SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Querétaro

INSTITUTO TECNOLÓGICO DE QUERÉTARO

DISEÑO E IMPLEMENTACIÓN DE UNA INTERFAZ GRÁFICA PARA GENERACIÓN DE TRAYECTORIAS DE UN ROBOT MÓVIL DE CONFIGURACIÓN DIFERENCIAL

TESIS

Que para obtener el Grado de

MAESTRA EN INGENIERÍA

presenta

ALICIA GUADALUPE MIJANGOS CONTRERAS

Dirigido por:

M.C. Omar Alejandro Cervantes Gloria

Dr. Raúl Ramírez López

M.C. Daniel Hernández Arriaga

Enero 2020

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Querétaro
División de Posgrado e Investigación

"2019, Año del Caudillo del Sur, Emiliano Zapata"

Querétaro, Qro. 10 de Enero del 2020
DIV. DE EST. POSG. E INV.
DEPI-004/2020

MIJANGOS CONTRERAS ALICIA GUADALUPE
ESTUDIANTE
MAESTRÍA EN INGENIERÍA
PRESENTE

De acuerdo con el Reglamento para Exámenes Profesionales de la Dirección General de Educación Superior Tecnológica, se le autoriza la impresión de la Tesis, para obtener el Grado de MAESTRIA EN INGENIERÍA, titulada:

"DISEÑO E IMPLEMENTACIÓN DE INTERFAZ GRÁFICA PARA GENERACIÓN DE TRAYECTORIAS DE UN ROBOT MÓVIL DE CONFIGURACIÓN DIFERENCIAL"

Para el correspondiente Examen de Grado.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
"la tierra será como sean los hombres"

SEP

MA. DEL CONSUELO ALCÁNTARA TÉLLEZ
JEFA DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

C.C.P.. Archivo
*MCAT/mrr**

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Querétaro

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN
COORDINACIÓN DE POSGRADO

CONSTANCIA DE APROBACIÓN DE TESIS

Santiago de Querétaro, Qro. a 29 de noviembre de 2019

Ma. Del Consuelo Alcántara Téllez
Jefa de la División de Estudios de Posgrado e Investigación
Instituto Tecnológico de Querétaro
P R E S E N T E.

Nos permitimos hacer de su conocimiento que después de haber procedido a la revisión y evaluación rigurosa y detallada de la Tesis de la C.

ING. ALICIA GUADALUPE MIJANGOS CONTRERAS

Cuyo título es:

DISEÑO E IMPLEMENTACIÓN DE UNA INTERFAZ GRÁFICA PARA GENERACIÓN DE TRAYECTORIAS DE UN ROBOT MÓVIL DE CONFIGURACIÓN DIFERENCIAL

Este jurado considera APROBADA dicha Tesis y se le notifica que la tesista puede continuar con los trámites correspondientes para obtener el Grado de Maestría.

Sin más por el momento, nos despedimos de usted.

Atentamente,

Comité Tutorial

DIRECTOR
Omar Alejandro Cervantes Gloria

CODIRECTOR
Daniel Hernández Arriaga

ASESOR
Raúl Ramírez López

c. c. p. Presidente comité tutorial