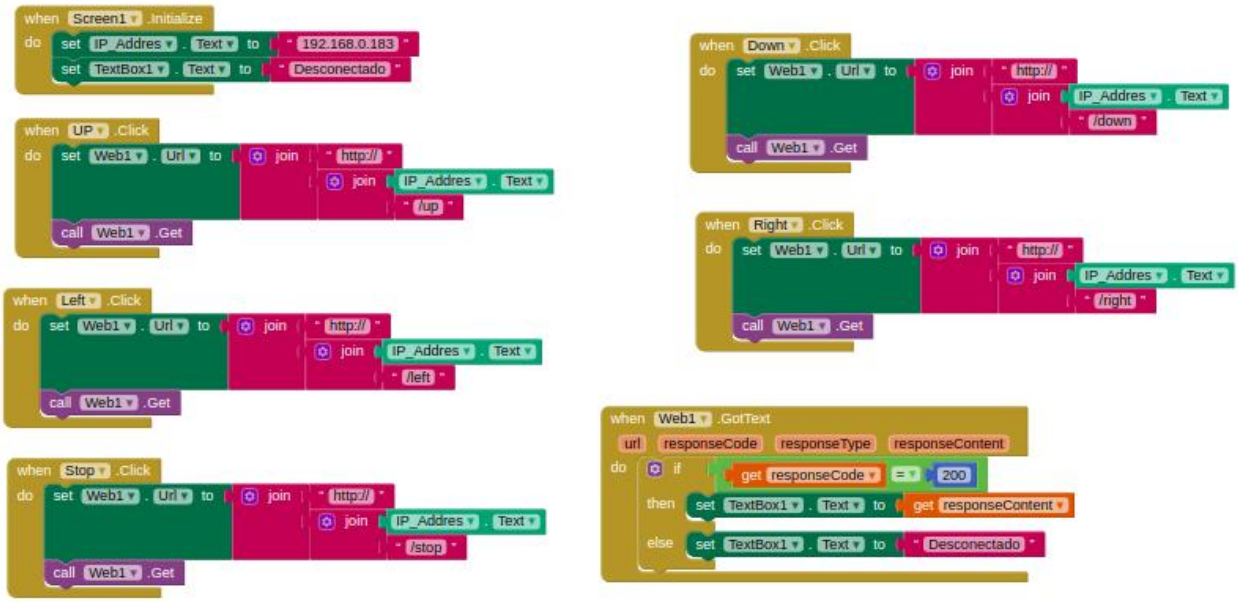


Figura 54: Programación y diseño de la interfaz gráfica para la intercomunicación.

17 VALIDACIÓN DE FUNCIONAMIENTO EN GENERAL

En este capítulo se dejará registrado todas las pruebas finales realizadas, con el fin de sustentar las técnicas aplicadas en cada objetivo en específico.

Teniendo en cuenta las recomendaciones dadas en la última sustentación se realizó más pruebas con el robot a escala con el fin de probar la geolocalización del robot, el envío de las variables de latitud y longitud en tiempo real; el seguimiento del robot al algoritmo de trayectorias realizado. A comparación de la anterior prueba, se cambió el microcontrolador por un Arduino DUE con el fin de aplicar el control al seguimiento de la trayectoria. Y el código empleado se puede ilustrar en parte a continuación, el resto se puede verificar en los entregables finales:

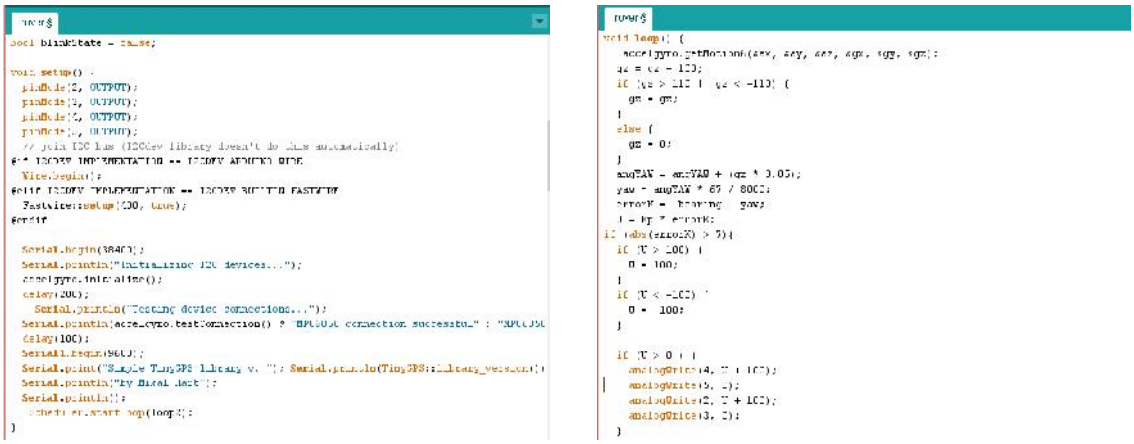


Figura 55: Código empleado en robot a escala

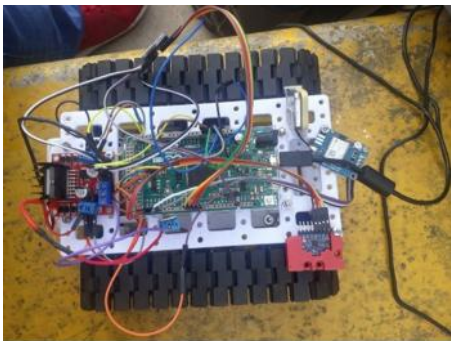


Figura 56: Robot a escala final implementado

Las pruebas realizadas tuvieron locación en el parque cristo rey, en cual se pudo comprobar el envío de datos en tiempo real de las variables de longitud y latitud, además se pudo verificar el control aplicado al seguimiento de la trayectoria por medio de puntos demarcados con anterioridad.

Figura 57: Locación de la prueba en robot a escala.

Los resultados encontrados son satisfactorios porque se pudo verificar que el robot es capaz de girar sobre su propio eje hasta encontrar el ángulo que lo ubique al frente del objetivo. También se pudo concluir que el GPS empleado demarca un margen de error de 1.5 a 2.6 metros de distancia, algo que nos permite corregir en el robot grande por medio de la fusión de otro gps que permite disminuir el margen de error.

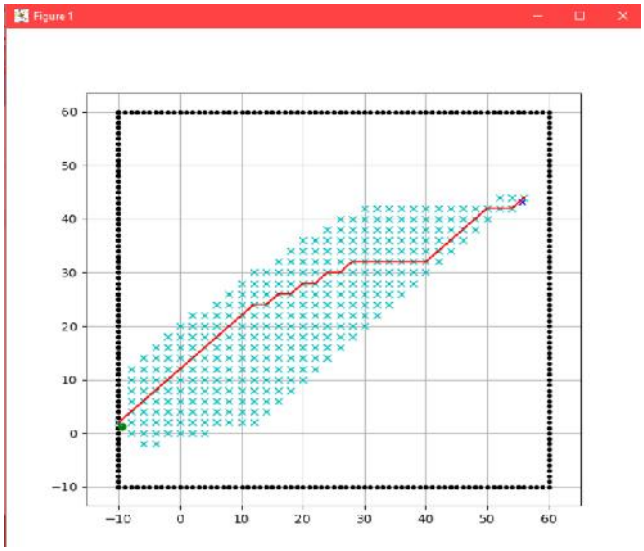


Figura 58: Prueba final robot a escala.

El comportamiento del robot generó gran satisfacción en la prueba porque demostró la capacidad de seguir una trayectoria y frenar en el punto señalado con anterioridad.

La figura 57 ilustra el resultado obtenido en la vida real, mostrando el camino generado por el robot y el seguimiento aplicado a una de las pruebas, en total se realizaron 20 pruebas ese día y sus resultados están en la tabla a continuación:

Tabla 13: Resultados de las pruebas en robot a escala.

Prueba #	T. En generar trayectorias	T. En seguir la trayectoria	Distancia error del objetivo
1	15s	32s	0,9m
2	9s	36s	2,1m
3	11s	28s	1,9m
4	14s	32s	2m
5	16s	27s	1,5m
6	12s	15s	1,7m
7	10s	30s	1,9m
8	7s	25s	1,3m
9	11s	26s	1,4m
10	9s	19s	1,7m
11	12s	21s	1,2m
12	14s	24s	1,4m
13	6s	22s	1,9m
14	8s	19s	1,1m
15	10s	35s	0,7m
16	11s	27s	0,6m
17	6s	23s	2,4m
18	7s	25s	1,4m
19	9s	24s	1,6m
20	8s	22s	1,4m

Figura 59: Error registrado en robot a escala

- **PROTOTIPO FINAL DEL COOFEE ROVER.**

Finalmente se implementó todo en el robot grande con el fin de validar su funcionamiento en las condiciones exigidas.

Por tal razón fue instrumentado con sensores y actuadores. 3 sensores ultrasónicos HC-SR04 los cuales permiten al robot entender la presencia de algún obstáculo durante su funcionamiento, así mismo para la geolocalización se hizo fusión de datos con dos sensores gps, un NEO7-n y un SIM808 los cuales tienen el objetivo de disminuir el margen de error presente en dichas tomas; para entender el norte, el ángulo de inclinación y los pasos a dar, se añadió un sensor IMU OBN055 (magnetómetro, giroscopio y acelerómetro) que cumplen su respectiva función. Por medio de una batería de 12V conectada a los dos motores (MY1018) son los actuadores encargados del movimiento en el robot, todo comandado por un microcontrolador Arduino DUE, el cual envía el pwm adecuado al puente h (POLOLU DUAL).

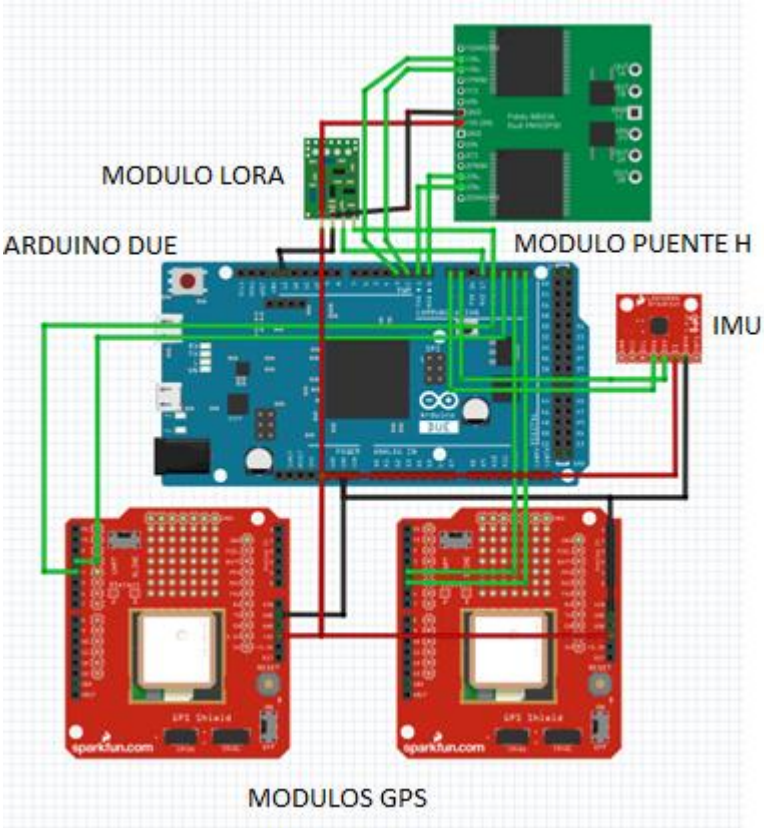


Figura 60: Circuito electrónico implementado

Se debe recordar que el Arduino cuenta con un módulo de comunicación tipo LORA el cual recibe y envía la información al puesto de control, el cual consiste de un dispositivo con unidad de procesamiento para el programa MATLAB, desde una interfaz propia el usuario podrá visualizar el mapa establecido con anterioridad, la ubicación en tiempo real del robot, el algoritmo de trayectorias e indicar el punto en donde se quiere el robot.

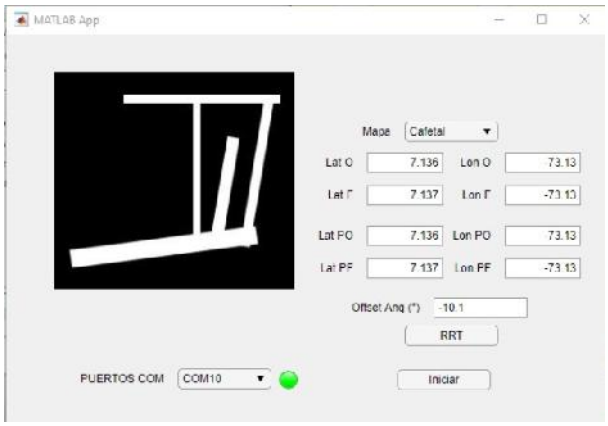


Figura 61: Interfaz gráfica final para el usuario.

Las figuras que se muestran en la interfaz de mando fueron diseñadas con anterioridad por medio de una aplicación de mapas (FieldAreaMeasure) que permiten registrar coordenadas en latitud, longitud y foto de vista superior, con lo cual se hace un análisis para convertir los caminos aptos de color blanco y los no aptos de color negro.

Figura 62: Ejemplo interfaz en FIELD AREA MEASURE

Anexo a este documento está el video (VALIDACIÓN ROVER) en el cual se visualiza el óptimo funcionamiento y validación de los objetivos del proyecto.
Además, en la siguiente Tabla se ilustra los resultados obtenidos en las pruebas realizadas.

Tabla 14: Pruebas Robot prototipo Rover.

Prueba #	T. En generar trayectorias (S)	Distancia de comunicación efectiva (m)	Interrupciones por IMU o Ultrasónico	Porcentaje de acierto robot vs maps	Distancia error del objetivo (m)
1	3	100	no	35%	0,1
2	2	200	no	37%	1,1
3	4	500	no	45%	0,9
4	2	500	no	50%	0,8
5	2	500	no	25%	1,2
6	3	500	no	26%	1,4
7	5	500	no	27%	1,3
8	4	500	no	29%	0,5
9	2	500	no	30%	0,1
10	3	500	no	24%	0,9
11	2	500	no	45%	0,7
12	1	500	no	49%	0,8
13	5	500	si	47%	0,4
14	4	500	no	45%	0,4
15	2	500	no	35%	0,4
16	3	500	no	35%	0,4
17	1	500	no	36%	0,4
18	2	750	no	38%	0,4
19	1	750	no	40%	0,7
20	2	750	no	22%	0,3
21	3	750	no	23%	0,5
22	2	750	no	30%	0,1
23	4	750	no	40%	0,8
24	2	1000	no	28%	0,2
25	1	1000	no	36%	0,3
26	3	1000	si	45%	0,4

27	5	1000	no	84%	0,6
28	4	1000	no	45%	0,7
29	2	1200	no	38%	0,4
30	1	1200	no	72%	0,4
31	4	1200	no	61%	0,6
32	2	1450	si	63%	0,1
33	2	1450	no	54%	0,3
34	2	1450	no	52%	0,7
35	3	1450	no	35%	0,2
36	3	1450	no	45%	0,4
37	5	1700	no	39%	0,6
38	4	1700	no	34%	0
39	1	1700	no	43%	0,4
40	2	2000	no	41%	0,2
41	3	2000	no	62%	0,2
42	4	2000	no	18%	0,4
43	5	2000	no	29%	0,3
44	2	2000	no	35%	0,5
45	1	2500	no	42%	0,7
46	4	2500	si	55%	1,3
47	3	2500	no	48%	1,2
48	3	2500	no	53%	1,4
49	1	2500	no	24%	0,9
50	2	2500	no	11%	0,4

18 CONCLUSIONES

La crisis mundial frente a la pandemia Covid-19 ha generado inconveniente a todos los seres humanos en la faz de la tierra, a tal punto que afecto directamente los avances de este documento. Iniciando proyecto se habían planteado metas que fueron difíciles de cumplir en su totalidad. El aislamiento preventivo obligatorio decretado a nivel nacional no permitió el trabajo en grupo, ni las compras, ni la implementación y mucho menos la validación de lo construido. Por tal motivo se modificó el cronograma para dejar las etapas que requieran construcción para el momento en el que sea seguro. De igual forma se trabajó y complemento lo faltante con investigación a fondo que permitió la selección de la ruta de procesos e implementos que se requieren para culminar el proyecto.

Al alcanzar a hacer la visita al campo caficultor permitió entender las necesidades reales de la industria, familiarizarnos con el proceso para conocer las condiciones climáticas, de vías y envío de información dentro del campo. Obteniendo una visión más aterrizada del proyecto.

Se puede estipular que por medio de la investigación a fondo realizada pertinentemente se lograron utilizar varios criterios de selección para las diferentes etapas que constituyen en su totalidad los objetivos del proyecto y así de esta manera comprar, implementar y desarrollar ya en la planta física del robot llevada a puesta punto en el campo caficultor, minimizar al máximo los errores y la equivoca metodología de la prueba y error.

Además, se puede concluir que el sistema de comunicación LORA es más efectivo que el bluetooh, wifi o de radio comunicación básicos, ya que puede enviar datos a largas distancias, sin la necesidad de tener línea de vista.

Se debe tener en cuenta que se puede mejorar el sistema de tracción del robot al

implementar materiales de mejor calidad, actualmente el UHM de los discos son buenos, pero presentan inconvenientes con la unión al rodamiento, lo que afecta su funcionamiento.

Las baterías o fuente de poder es un factor importante para estos proyectos, por tal razón se sugiere revisar o proyectar una manera que permita alargar el uso del robot, actualmente se está trabajando con una batería que tiene un funcionamiento entre 30 a 40 minutos.

El sistema inteligente requiere de un tiempo de inicio para poder calibrar sus sensores. Aproximadamente entre 90 y 120 segundos.
Para consumos de corriente alto y voltaje compartido se debe usar baterías tipo polímero de litio.

El experimento obtenido del sensor GPS al cambiar las antenas fue un éxito ya que se logró un margen de error de aproximadamente de 40cm con respecto con el punto final indicado.

Se opto por crear un documento (ANEXOS) en el cual se registra información importante que complementa la investigación realizada, no se dejó registro en este documento por el máximo de páginas permitidas.

Finalmente podemos concluir que el trabajo realizado en este proyecto nos permitió profundizar los conocimientos obtenidos en la vida universitaria y nos generó una experiencia importante para la realización de proyecto mecatrónicos, lo que nos permitirá ser mejores ingenieros.

19 MEJORAS A FUTURO

En este tipo de proyectos siempre existe la posibilidad de ir mejorando poco a poco el funcionamiento del mismo, de una formas más exacta o práctica, por tal razón se citarán algunas mejoras que se pueden implementar en trabajos a futuro.

- Las bandas u orugas del robot, las cuales permiten el movimiento del mismo, actualmente están hechas de bandas de llanta reciclada, las cuales fueron adaptadas, se solicita un cambio de bandas por unas más profesionales que permitan tener un mejor agarre y ampliar la vida útil del robot.

- Se debe estudiar el rediseño de los discos que hacen parte de la tracción del robot, ya que presenta algunos problemas cuando se acopla el rodamiento, ya que la fuerza a la que es sometido genera algunos desvíos.

- Buscar la manera de implementar una nueva fuente de poder o de alimentación la cual permita ampliar el tiempo de uso.

Además de mejores en este proyecto, también se puede ir añadiendo más proyectos de investigación, los cuales permiten optimizar las funciones actuales del robot. Tales cómo:

- Añadir brazo robótico con la capacidad de recolectar el café de manera autónoma.

- Implementar inteligencia artificial al robot, para mejorar su procesamiento, ampliar sus capacidades y disminuir los márgenes de error.

BIBLIOGRAFÍA

- El uso de robots en tareas agricolas [Articulo]. Antonio Barrientos y Jaime del Cerro (Interempresas). Obtenido de la red el día 04/08/19. http://www.interempresas.net/Horticola/Articulos/151745-El-uso-de-robots-en-tareas-agricolas.html
- Sistema de Visión Artificial para el Análisis de Imágenes de Cultivo basado en Texturas Orientadas [Articulo]. Escuela Politécnica Nacional. Obtenido de la red el día 04/08/19. https://revistapolitecnica.epn.edu.ec/ojs2/index.php/revista_politecnica2/article/viewFile/104/pdf
- Planeamiento de trayectorias de un robot movil [Trabajo de Grado]. Diego Alexander Tibaduiza (UIS). Obtenido de la red el día 05/08/19 http://tangara.uis.edu.co/biblioweb/tesis/2006/119245.pdf
- PLANIFICACIÓN DE TRAYECTORIAS DE ROBOTS MÓVILES DE DIFERENTES ARQUITECTURAS EN ENTORNOS DINÁMICOS [Trabajo de Grado]. JUAN DAVID PRECIADO AGUILAR. Obtenido en la red el día 05/08/19. https://repository.unimilitar.edu.co/bitstream/handle/10654/18006/PreciadoAguilarJuanDavid2018.pdf?sequence=1&isAllowed=y
- Coffee Co-Mission [WEB]. Cosecha y siembra de cafe. Obtenido de la red el día 05/08/19 https://coffeecomission.com/
- DISEÑO Y CONSTRUCCIÓN DE UN ROBOT MÓVIL COMO PLATAFORMA PARA EL APOYO A LAS LABORES EN LOS CULTIVOS DE CAFÉ [Trabajo de grado]. BRAJAN NICOLÁS RUIZ ROMERO. Obtenido virtualmente por director del proyecto.
- Qué es el café?. CafeteraCapsula 10. Obtenido de la red el día 5/04/20.https://cafeteracapsulas10.com/tipos-grano-cafe/

- El cafeto. Mundo cafeto. Obtenido de la red el día 5/04/20. https://mundocafeto.com/planta/el-cafeto/

- El uso de robots en tareas agrícolas. Interempresas. Obtenido de la red el día 05/04/20. https://www.interempresas.net/Horticola/Articulos/151745-El-uso-de-robots-en-tareas-agricolas.html

- Cultivemos café. Cenicafé. Obtenido de la red el día 05/04/20. https://www.cenicafe.org/es/index.php/cultivemos_cafe/index.php

- Robótica en agricultura. Huerta digital. Obtenido de la red el día 05/04/20 https://lahuertadigital.es/robotica-agricultura-perspectivas/

- Clasificación de robots. Google. Obtenido de la red el día 05/04/20 https://sites.google.com/site/elavancedelarobotica/clasificacion-de-los-robots/clasificacion-de-los-robots-su

- Clasificación de los robots. ESNECA. Obtenido de la red el día 05/04/20. https://www.esneca.com/blog/clasificacion-de-los-robots-segun-su-funcion/

- Fases del cultivo de café. Molido y servido. Obtenido de la red el día 05/04/20. https://www.molidoyservido.com/fases-del-cultivo-de-cafe/

- Guía tecnológica del cultivo. Federación de cafeteros. Obtenido de la red el día 05/05/20. https://federaciondecafeteros.org/static/files/8Capitulo6.pdf

- Siliciano, Bruno; Sciavicco, Lorenzo; Villani, Luigi; Oriolo, Giuseppe; Robotics Modelling, Planning and Control, Italy: Springer, 2009, 625p.

- Gregory Kahn; Pieter Abbeel; Sergey Levine; Berkeley AI Research (BAIR). University of California, BADGR: An Autonomous Self-Supervised Learning-Based Navigation System, 2020, 10p.

- Shaoshan Liu; Jean-Luc Gaudiot. Autonomous Vehicles Should Start Small, Go Slow. Obtenido de la red el día 05/04/20. https://spectrum.ieee.org/robotics/robotics-software/autonomous-vehicles-should-start-small-go-slow

- Florentino Prieto Rodríguez. Proyecto Fin de Grado Ingeniería Electrónica, Robótica y Mecatrónica. Métodos de Generación de Trayectorias. Universidad de Sevilla. Obtenido de la red el día 05/04/20 http://bibing.us.es/proyectos/abreproy/91252/fichero/TFG+%5BFlorentino+Prieto%5D.pdf

- Patrick McGarey. Visual Odometry (VO). Universidad de Toronto. Obtenido de la red el día 05/04/20. http://www.cs.toronto.edu/~urtasun/courses/CSC2541/03_odometry.pdf

- Diego Alexander Tibaduiza Burgos; Roberto Martínez Ángel; Jaime Barrero Pérez. ALGORITMOS DE PLANIFICACIÓN DE TRAYECTORIAS PARA UN ROBOT MÓVIL. Obtenido de la red el día 05/04/20. https://www.researchgate.net/publication/266393925_ALGORITMOS_DE_PLANIFICACION_DE_TRAYECTORIAS_PARA_UN_ROBOT_MOVIL

- Miguel Díez-Ochoa Díez. Localización de robots móviles en entornos tridimensionales mediante visión estéreo y CUDA. Universidad Carlos III de Madrid. Obtenido de la red el día 05/04/20. https://e-archivo.uc3m.es/bitstream/handle/10016/22565/PFC_miguel_diez_ochoa_diez_2014.pdf?sequence=1&isAllowed=y

- Manuel Alejandro Olivares Ávila; José Alberto Gallardo Arancibia. SISTEMA DE LOCALIZACIÓN AUTÓNOMA PARA ROBOTS MÓVILES BASADO EN FUSIÓN DE SENSORES PROPIOCEPTIVOS. Obtenido de la red el día 05/04/20 https://pdfs.semanticscholar.org/ef5f/9e23f0c9ddcb8684120452a5d83f206c07d6.pdf

- José María Armingol Moreno. LOCALIZACIÓN GEOMÉTRICA DE ROBOTS MÓVILES AUTÓNOMOS. TESIS DOCTORAL. Obtenido de la red el día 05/04/20 https://e-archivo.uc3m.es/bitstream/handle/10016/11629/tesis_armingol_1997.pdf?sequence=1&isAllowed=y

- Helbert Eduardo Espitia Cuchango; Jorge Iván Sofrony Esmeral. ALGORITMO PARA PLANEAR TRAYECTORIAS DE ROBOTS MÓVILES, EMPLEANDO CAMPOS POTENCIALES Y ENJAMBRES DE PARTÍCULAS ACTIVAS BROWNIANAS [PDF]. Universidad distrital Francisco Jose de Caldas.

- Siliciano, Bruno; Sciavicco, Lorenzo; Villani, Luigi; Oriolo, Giuseppe; Robotics Modelling, Planning and Control, Italy: Springer, 2009, 625p.

- Aníbal Ollero Baturone; ROBÓTICA Manipuladores y robots móviles, España: marcombo, 2001, 400p.

- Gregory Kahn, Pieter Abbeal, Sergey Levine Berkeley. BADGR: An Autonomous Self-Supervised Learning-Based Navigation System 13/02/2020

- Michael Andrés Moya Quimbita. Evaluación de pasarela LoRa/LoRaWAN en entornos urbanos. 01/04/2018

- Giovanni Rodrigo Bermúdez Bohórquez. MODELAMIENTO CINEMÁTICO Y ODOMÉTRICO DE ROBOTS MOVILES. ASPECTOS MATEMÁTICOS. 21/01/2013

- Rogerio Bonatti, Ratnesh Madaan, Vibhav Vineet, Sebastian Scherer, Ashish Kapoor. LEARNING VISUOMOTOR POLICIES FOR AERIAL NAVIGATION USING CROSS-MODAL REPRESENTATIONS. 8/03/2020

- Aguilera Hernandez Martha, Bautista Miguel, Iruegas Joaquín. DISEÑO Y CONTROL DE ROBOTS MÓVILES. 2007

- Solaque Guzmán Leonardo Enrique, Molina Villa Manuel Alejandro, Rodríguez Vásquez Edgar Leonardo. SEGUIMIENTO DE TRAYEECTORIAS CON UN ROBOT MÓVIL DE CONFIGURACIÓN DIFERENCIAL. 01/01/2014

- González Ramón, Rodríguez Francisco, Guzmán José Luis. ROBOTS MÓVILES CON ORUGAS HISTORIA, MODELADO, LOCALIZACIÓN Y CONTROL. 03/12/2015

- Gracia Calandín Luis Ignacio. MODELADO CINEMÁTICO Y CONTROL DE ROBOTS MÓVILES CON RUEDAS. 06/10/18

- Munguía-Alcalá Rodrigo Francisco. SLAM Con mediciones angulares. Obtenido de la red el día 15/08/20. https://www.elsevier.es/es-revista-ingenieria-investigacion-tecnologia-104-

articulo-slam-con-mediciones-angulares-metodo-S1405774313722418#:~:text=El%20SLAM%20(simultaneous%20localization%20and,al%20mismo%20tiempo%20para%20localizarse.

- Armingol Moreno José María. LOCALIZACIÓN GEOMÉTRICA DE ROBOTS MÓVILES AUTÓNOMOS. 2016

ANEXO A:

TIPOS DE LOCALIZACIÓN

SITEMA ODOMETRICO

La exactitud depende básicamente del diseño cinemático:

- Los vehículos con ruedas pequeñas son más propensos a la acumulación de error en la orientación, que los de ruedas grandes.
- La rueda ideal debería estar fabricada de aluminio y cubierta de una capa delgada de goma, para una mejor tracción.
- Configuración diferencial considera desplazamiento y velocidad como valores medios asociados a la rueda derecha e izquierda.

$$D = \frac{D_{dcha} + D_{izda}}{2} \quad , \quad V = \frac{V_{dcha} + V_{izda}}{2}$$

- La configuración en triciclo utiliza una única rueda directriz, y dos ruedas pasivas (se trata de una configuración básicamente utilizada e AGVs debido a su simplicidad), ello provoca un desplazamiento del centro de gravedad del vehículo en movimientos en planos inclinados, así como perdidas de tracción.
- La configuración synchro-drive proporciona mejores resultados que la diferencial y triciclo, sobre todo en suelos irregulares, debido a que todas las ruedas se encuentran acopladas, girando en la misma dirección y a la misma velocidad.

SISTEMA DE NAVEGACIÓN INERCIAL

Los sistemas de navegación inercial (INS) constituyen otra alternativa en la localización de robots móviles, aunque inicialmente fueron desarrollados para su empleo en aeronáutica. Básicamente su modo de funcionamiento está asociado a una medición continua de la aceleración en cada uno de los tres ejes direccionales. Una plataforma giroscópica estabilizada es utilizada para mantener la orientación de los tres acelerómetros empleados en el proceso. Los giróscopos son los encargados de proporcionar información respecto a la variación angular, mientras que los acelerómetros facilitan las variaciones de velocidad.

Para obtener la orientación y posición del vehículo será preciso integrar una y dos veces respectivamente la información relacionada con las variaciones angulares y de velocidad lineal, eso conduce a que cualquier pequeño error cometido en la

medida puede causar un crecimiento importante en el error de la posición y orientación del vehículo.

RELOCALIZACIÓN DE ROBOTS MÓVILES POR MEDIO DE MARCAS ACTIVAS.

La relocalización de robots móviles mediante marcas activas se basa en la medición de las direcciones de incidencia de tres o más marcas emisoras. En la mayoría de los casos estas marcas están constituidas por paneles luminosos, transmisores de radio frecuencia, ultrasonidos, etc., siendo conocida su ubicación en el entorno de trabajo.

Veremos entonces implícitamente el método de adaptación de este mecanismo de localización de robots móviles y como de cierta manera encontraremos ciertos rangos de errores y como emplean los instrumentos para encontrar resultados más precisos de posición, orientación y velocidad:

- Sistema de visión para relocalización de robots móviles, externo al robot (este sistema es capaz de cubrir un área de trabajo restringido). El robot lleva fijado un patrón de referencia constituido por puntos coplanarios, formado por diodos infrarrojos. El área de trabajo está cubierta por tres cámaras de vídeo de forma que, en todo momento, al menos una de ellas vea el patrón. En el bucle de predicción y verificación han desarrollado un método probabilístico que es aplicado para identificar y localizar los puntos del patrón. Los errores obtenidos durante el proceso de relocalización oscilan entre 2-10 mm.

- Desarrollo de una serie de paneles constituidos por grupos de LEDs, como marcas de referencia. El seguimiento de dichas marcas conocidas y colocadas sobre el entorno se realiza a través de un sistema de visión. Para llevar a cabo la integración de las medidas y la determinación de la estimación óptima de la posición del robot utilizan un filtro de Kalman extendido. El principal problema que presenta el sistema está asociado a la dificultad para identificar los puntos característicos de los paneles para el caso de presentarse ángulos de enfoque elevados, debido a que la intensidad observada de los LEDs, depende de la posición de la cámara.

ANEXO B

TARJETAS ENCONTRADAS EN EL MERCADO ACTUAL PARA COMUNICACIÓN LORA

Tabla 15: Tabla comparativa de las tarjetas LORA en el mercado actual

MODULO	VOLTAJE / A	REFERENCIA	INTERFAZ	ALCANCE U FRECUENCIA	ANTENA	CARACTERISTICAS	VIDEO Y ENLACE DE REFERENCIA
Tarjeta WiFi, Bluetooth GPS, LoRa LILYGO TTGO t-Beam	1.8 - 3.7 V	TTGO LORA 32 866/915Mhz	SPI	6.5 Km	Metal Wifi	Modulo de desarrollo IoT inalámbrico con comunicación WiFi, Bluetooth, GPS y LoRa. Con soporte para batería 18650 y antena.	https://www.youtube.com/watch?v=xrOo5SROzD8
Modulo GPS LoRaWAN con acelerómetro	3.3 V	LoraWan SX1278	Half Duplex SPI	3.5 Km	Sin antena integrada	Los transceptores SX1276 / 77/78/79 cuentan con el módem de largo alcance LoRa® que proporciona comunicación de espectro ensanchado de rango ultra largo e inmunidad de alta interferencia mientras minimiza el consumo de corriente. El uso de la técnica de modulación LoRa patentada de Semtech SX1276 / 77/78/79 puede lograr una sensibilidad de más de -148dBm utilizando un cristal de bajo costo y una lista de materiales. La alta sensibilidad combinada con el amplificador de potencia integrado de + 20dBm produce un presupuesto de enlace líder en la industria, lo que lo hace óptimo para cualquier aplicación que requiera rango o solidez.	https://www.youtube.com/watch?v=uN93JcDBLMs
Seeeduino LoRaWAN con GPS	3.3 V	LoRaWAN/RHF76-052	UART	Dual band, 434/470MHz and 868/915MHz	uFL antenna	Seeeduino LoRaWAN es compatible con LoRaWAN Clase A / C y admite una variedad de frecuencias de comunicación. Los 4 conectores Grove estándar integrados permiten que Seeeduino LoRaWan se conecte con cientos de sensores y actuadores Grove de Seeedstudio convenientemente, como resultado, los usuarios pueden centrarse más en la aplicación sin preocuparse por el problema de compatibilidad entre los diferentes módulos. Además, la placa ha incorporado un chip integrado de administración de batería de litio que permite cargar la placa mediante una interfaz USB. En el modo de bajo consumo, una batería de litio cargada por completo puede alimentar la placa durante varios meses	https://www.youtube.com/watch?v=4df5kaaKa6I&feature=youtu.be

ANEXO C

CARACTERIZACIÓN DE ALGORITMOS DE TRAYECTORIA

A ESTRELLA

Antes de explicar el algoritmo de A* conviene explicar un poco que es un algoritmo voraz. Así pues, un algoritmo voraz (también conocido como ávido, devorador o goloso) es aquel que, para resolver un determinado problema, sigue una heurística consistente en elegir la opción óptima en cada paso local con la esperanza de llegar a una solución general óptima. Este esquema algorítmico es el que menos dificultades plantea a la hora de diseñar y comprobar su funcionamiento. Normalmente se aplica a los problemas de optimización.

El problema de algunos algoritmos de búsqueda, como puede ser el algoritmo voraz, es que se guían en exclusiva por la función heurística, la cual puede no indicar el camino de coste más bajo, o por el coste real de desplazarse de un nodo a otro (como los algoritmos de escalada), pudiéndose dar el caso de que sea necesario realizar un movimiento de coste mayor para alcanzar la solución. Es por ello bastante intuitivo el hecho de que un buen algoritmo de búsqueda informada debería tener en cuenta ambos factores, el valor heurístico de los nodos y el coste real del recorrido.

Características

Como todo algoritmo de búsqueda en anchura, A* es un algoritmo completo: en caso de existir una solución, siempre dará con ella.

Si para todo nodo n del grafo se cumple g(n) = 0, nos encontramos ante una búsqueda voraz. Si para todo nodo del grafo se cumple h(n) = 0, A* pasa a ser una búsqueda de coste uniforme no informada.

Para garantizar la óptima calidad del algoritmo, la función h(n) debe ser admisible, o sea que no sobreestime el coste real de alcanzar el nodo objetivo.

De no cumplirse dicha condición, el algoritmo pasa a denominarse simplemente A, y a pesar de seguir siendo completo, no se asegura que el resultado obtenido sea el camino de coste mínimo. Asimismo, si garantizamos que h(n) es consistente (o monótona), es decir, que para cualquier nodo n y cualquiera de sus sucesores, el coste estimado de alcanzar el objetivo desde n no es mayor que el de alcanzar el sucesor más el coste de alcanzar el objetivo desde el sucesor.

Funcionamiento

Este algoritmo utiliza una función de evaluación f(n) = g(n) + h'(n), donde h'(n) representa el valor heurístico del nodo a evaluar desde el actual, n, hasta el final, y g(n), el costo real del camino recorrido para llegar a dicho nodo, n. A* mantiene dos estructuras de datos auxiliares, que podemos denominar abiertos, implementado como una cola de prioridad ordenada por el valor f(n) de cada nodo, y cerrados, donde se guarda la información de los nodos que ya han sido visitados. En cada paso del algoritmo, se expande el nodo que esté primero en abiertos, y en caso de que no sea un nodo objetivo, calcula la f(n) de todos sus hijos, los inserta en abiertos, y pasa el nodo evaluado a cerrados. El algoritmo es una combinación entre búsquedas del tipo primero en anchura con primero en profundidad: mientras que h'(n) tiende a primero en profundidad, g(n) tiende a primero en anchura. De este modo, se cambia de camino de búsqueda cada vez que existen nodos más prometedores.

1 Función heurística de A*

f(n) = g(n) + h(n): Coste real del plan (camino) de mínimo coste que pasa por n.

f*(n) = g(n) + h*(n): Estimación de f.

2 Estrategia de A* Entre las hojas del árbol de búsqueda, elegir el nodo de valor f* mínimo.

3 Interpretación fuerte de A* Una heurística suele facilitar la resolución de un problema, pero no garantiza que se resuelva. Una heurística es una regla de tres para un problema. Busqueda: Optimalidad o incluso completitud no garantizados.

4 Esquematización de A* Se basa en la búsqueda general. Almacenar el valor g de cada nodo expandido. Mantener la estructura abierta ordenada por valores crecientes de f*. Insertar nuevos nodos en la estructura abierta según sus valores de f*.

En el código empleado para este algoritmo hay una matriz de nombre MAP donde se encuentran los puntos “OPEN” (aquellos por donde el vehículo puede pasar) y “CLOSED” (aquellos por donde no puede pasar). El tamaño de MAP es del tamaño del mapa donde se realiza la simulación. Primero, antes de empezar el algoritmo, se colocan en la lista de puntos “CLOSED” los obstáculos de la zona y en la de puntos “OPEN” el punto de inicio del vehículo y calcula el coste de la función f(n) entre la posición inical y final. Quita de la lista “OPEN” el nodo con el coste más pequeño y se pone en la lista “CLOSED”, este es el nodo n (si dos nodos tienen exactamente el mismo coste se coge el último nodo en mirar de los dos). Si n es el nodo final entonces se usa el algoritmo para obtener la trayectoria, si no, continúa. Determina todos los nodos siguientes a n y calcula el coste de cada uno que no esté en la lista “CLOSED”. Asocia con cada nodo sucesivo que no esté en la lista “OPEN” o “CLOSED” el coste calculado y se pone en la lista de “OPEN”, colocando puntero en n (n es el nodo padre).

DESCOMPOSICIÓN EN CELDAS:

Con la descomposición de celdas, el planificador de rutas del robot tiene como objetivo encontrar un grafo de adyacencia que indique cuáles celdas libres de obstáculos comparten una frontera común, con el objetivo de identificar el espacio libre por donde se puede desplazar el robot. Los nodos del grafo corresponden a las celdas y las aristas conectan nodos de celdas adyacentes.

La descomposición de celdas en el planificador se realiza siguiendo los siguientes pasos:

1) Se construye el espacio de configuración asociado al espacio de trabajo del robot sobre el cual se identificarán las celdas.
2) Se construyen las celdas de acuerdo con las indicaciones del algoritmo.
3) Se construye el grafo de adyacencias.
4) Se determinan las celdas que contienen el comienzo y el final, respectivamente.
5) Se busca una ruta dentro del grafo de adyacencia.

Las celdas tienen una estructura simple, lo cual permite que sean cubiertas con movimientos sencillos del robot, tales como giros y avances en línea recta, ya que una vez el robot visita cada celda, el cubrimiento puede ser reducido a encontrar un camino a través del grafo de adyacencia.
Dentro de los algoritmos para la descomposición por celdas, uno de los más conocidos es la descomposición trapezoidal, que está relacionada con la representación poligonal del espacio, y aunque requiere mayor complejidad computacional que otras técnicas, presenta mayor exactitud y completitud. Esta técnica hace referencia, en particular, a las celdas de dos dimensiones que tienen forma de trapezoides.

En esta técnica se establece un sistema de coordenadas (x, y) para modelar el escenario en el cual se va a desplazar el robot. Así, el espacio libre estará delimitado por un polígono trapezoidal, y se asume que todos los obstáculos son polígonos.
Para realizar la descomposición en cada vértice v se dibujan dos segmentos, uno llamado extensión vertical superior, y el otro, extensión vertical inferior. El primero se refiere a incrementar la coordenada y, y el segundo, a decrementarla. Las extensiones verticales superior e inferior comienzan en cada vértice y terminan cuando interceptan el primer borde del polígono que se localice inmediatamente arriba o debajo de v, respectivamente; o bien, cuando llegan a la frontera del escenario. De esta manera los vértices tendrán una extensión vertical superior, inferior, ambas, o ninguna (en el caso de esquinas cóncavas).

Una vez que las celdas que contienen el punto inicial y el final estén definidas, se calcula el grafo de adyacencia para determinar la trayectoria, Sin embargo, el resultado de la búsqueda del grafo es una secuencia de nodos, y no una secuencia de puntos embebidos en el espacio libre. Es importante tener esto en cuenta, pues cada celda tendrá una posición y un área determinadas. Luego, se construye el camino conectando los puntos medios de las extensiones verticales a los centroides

de cada trapezoide. Así se obtiene un camino libre de colisiones, a través del espacio libre que se deriva del grafo de adyacencia.

Lo siguiente que hace el código es elegir desde que zona parte el vehículo y donde acaba para luego elegir la ruta a seguir. Para determinar la ruta se hace de la siguiente forma:

17 Se guarda la información de las celdas en las que se encuentran el inicio y el final.
18 Se determina las celdas adyacentes y uniendo los puntos medios de cada recta que forman las celdas y se determina la ruta más corta de entre las posibles.
19 Para obtener estas rutas se usa un grafo de conectividad como el siguiente:

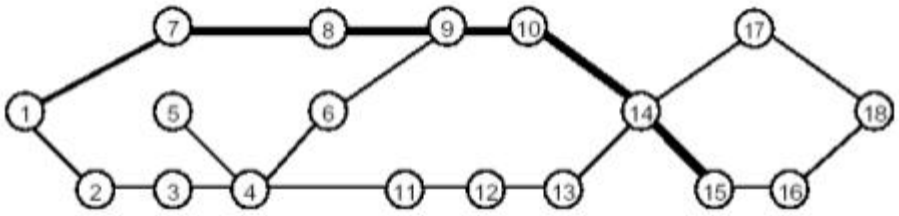


Figura 63: Grafo de conectividad

*RRT y RRT**

Un Rapidly exploring Random Tree (RTT) es un algoritmo diseñado para buscar eficientemente espacios no convexos de alta dimensión mediante la construcción aleatoria de un árbol de relleno de espacio. El árbol se construye de forma incremental a partir de muestras extraídas al azar del espacio de búsqueda y está intrínsicamente sesgada para crecer hacia grandes áreas no buscadas del problema.

Rapidly exploring Random Tree (RTT)

Un RRT crece en forma de árbol arraigado en la configuración inicial usando muestras aleatorias del espacio de búsqueda. A medida que se dibuja cada muestra, se intenta una conexión entre ella y el estado más cercano en el árbol. Si la conexión es factible (pasa completamente a través del espacio libre y obedece cualquier restricción), esto resulta en la adición del nuevo estado al árbol. Con un muestreo uniforme del espacio de búsqueda,

la probabilidad de expandir un estado existente es proporcional al tamaño de su región de Voronoi. Como las regiones Voronoi más grandes pertenecen a los estados en la frontera de la búsqueda, esto significa que el árbol se expande preferentemente hacia grandes áreas no buscadas.

Rapidly exploring Random Tree Star (RRT*)

RRT * es un algoritmo basado en RRT, que se centra en mejorar la calidad de las soluciones. Mientras RRT simplemente conecta un aleatorio con el nodo más cercano en el árbol, RRT * busca los nodos en una esfera de volumen con un radio específico para encontrar el nodo que hace el menor costo para llegar a la muestra aleatoria.

DIAGRAMA VORONOI

Los Diagramas de Voronoi son uno de los métodos de interpolación más simples, basados en la distancia euclidiana, especialmente apropiada cuando los datos son cualitativos. Se crean al unir los puntos entre sí, trazando las mediatrices de los segmentos de unión. Las intersecciones de estas mediatrices determinan una serie de polígonos en un espacio bidimensional alrededor de un conjunto de puntos de control, de manera que el perímetro de los polígonos generados sea equidistante a los puntos vecinos y designan su área de influencia.

La definición formal de los Diagramas de Voronoi es la siguiente:

Denotemos a la distancia euclidiana entre dos puntos p y q por ||p -q||. En el plano entonces se tiene:

||p- q|| = =√($p\ -q\ $)2+ ($p\ -q\ $)2

Sea P = p1, p2, pn el conjunto de n puntos distintos en el plano que son denominados como sitios. Se define el diagrama de Voronói de P como la subdivisión del plano en n regiones, una para cada pi Ɛ P, cumpliendo la propiedad de proximidad en la que un punto q pertenece a la región de un sitio pi si y sólo si ||q pi|| < ||q pj|| para cada pj Ɛ P, j≠i . Se denotará al diagrama de Voronoi de P

mediante Vor(P). Cada región que corresponde a un sitio pi se denotará como V(pi) y será llamada región de voronoi de pi.

Teorema 1. Sea P un conjunto de n sitios en el plano. Si todos los sitios son colineales, entonces Vor(P) consiste en n 1 líneas paralelas. De otra forma, Vor(p) es conexo y sus aristas podrían ser segmentos o semi líneas.

Teorema 2. Sea n ≥ 3, el número de vértices en el diagrama de Voronói de un conjunto de n sitios en el plano es a lo más 2n y el número de aristas es a lo más 3n.

Algoritmos para la construcción del diagrama de Voronói

Algoritmo por fuerza bruta:

Una primera aproximación para la construcción del diagrama de Voronoi consiste en explotar la geometría de cada región de Voronoi. Por cada sitio pi ε P se construirá su región de Voronoi mediante el cálculo explícito de los n 1 semiplanos originados debido a los bisectores trazados con respecto a los demás sitios. A continuación, se computará la intersección de estos n 1 semiplanos para dar origen a γ(pi).

Este algoritmo tiene muchas desventajas de entre las cuales se tienen las que a continuación se describen. En primera instancia, el cálculo explícito de los semi planos y su intersección puede provocar problemas de precisión en la computadora generados, evidentemente, por una versión incorrecta de Vor(P). El segundo inconveniente involucra que no se produce información inmediata que se pueda aprovechar acerca del vecindario de cada sitio. Finalmente, dado que se trata de un algoritmo ineficiente, no resulta extraño descubrir que su complejidad computacional sea alta. El algoritmo está en el orden de:

(n 2 log n)

Algoritmo divide y vencerás:

Este método primero divide un problema en dos subproblemas más pequeños de modo que cada subproblema sea idéntico al problema original, excepto porque su tamaño o dimensión es menor. Luego, ambos subproblemas se resuelven y las subsoluciones se fusionan en la solución final.

Obviamente, también estos dos subproblemas pueden resolverse aplicando la estrategia divide--y--vencerás, es decir, se pueden resolver de manera recurrente o recursiva.

La resolución de un problema mediante esta técnica consta fundamentalmente de los siguientes pasos:

1. En primer lugar ha de plantearse el problema de forma que pueda ser descompuesto en k subproblemas del mismo tipo, pero de menor tamaño. Es decir, si el tamaño de la entrada es n, hemos de conseguir dividir el problema en k subproblemas (donde $1 \leq k \leq n$), cada uno con una entrada de tamaño n/k y donde $0 \leq n/k < n$. A esta tarea se le conoce como división.
2. En segundo lugar han de resolverse independientemente todos los subproblemas, bien directamente si son elementales o bien de forma recursiva. El hecho de que el tamaño de los subproblemas sea estrictamente menor que el tamaño original del problema nos garantiza la convergencia hacia los casos elementales, también denominados casos base.
3. Por último, combinar las soluciones obtenidas en el paso anterior para construir la solución del problema original.

Dado el problema de construir el diagrama de Voronoi para el conjunto P de sitios, ahora se dividirá a éste último en dos subconjuntos P1 y P2, con aproximadamente el mismo tamaño, de los que se debe encontrar su diagrama de Voronoi independientemente. Finalmente, Vor(P1) y Vor(P2) deben ser unidos para poder obtener Vor(P).

Dada una petición P1 P2 de P, sea σ(P1,P2) el conjunto de aristas de Voronoi que son compartidas por pares de regiones de Voronoi γ(pi ε P1) y γ(pi ε P2). La colección de aristas σ(P1,P2) es el conjunto de aristas de una subgráfica de Vor(P) con las siguientes propiedades:

σ(P1,P2) consta de ciclos y cadenas de aristas disjuntas. Si una cadena tiene una sola arista, ésta es una línea recta; de otra forma sus dos aristas extremas son rayos semi--infinitos.

Si P1 y P2 son linealmente separados (si más de un punto pertenece a la línea de separación, todos estos puntos son asignados a un mismo conjunto de la partición), σ(P1,P2) consiste en una sola cadena monotónica.

Con el fin de separar a P en dos subconjuntos se le deberá ordenar con respecto a las abscisas y tomar la recta m que pase por la mediana, de tal forma que se tengan dos subconjuntos de aproximadamente el mismo tamaño. Adicionalmente, dada esta elección de recta de separación, se puede decir que σ parte al plano en una porción izquierda πL y una porción derecha πR. Con base en esto, se tiene la siguiente propiedad:

Si P1 y P2 son linealmente separados por una línea vertical con P1 a la izquierda y P2 a la derecha, entonces el diagrama de Voronoi Vor(P) es la unión de Vor(P1) ΩΩ πL y Vor(P2) ΩΩ πR.

El siguiente algoritmo establece la forma de calcular el diagrama de Voronoi mediante la técnica divide y vencerás:

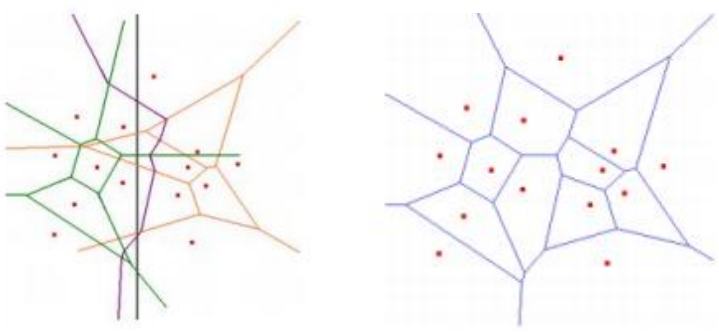


Figura 64: Pasos para el procesamiento del algoritmo Voronói.

ALGORITMO DIJKSTRA

El algoritmo de Dijkstra, también llamado algoritmo de caminos mínimos, es un algoritmo para la determinación del camino más corto dado un vértice origen al resto de los vértices

en un grafo con pesos en cada arista. Su nombre se refiere a Edsger Dijkstra, quién lo describió por primera vez en 1959. La idea subyacente en este algoritmo consiste en ir explorando todos los caminos más cortos

Funcionamiento del algoritmo Teniendo un grafo dirigido ponderado de N nodos no aislados, sea x el nodo inicial, un vector D de tamaño N guardará al final del algoritmo las distancias desde x al resto de los nodos.

1. Inicializar todas las distancias en D con un valor infinito relativo ya que son desconocidas al principio, exceptuando la de x que se debe colocar en 0 debido a que la distancia de x a x sería 0.
2. Sea a = x (tomamos a como nodo actual). Métodos de Generación de Trayectoria
3. Recorremos todos los nodos adyacentes de a, excepto los nodos marcados, llamaremos a estos nodos no marcados vi.
4. Para el nodo actual, calculamos la distancia tentativa desde dicho nodo a sus vecinos con la siguiente fórmula:

Dt(vi) = Da + d(a,vi).

Es decir, la distancia tentativa del nodo 'vi' es la distancia que actualmente tiene el nodo en el vector D más la distancia desde dicho el nodo 'a' (el actual) al nodo vi. Si la distancia tentativa es menor que la distancia almacenada en el vector, actualizamos el vector con esta distancia tentativa. Es decir: Si dt(vi) < Dvi Dvi = dt(vi)

5. Marcamos cómo completo el nodo a.
6. Tomamos como próximo nodo actual el de menor valor en D (puede hacerse almacenando los valores en una cola de prioridad) y volvemos al paso 3 mientras existan nodos no marcados.

Una vez terminado el algoritmo, D estará completamente lleno.

Matlab tiene una función que hace Voronoi pasándole una serie de puntos. Estos puntos se obtienen mediante la función "rand" de matlab también. Cuando se obtiene voronoi se pasa a unas funciones vx y vy. Luego se dibujan los obstáculos y se borra el interior de los obstáculos y se guardan en variable vx y vy nuevas. Una vez hecho esto se agregan el punto inicial y final y se van calculando la distancia entre los puntos. Una vez esto se utiliza el algoritmo de Dijkstra para encontrar el camino más corto mirando entre las posibilidades.

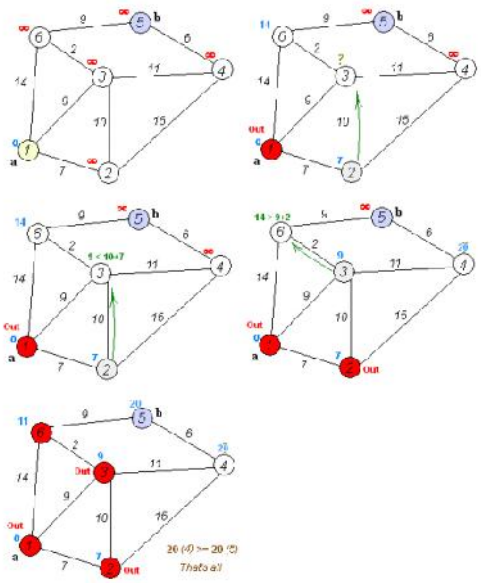


Figura 65: Ejecución del algoritmo de Dijkstra

ANEXO D

DISPOSITIVOS EMPLEADOS

Como es bien sabido los algoritmos deben ser alimentados de información, esta información se establece en tiempo real por medio de los sensores en campo. Para esta operación se empleará el sistema IMU.

IMU es un dispositivo electrónico que mide e informa acerca de la velocidad, orientación y fuerzas gravitacionales de un aparato, usando una combinación de acelerómetros y giróscopos.

Tabla 16: Tarjetas IMU

NOMBRE	VOLTAJE	INTERFAZ	CONECTORES	SENSORES	COMPATIBILIDAD	REFERENCIAS	PRECIO	ENLACE
DFRobot [illegible]	3.3 A 5V	[illegible]	Conectores de codificador de 2 vías usando D2, D3, D8, D9 [illegible]	Acelerómetro ADXL345 [illegible]	Arduino [illegible]	adicionalmente incluye un zócalo para módulo Xbee, haciéndolo adecuado para proyectos que necesiten comunicación inalámbrica [illegible]	[illegible]	[illegible]
MPU9250	3V/3.3V~5V DC	I2C y SPI	VIN: Alimentación voltaje 5V [illegible] GND: 0V Tierra [illegible]	giroscopio de 3 ejes, acelerómetro de 3 ejes, magnetómetro de 3 ejes. [illegible] Rango Acelerómetro: ±2g, ±4g, ±8g, ±16g [illegible] Rango de Presión: 300 a 1100 hPa [illegible] Resolución de temperatura: 0.01°C [illegible]	basado en tecnología BOSCH piezo-resistiva de alta precisión, linealidad y estabilidad [illegible] presión y temperatura	[illegible]	[illegible]	[illegible]
GY-801	3V~3.6V	Resistencias I2C [illegible]		L3G4200D + ADXL345 + HMC5883L + BMP180	protocolo I2C	Este IMU 10DOF se basa en los L3G4200D sensor, ADXL345, HMC5883L y BMP180. El sensor L3G4200D de tres ejes, mide la velocidad angular de rotación en el eje X, Y y Z, las señales [illegible] protocolo I2C. El ADXL345 es un pequeño y delgado, de baja potencia, acelerómetro de tres ejes con una alta resolución de medición (13 bits) [illegible]	$63.300	[illegible]
[illegible] MPU6050 [illegible]	3V/3.3V~5V	I2C		Giroscopio y acelerómetro	[illegible] microcontroladores AVR, ARM y Microchip.	Integra un DMP (Procesador digital de movimiento) capaz de realizar complejos algoritmos de captura de movimiento de 9 ejes. [illegible]	$10.300	[illegible]
HMC5883L	3V~5V	I2C		Compás/Brújula	[illegible]	Este módulo se caracteriza por su funcionamiento basado en un campo magnético de tres ejes [illegible]	$15.400	[illegible]
[illegible] TechIMU V4	3,3 ~ 4,0 VDC	Interfaz USB, interfaz para [illegible], interfaz CAN [illegible]		acelerómetro 3D, un giroscopio 3D y un magnetómetro 3D	DEPENDE	Cada [illegible] integra tres sensores micromecánicos diferentes, un acelerómetro 3D, un giroscopio 3D y un magnetómetro 3D. [illegible]	X	[illegible]

El sistema teleoperador está formado por cuatro bloques principales:

***SISTEMA DE ADQUISICIÓN DE DATOS*:** formado por elementos que proporcionan la posición.

Figura 66: Funduino joystick Shield, arduino explora, SparkFun Joystick Shield

***CANAL DE COMUNICACIÓN*:** formado por la tarjeta de transmisión y recepción de señales de radiofrecuencia 2.4GHz.

Figura 67: Tarjetas de radio frecuencia: nrf24l01, Hc05, Wifi, LORA.

***CONTROL*:** realizado por un microcontrolador que recibe señales de retroalimentación de sensores implementados en el robot.

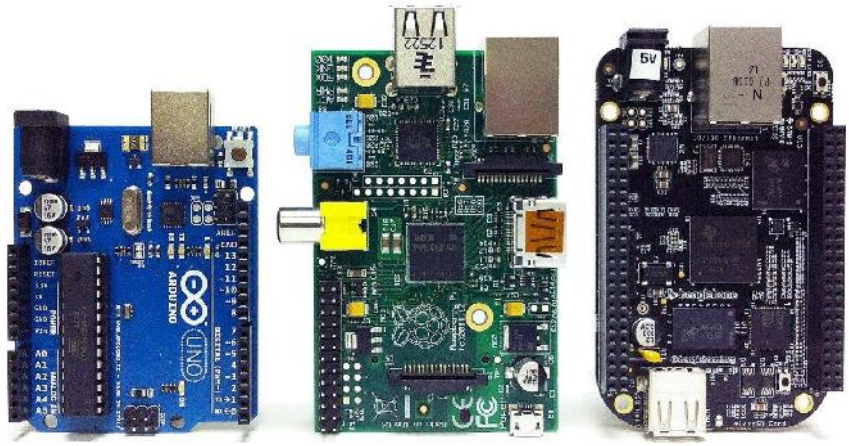


Figura 68: Arduino, Raspberry Pi, BeagleBoard.

***ETAPA DE POTENCIA*:** implementa un interruptor para que con las señales de control obtenidas del PIC se gire hacia un sentido u otro el motor indicado por el control.

ANEXO E

TIPOS DE CONFIGURACIONES DE CINEMÁTICA EN ROBÓTICA

Se entiende por cinemática el estudio del movimiento sin considerar las fuerzas que lo producen.

***CONFIGURACIÓN SÍNCRONA*:** La configuración síncrona o “synchro-drive” en la cual existen transmisiones que permiten orientar las tres ruedas simultáneamente con una velocidad angular y hacer que el vehículo se desplace con una velocidad lineal.

Modelo Jacobiano $\begin{bmatrix} x' \\ y' \\ \varphi' \end{bmatrix} = \begin{bmatrix} -\sin\varphi & 0 \\ \cos\varphi & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}$

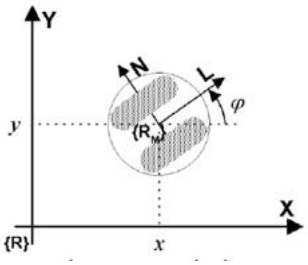


Figura 69: Ubicación cartesiana de la configuración síncrona.

CONFIGURACIÓN TRICICLO: La configuración en la cual la rueda delantera se utiliza tanto para la orientación como para la tracción. En este caso las variables de control suelen tomarse como el ángulo α de dirección de la rueda delantera (o su velocidad angular) y la velocidad de giro de la misma rueda ωt (o su velocidad lineal correspondiente $vt=ct$). Se supondrá que el punto de guía (x,y) está en el centro del eje trasero. Las velocidades lineal v y angular ω del vehículo que corresponden a las entradas vienen dadas por su modelo.

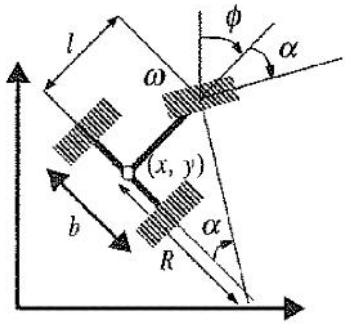


Figura 70: Ubicación cartesiana del modelo Triciclo

***CONFIGURACIÓN ACKERMAN*.**

Considérese ahora un vehículo de cuatro ruedas con sistema de locomoción configuración Ackerman. Se supone que el centro de guiado del vehículo (origen del sistema de referencia local {L}) está situado en la mitad del eje de las ruedas de tracción (ruedas traseras). En general no existen expresiones explicitas de la cinemática de la configuración Ackerman, debiendo aplicarse la integración numérica. Hay que tener en cuenta que la velocidad real de las cuatro ruedas es diferente y, por consiguiente, el cálculo de la velocidad del centro de guiado del vehículo a partir de la velocidad de una rueda da lugar a errores.

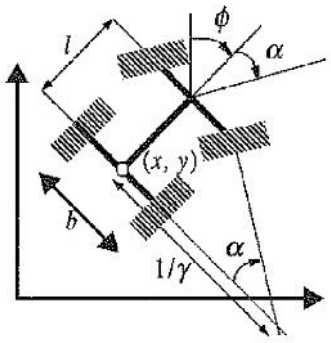


Figura 71: Ubicación cartesiana del modelo Ackerman

TERRAMECÁNICA:

La importancia del análisis terramecánico radica en que la cantidad de energía consumida durante la locomoción es función del empuje y la resistencia al avance producida por la interacción con el suelo, si el vehículo no genera suficiente empuje para superar esa resistencia, el vehículo queda atascado. A diferencia de los vehículos de carretera en los que la locomoción depende del estado de las ruedas, en los vehículos todo terreno también se debe considerar el estado del suelo.

Muchas de las técnicas para analizar la interacción rueda-suelo han sido desarrolladas y mejoradas a partir de las *teorías de Bekker*. Incluso él mismo usó múltiples técnicas de modelado para describir dicha interacción. Hoy en día, se emplean modelamientos basados en métodos empíricos, teóricos, numéricos y combinaciones entre ellos. Ningún método es ideal para todas las circunstancias y hay avances en cada una de tales técnicas.

***MÉTODOS EMPÍRICOS*:** Tal vez son los modelos terramecánicos más antiguos. Los métodos empíricos fueron creados en un principio al recolectar datos experimentales de un vehículo o una sola rueda para condiciones de operación específicas. A partir de los datos obtenidos se establecen correlaciones entre el vehículo, la rueda y las condiciones de operación.

Figura 72: Penetrómetro

MÉTODOS TEÓRICOS

Estos métodos se encuentran en el otro lado del espectro para el modelado terramecánico, se basan en *modelos elasto-plásticos* altamente idealizados, aunque existen modelos no lineales avanzados. En ellas se considera al suelo como una sustancia elástica que retorna a su estado inicial después de haber sido liberada de su carga, en donde el esfuerzo es proporcional a la deformación. Así, el esfuerzo

provocado en el suelo dependiendo del tipo de carga puede ser calculado al resolver ecuaciones diferenciales con condiciones de frontera conocidas. Cuando la carga supera el límite de elasticidad y hacen que el suelo falle, se aplica la teoría de plasticidad. El criterio de falla más usado es el Mohr-Coulomb, el cual relaciona el máximo esfuerzo cortante de un material al esfuerzo normal en la superficie y la cohesión del material con la resistencia al corte interno.

MÉTODOS SEMI-EMPÍRICOS

Los investigadores empezaron a combinar técnicas teóricas y empíricas para superar las deficiencias de los métodos teóricos. Uno de los métodos semi-empíricos es conocido como el *Método Bekker*, llamado así para diferenciarlo de otros métodos semi-empíricos.

El Modelo Bekker incorpora algunos elementos del método teórico como el criterio Mohr- Coulomb para el esfuerzo cortante, así como relaciones derivadas de la experimentación empírica. En el caso de la interacción rueda-suelo, el método de Bekker describe la distribución de esfuerzos cortantes y normales a lo largo de la interface rueda-suelo, dependiente de las propiedades de la rueda (radio) y del estado (tasa de deslizamiento). El método de Bekker se ha vuelto el estándar de la industria debido a su relativa facilidad de cómputo y su habilidad de ajustar los datos experimentales por medio del sintonizado de parámetros.

Los métodos empíricos y semi-empíricos, como el Método Bekker, están limitados en su aplicación debido a su dependencia en datos experimentales y sus simplificaciones y suposiciones en el modelado.

MÉTODOS NUMÉRICOS

En terramecánica se han también modelado a partir de los años 70’s el comportamiento rueda-suelo por medio del Método de Elementos Finitos, FEM por su sigla en inglés y el Método de Elementos Discretos (DEM) y cada vez son más populares debido al incremento de la potencia de computación. Los grandes efectos dinámicos resultado de las pequeñas interacciones, ya sea considerando el suelo

como un medio continuo en el caso del FEM o como pequeñas partículas individuales del DEM pueden modelarse gracias a los métodos numéricos. Las interacciones complejas que ocurren durante el contacto pueden ser tenidas en cuenta, incluyendo la dinámica del suelo de una rueda irregular atravesando terrenos difíciles, sin la necesidad de imponer restricciones o hacer suposiciones.

ANEXO F

PARTES DEL ROBOT:

SENSORES

Los sensores Permiten la adquisición de la información necesaria para el control de la plataforma móvil. En el estudio de los sensores debe involucrarse la medida de las magnitudes y su representación en forma compatible para su procesamiento.

En la toma de medidas siempre existe un cierto grado de incertidumbre. En principio, el incremento de la información hace posible la reducción de la incertidumbre. Para ello se trata de tomar más medidas o de emplear sensores redundantes.

Existen diferentes portadores de información basados en distintos principios físicos y químicos. Así, entre los principios y parámetros involucrados cabe mencionar:

- Mecánica: posición, velocidad, tamaño, fuerza.
- Termodinámica: temperatura, calor, entropía.
- Electricidad: voltaje, intensidad, resistencia, capacidad.
- Magnetismo: intensidad de campo, densidad de flujo.
- Química: concentración de un material, estructura cristalina.
- Radiación (ondas electromagnéticas) de todas las frecuencias, desde ondas de rayos, intensidad, frecuencia, polarización, fase.

Procesamiento y transmisión de la información

- Hidráulica mediante el empleo de componentes fluidos. En este caso el límite de la velocidad del sonido en un fluido, que es de aproximadamente 103 m/s
- Eléctrica y electrónica. En la actualidad se emplean circuitos electrónicos. El límite de velocidad viene dado por la movilidad de las cargas en un material semiconductor, que es de aproximadamente 105 m/s.
- Radiante empleado componentes ópticos. El límite es la velocidad de la luz en la guía: aproximadamente 108 m/s.

En la actualidad, se emplea casi con exclusividad el *procesamiento electrónico*. Para su empleo es necesario traducir las magnitudes a señales eléctricas.

En numerosas aplicaciones, además de las propias magnitudes, interesa conocer sus derivadas en el espacio o en el tiempo. Para ello puede procederse a la medida de la magnitud y al cálculo de la derivada mediante procesamiento.

Tipos de sensores en robótica

Si bien la variedad de sensores que pueden emplearse en un robot móvil está directamente relacionada con el campo de aplicación de éstos, hay una serie de sensores que se pueden considerar más acordes con las funciones del robot y a ellos nos vamos a referir.

Descriptores estáticos y dinámicos

Se puede definir el comportamiento del sensor mediante descriptores. Estos descriptores serán estáticos cuando definan el comportamiento del sensor en régimen permanente o serán dinámicos cuando caractericen la respuesta temporal del sensor antes determinados estímulos.

Los siguientes son descriptores estáticos,

- **Rango:** valores mínimos y máximos para las variables de entrada y salida.
- **Exactitud:** la desviación de la lectura de un sistema de medida respecto de una entrada conocida.
- **Repetitividad:** la capacidad de reproducir una lectura con una precisión dada.
- **Reproducibilidad:** es igual que la repetitividad, pero las lecturas se realizan bajo condiciones diferentes.
- **Resolución:** la cantidad más pequeña de incremento que puede ser determinada.
- **Error:** la diferencia entre el valor medido por el sensor y el valor real.
- **Linealidad:** cuando la respuesta del sensor es muy semejante a $m \cdot x + h$.

- **Sensibilidad:** es la razón de cambio de la salida frente a cambios en la entrada.
- **Excitación:** es la cantidad de corriente o tensión necesaria para el funcionamiento del sensor.
- **Estabilidad:** una medida de la posibilidad de un determinado sensor de mostrar una misma salida en un rango en el que la entrada permanece constante.
- **Ruido:** la señal que se acopla a la señal de salida que se podría considerar ideal, que hace que la señal esperada difiera de la real y ocasione problemas en los procesadores que la interpretan.

En la Figura 73 se presentan los descriptores dinámicos,

- **Tiempo de retardo, *td:*** el tiempo que tarda la señal de salida del sensor en alcanzar el 50% de su valor final.
- **Tiempo de crecimiento, *tr*:** el tiempo que tarda la señal de salida del sensor desde el valor original hasta alcanzar el valor final.
- **Tiempo de pico, *tp*:** el tiempo que tarda la señal de salida del sensor en alcanzar el pico máximo de su sobre oscilación.
- **Pico de sobre oscilación, *MP*:** expresa cuánto se eleva la evolución temporal de la señal de salida del sensor respecto al valor final.
- **Tiempo de establecimiento, *ts*:** el tiempo que tarda la señal de salida del sensor en quedar confinada a la banda de 5% alrededor del valor final.

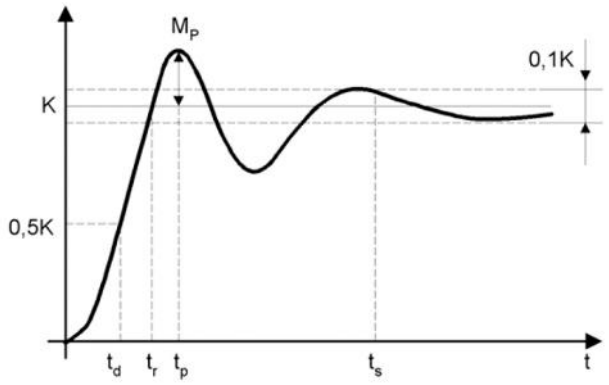


Figura 73: Descriptores dinámicos de un sensor

Aparte de los descriptores estáticos y dinámicos, es necesario considerar otros factores que pueden llegar a afectar la respuesta del sensor. Así, por ejemplo, el fabricante suele especificar condiciones ambientales tales como vibraciones, humedad, radiación o rango de temperatura en los cuales se garantiza el correcto funcionamiento.

Sensores en robots móviles

Para que un robot móvil pueda satisfactoriamente afrontar tareas como detectar obstáculos, monitorear la ejecución de la tarea, etc. se requiere que éste sea capaz de determinar su localización (posición y orientación) con respecto a un sistema de referencia absoluto. De forma general, determinar la posición de un robot móvil equivale a encontrar las componentes de translación (x, y, z) y de rotación (ϕx, ϕy, ϕz) del sistema de coordenadas solidario al robot {RM} (por tanto, móvil) con respecto a un sistema absoluto {R}.

- **Estimadores explícitos**

Los estimadores explícitos proporcionan la posición y orientación del robot directamente a partir de medidas sin que exista un procesamiento de información para interpretar el entorno.

- **Estimación basada en medidas internas**
- **Sistemas odométricos**

- **Sistemas de navegación inercial**
- **Estimación basada en estaciones de transmisión**

ACTUADORES

Los actuadores tienen por misión generar el movimiento de los elementos del robot según las órdenes dadas por la unidad de control. Los actuadores utilizados pueden emplear energía *neumática*, *hidráulica* o *eléctrica*. Cada uno de estos sistemas presenta características diferentes, siendo preciso evaluarlas a la hora de seleccionar el tipo de actuador más conveniente. Las características a considerar son entre otras:

- Potencia
- Controlabilidad
- Peso y volumen
- Precisión
- Velocidad mantenimiento
- Coste

- ***Actuadores eléctricos***

Las características de control, sencillez y presión de los accionamientos eléctricos han hecho que sean los más usados en los robots industriales actuales.

Motores de corriente continua (DC) Los motores de corriente continua son los más usados en la actualidad debido a su facilidad de control, son bastante usadas en sistemas de control en lazo cerrado, en particular para el control de velocidad y torque. Luego, existen dos modos de operación de un motor DC: a) modo por inducido y b) modo por excitación.

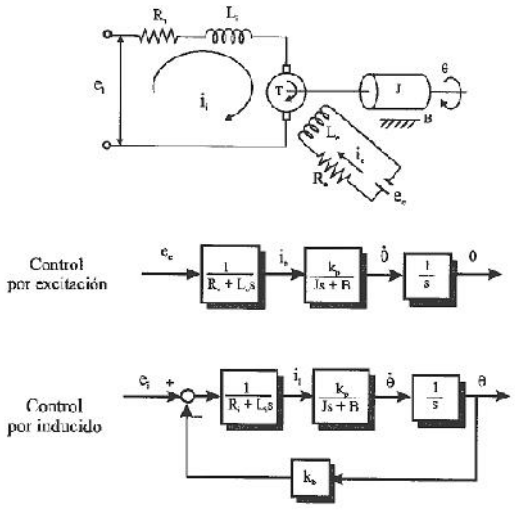


Figura 74: Modelo de motor DC

Tabla 17: Componentes del modelo en motor DC

ei=tensión inducido	ee=tensiona inductor	J=inercia
ri=resistencia inducido	re=resistencia inductor	B=coeficiente fricción viscosa
Li=inductancia inducido	Le=inductancia inductor	θ=desplazamiento angular
Ii=corriente inducido	ie=corriente inductor	T=torque

En el caso de control por inducido, la intensidad del inductor se mantiene constante, mientras que la tensión del inducido se utiliza para controlar la velocidad de giro, que dependiendo de su *polaridad determina la dirección de rotación del motor*. En los controlados por excitación se actúa, al contrario.

Puente en H: es un circuito electrónico que permite a un motor eléctrico DC girar en ambos sentidos, avance y retroceso. Los puentes H están disponibles como circuitos integrados, pero también pueden construirse a partir de componentes discretos dependiendo de la potencia del motor dc seleccionado.

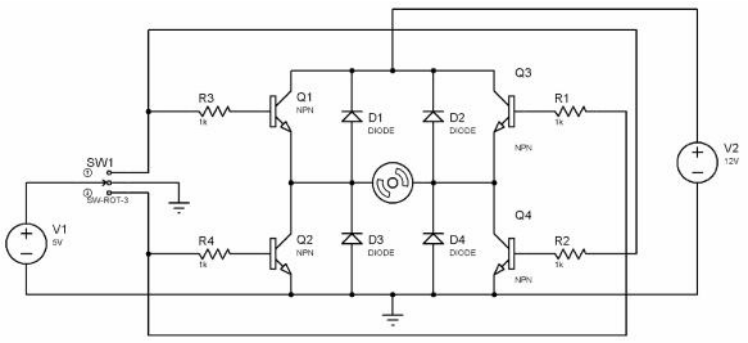


Figura 75: Diagrama del puente H

TELEROBÓTICA

La intervención del operador humano es imprescindible en un gran número de las aplicaciones de la robótica, especialmente en entornos no estructurados y dinámicos en los cuales los problemas de percepción y planificación automática son muy complejos. En muchos casos, el operador esta físicamente separado del robot, existiendo un sistema de telecomunicaciones entre los dispositivos que utilizan directamente el operador y el sistema de control local del robot.

- **Tele operación:** La extensión de capacidades sensoriales y destreza humanas a una localización remota.
- **Tele actuación:** Aspectos específicos de generación de órdenes a los actuadores.
- **Telesensorización:** Captación y visualización de información sensorial.

- ***Tele operación:*** Un sistema de teleoperación consta de los siguientes elementos:
  - **Operador o teleoperador:** es un ser humano que realiza a distancia el control de la operación. Su acción puede ir desde un control continuo hasta una intervención intermitente, con la que únicamente se ocupa de monitorizar y de indicar objetivos y planes cada cierto tiempo.

- **Dispositivo teleoperado:** podrá ser un manipulador, un robot, un vehículo o dispositivo similar. Es la máquina que trabaja en la zona remota y que está siendo controlada por el operador.
- **Interfaz:** conjunto de dispositivos que permiten la interacción del operador con el sistema de teleoperación. Se considera al manipulador maestro como parte del interfaz, así como a los monitores de vídeo, o cualquier otro dispositivo que permita al operador mandar información al sistema y recibir información del mismo.
- **Control y canales comunicación:** conjunto de dispositivos que modulan, transmiten y adaptan el conjunto de señales que se transmiten entre la zona remota y la local. Generalmente se contará con uno o varias unidades de procesamiento.
- **Sensores:** conjunto de dispositivos que recogen la información, tanto de la zona local como de la zona remota, para ser utilizada por el interfaz y el control.

ANEXO G

CODIGOS

LORA

```
#include <LoRa.h>
const int csPin = 18;          // LoRa radio chip select
const int resetPin = 14;       // LoRa radio reset
const int irqPin = 26;
String Datos;
void setup() {
  Serial.begin(9600);
  while (!Serial);

LoRa.setPins(csPin, resetPin, irqPin);// set CS, reset, IRQ pin
  if (!LoRa.begin(915E6)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
}

void loop() {
  // try to parse packet
  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    // read packet
    while (LoRa.available()) {
      Serial.print((char)LoRa.read());
    }
Serial.println();
  }
   while (Serial.available() > 0) {
```

```
  Datos = Serial.readStringUntil(10);
 }
  // send packet
  LoRa.beginPacket();
  LoRa.print(Datos);
  LoRa.endPacket();
  Serial.flush();

}
```

ROVER ARDUINO (GPS, IMU, SIST. INTELIGENTE)

```
#include "DualVNH5019MotorShield.h"
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#include <TinyGPS++.h>
#include <TinyGPS.h>
#include <Scheduler.h>
#include <DueTimer.h>

DualVNH5019MotorShield md;
TinyGPSPlus gps;
#define BNO055_SAMPLERATE_DELAY_MS (200)

// Check I2C device address and correct line below (by default address is 0x29 or
0x28)
//                                   id, address
Adafruit_BNO055 bno = Adafruit_BNO055(-1, 0x28);
const int Trigger = 51;   //Pin digital 2 para el Trigger del sensor
```

```
const int Echo = 50;   //Pin digital 3 para el Echo del sensor

float lastLat = 6.867517;
float lastLong = -73.052782;
float flat, flon;
float distance = 0;
float bearing = 0;
float YAW = 0;
float Kp = 20;
String yw ;
float errorK = 0;
int U = 0;
float Up = 0;
String coorde;
 String Rlat = "";
  String Rlon = "";
  int espa ;
void stopIfFault()
{
  if (md.getM2Fault())
  {
    Serial.println("M2 fault");
    while (1);
  }
  if (md.getM1Fault())
  {
    Serial.println("M1 fault");
    while (1);
  }
}
```

```
void setup() {
  // put your setup code here, to run once:
  Serial1.begin(9600);
  Serial.begin(115200);
  Serial3.begin(115200);

  pinMode(Trigger, OUTPUT); //pin como salida
  pinMode(Echo, INPUT);  //pin como entrada
  digitalWrite(Trigger, LOW);
 // Serial.println("Orientation Sensor Raw Data Test"); Serial.println("");
  md.init();
  /* Initialise the sensor */
  if (!bno.begin())
  {
    /* There was a problem detecting the BNO055 ... check your connections */
    Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
    while (1);
  }
  bno.setExtCrystalUse(true);

}

void loop() {
  long t; //timepo que demora en llegar el eco
  long d; //distancia en centimetros

  digitalWrite(Trigger, HIGH);
  delayMicroseconds(10);          //Enviamos un pulso de 10us
  digitalWrite(Trigger, LOW);
```

```
  t = pulseIn(Echo, HIGH); //obtenemos el ancho del pulso
  d = t/59;

    sensors_event_t event;
  bno.getEvent(&event);
  YAW = (float)event.orientation.x;
if (YAW > 180) {
        YAW = YAW - 360;
      }
  /* Display the floating point data */
  // put your main code here, to run repeatedly:
  errorK = bearing - YAW;
  U = Kp * errorK;
  if (abs(errorK) > 2 && d>50) {
    if (U > 1000) {
      U = 1000;
    }
    if (U < -1000) {
      U = -1000;
    }

    if (U > 0 ) {

      md.setM2Speed(-U+40);
      md.setM1Speed(0);

      stopIfFault();
    }
    else {
```

```
    md.setM1Speed(-U+40);
    md.setM2Speed(0);

    stopIfFault();

  }
  Up = U;
  delay(2);
}
else {
  if(distance>3&& d>50){
  md.setM1Speed(300);
  md.setM2Speed(-300);
  }
  else{
    if(distance>3){
  md.setM1Speed(0);
  md.setM2Speed(0);

  }
  }
}
 while (Serial3.available() > 0) {
coorde = Serial3.readStringUntil(10);
espa = coorde.indexOf(' ');
Rlat= coorde.substring(0,espa);
Rlon = coorde.substring(espa,coorde.length());
lastLat = Rlat.toDouble();
lastLong = Rlon.toDouble();
}
```

```
  while (Serial1.available() > 0) {
    if (gps.encode(Serial1.read())) {
      flat = gps.location.lat();
      flon = gps.location.lng();

      distance = TinyGPS::distance_between(flat, flon, lastLat, lastLong);
      bearing = TinyGPS::course_to(flat, flon, lastLat, lastLong);
      if (bearing > 180) {
        bearing = bearing - 360;
      }
      Serial.print(flat,6);
      Serial.print(" ");
      Serial.print(flon,6);
      Serial.print(" ");
      Serial.print(distance,6);
      Serial.print(" ");
      Serial.println(errorK);
      Serial3.print(flat,6);
      Serial3.print(" ");
      Serial3.print(flon,6);
      Serial3.print(" ");
      Serial3.println(YAW);
    }
  }

  delay(BNO055_SAMPLERATE_DELAY_MS);
}
```

ALGORITMO DE TRAYECTORIAS (RRT)

```
I = imread('fin.jpg'); % Importamos imagen de la trayectoria
I = imbinarize(I(:,:,1),0); % Binariza
imshow(I); % Mostrar la imagen
occGrid = binaryOccupancyMap(not(I)); % Mapa de Ocupación
inflate(occGrid,1); show(occGrid); % Escala de pixeles con respecto a unidad de medida

start = [1.0, 0.0, -pi]; % Punto de inicio
goal = [300, 250, 0]; % Punto de llegada

% Show the start and goal positions of the robot
hold on
plot(start(1), start(2), 'ro') % Dibujar un circulo en azul
plot(goal(1), goal(2), 'mo') % Dibujar un circulo en rojo

% Show the start and goal headings
r = 0.5; % Radio de giro grafico
plot([start(1), start(1) + r*cos(start(3))], [start(2), start(2) + r*sin(start(3))], 'r-' ) % Trazar linea de direccion inicial
plot([goal(1), goal(1) + r*cos(goal(3))], [goal(2), goal(2) + r*sin(goal(3))], 'm-' ) % Trazar linea de direccion final
hold off % Quitar retencion de imagen

bounds = [occGrid.XWorldLimits; occGrid.YWorldLimits; [-pi pi]]; % Definiendo los limites del sistema

ss = stateSpaceDubins(bounds); % Inicializar el sistema a optimizar
ss.MinTurningRadius = 1; % Radio de minimo giro
```

```
stateValidator = validatorOccupancyMap(ss);
stateValidator.Map = occGrid;
stateValidator.ValidationDistance = 0.05;

planner = plannerRRT(ss, stateValidator);
planner.MaxConnectionDistance = 50.0;
planner.MaxIterations = 30000;

rng(0,'twister')

[pthObj, solnInfo] = plan(planner, start, goal);

show(occGrid)
hold on

% Search tree
plot(solnInfo.TreeData(:,1), solnInfo.TreeData(:,2), '.-');

% Interpolate and plot path

plot(pthObj.States(:,1), pthObj.States(:,2), 'r-', 'LineWidth', 2)

% Show the start and goal in the grid map
plot(start(1), start(2), 'ro')
plot(goal(1), goal(2), 'mo')
hold off
X = pthObj.States(:,1)*0.01;
Y =  pthObj.States(:,2)*0.01;
```

EJEMPLO Y TRATAMIENTO DE COOREDENADAS CON HAVERSINE

```
function [d1km d2km]=lldistkm(latlon1,latlon2)
% format: [d1km d2km]=lldistkm(latlon1,latlon2)
% Distance:
% d1km: distance in km based on Haversine formula
% (Haversine: http://en.wikipedia.org/wiki/Haversine_formula)
% d2km: distance in km based on Pythagoras' theorem
% (see: http://en.wikipedia.org/wiki/Pythagorean_theorem)
% After:
% http://www.movable-type.co.uk/scripts/latlong.html
%
% --Inputs:
%   latlon1: latlon of origin point [lat lon]
%   latlon2: latlon of destination point [lat lon]
%
% --Outputs:
%   d1km: distance calculated by Haversine formula
%   d2km: distance calculated based on Pythagoran theorem
%
% --Example 1, short distance:
%   latlon1=[-43 172];
%   latlon2=[-44  171];
%   [d1km d2km]=distance(latlon1,latlon2)
%   d1km =
%          137.365669065197 (km)
%   d2km =
%          137.368179013869 (km)
%   %d1km approximately equal to d2km
%
% --Example 2, longer distance:
```

```
%   latlon1=[-43 172];
%   latlon2=[20  -108];
%   [d1km d2km]=distance(latlon1,latlon2)
%   d1km =
%          10734.8931427602 (km)
%   d2km =
%          31303.4535270825 (km)
%   d1km is significantly different from d2km (d2km is not able to work
%   for longer distances).
%
% First version: 15 Jan 2012
% Updated: 17 June 2012
%---------------------------------------------------------------------------

radius=6371;
lat1=latlon1(1)*pi/180;
lat2=latlon2(1)*pi/180;
lon1=latlon1(2)*pi/180;
lon2=latlon2(2)*pi/180;
deltaLat=lat2-lat1;
deltaLon=lon2-lon1;
a=sin((deltaLat)/2)^2 + cos(lat1)*cos(lat2) * sin(deltaLon/2)^2;
c=2*atan2(sqrt(a),sqrt(1-a));
d1km=radius*c;    %Haversine distance
x=deltaLon*cos((lat1+lat2)/2);
y=deltaLat;
d2km=radius*sqrt(x*x + y*y); %Pythagoran distance

end
```

EJEMPLOS DE MEMORIA Y ECUCIONES DEL MODELO CINEMATICO.

```
function  [x,y,utmzone] = deg2utm(Lat,Lon)
% ------------------------------------------------------------------------
% [x,y,utmzone] = deg2utm(Lat,Lon)
%
% Description: Function to convert lat/lon vectors into UTM coordinates (WGS84).
% Some code has been extracted from UTM.m function by Luis Ortega y Nicolas
Silva.
%
% Inputs:
%    Lat: Latitude vector.   Degrees.  +ddd.ddddd  WGS84
%    Lon: Longitude vector.  Degrees.  +ddd.ddddd  WGS84
%
% Outputs:
%    x, y , utmzone.   See example
%
% Example 1:
%    Lat=[40.3154333; 46.283900; 37.577833; 28.645650; 38.855550; 25.061783];
%    Lon=[-3.4857166; 7.8012333; -119.95525; -17.759533; -94.7990166;
121.640266];
%    [x,y,utmzone] = deg2utm(Lat,Lon);
%    fprintf('%7.0f ',x)
%       458731  407653  239027  230253  343898  362850
%    fprintf('%7.0f ',y)
%      4462881 5126290 4163083 3171843 4302285 2772478
%    utmzone =
%       30 T
%       32 T
%       11 S
%       28 R
```

```
%       15 S
%       51 R
%
% Example 2: If you have Lat/Lon coordinates in Degrees, Minutes and Seconds
%    LatDMS=[40 18 55.56; 46 17 2.04];
%    LonDMS=[-3 29  8.58;  7 48 4.44];
%    Lat=dms2deg(mat2dms(LatDMS)); %convert into degrees
%    Lon=dms2deg(mat2dms(LonDMS)); %convert into degrees
%    [x,y,utmzone] = deg2utm(Lat,Lon)
%
% Author:
%   Luis Hermes Ortega Quiñonez
%
%   Unab, Colombia
% Version: Dic/20
%------------------------------------------------------------------------

% Argument checking
%
error(nargchk(2, 2, nargin));  %2 arguments required
n1=length(Lat);
n2=length(Lon);
if (n1~=n2)
   error('Lat and Lon vectors should have the same length');
end


% Memory pre-allocation
%
x=zeros(n1,1);
```

```
y=zeros(n1,1);
utmzone(n1,:)='60 X';

% Main Loop
%
for i=1:n1
   la=Lat(i);
   lo=Lon(i);

   sa = 6378137.000000 ; sb = 6356752.314245;

   %e = ( ( ( sa ^ 2 ) - ( sb ^ 2 ) ) ^ 0.5 ) / sa;
   e2 = ( ( ( sa ^ 2 ) - ( sb ^ 2 ) ) ^ 0.5 ) / sb;
   e2cuadrada = e2 ^ 2;
   c = ( sa ^ 2 ) / sb;
   %alpha = ( sa - sb ) / sa;            %f
   %ablandamiento = 1 / alpha;   % 1/f

   lat = la * ( pi / 180 );
   lon = lo * ( pi / 180 );

   Huso = fix( ( lo / 6 ) + 31);
   S = ( ( Huso * 6 ) - 183 );
   deltaS = lon - ( S * ( pi / 180 ) );

   if (la<-72), Letra='C';
   elseif (la<-64), Letra='D';
   elseif (la<-56), Letra='E';
   elseif (la<-48), Letra='F';
   elseif (la<-40), Letra='G';
```

```
elseif (la<-32), Letra='H';
elseif (la<-24), Letra='J';
elseif (la<-16), Letra='K';
elseif (la<-8), Letra='L';
elseif (la<0), Letra='M';
elseif (la<8), Letra='N';
elseif (la<16), Letra='P';
elseif (la<24), Letra='Q';
elseif (la<32), Letra='R';
elseif (la<40), Letra='S';
elseif (la<48), Letra='T';
elseif (la<56), Letra='U';
elseif (la<64), Letra='V';
elseif (la<72), Letra='W';
else Letra='X';
end

a = cos(lat) * sin(deltaS);
epsilon = 0.5 * log( ( 1 +  a) / ( 1 - a ) );
nu = atan( tan(lat) / cos(deltaS) ) - lat;
v = ( c / ( ( 1 + ( e2cuadrada * ( cos(lat) ) ^ 2 ) ) ) ^ 0.5 ) * 0.9996;
ta = ( e2cuadrada / 2 ) * epsilon ^ 2 * ( cos(lat) ) ^ 2;
a1 = sin( 2 * lat );
a2 = a1 * ( cos(lat) ) ^ 2;
j2 = lat + ( a1 / 2 );
j4 = ( ( 3 * j2 ) + a2 ) / 4;
j6 = ( ( 5 * j4 ) + ( a2 * ( cos(lat) ) ^ 2) ) / 3;
alfa = ( 3 / 4 ) * e2cuadrada;
beta = ( 5 / 3 ) * alfa ^ 2;
gama = ( 35 / 27 ) * alfa ^ 3;
```

```
  Bm = 0.9996 * c * ( lat - alfa * j2 + beta * j4 - gama * j6 );
  xx = epsilon * v * ( 1 + ( ta / 3 ) ) + 500000;
  yy = nu * v * ( 1 + ta ) + Bm;

  if (yy<0)
     yy=9999999+yy;
  end

  x(i)=xx;
  y(i)=yy;
  utmzone(i,:)=sprintf('%02d %c',Huso,Letra);
end
```

INTERFAZ GRAFICA PARTE 1

```
device = serialport("COM16",9600);
name = 'openstreetmap';
url = 'https://a.tile.openstreetmap.org/${z}/${x}/${y}.png';
copyright = char(uint8(169));
attribution = copyright + "OpenStreetMap contributors";
addCustomBasemap(name,url,'Attribution',attribution)

for()
   datos = readline(device);
   newStr = split(datos," ")
  2+2
   pause(0.5)
   zoomLevel = 15;
   if(i<1)
player = geoplayer(str2double(newStr(1)),str2double(newStr(2)),zoomLevel);
   end
```

```
    plotPosition(player,str2double(newStr(1)),str2double(newStr(2)));
 i = i+1;
end

clear device;
i = 0;
```

INTERFAZ GRAFICA PARTE 2

```
function varargout = Interfaz(varargin)

% Last Modified by GUIDE v2.5 10-Jan-2021 23:28:58

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Interfaz_OpeningFcn, ...
                   'gui_OutputFcn',  @Interfaz_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before Interfaz is made visible.
function Interfaz_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Interfaz (see VARARGIN)

% Choose default command line output for Interfaz
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Interfaz wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Interfaz_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

```
% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1 contents as
cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu1


% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2 contents as
cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu2


% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
```

```
%       str2double(get(hObject,'String')) returns contents of edit3 as a double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%       str2double(get(hObject,'String')) returns contents of edit4 as a double


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```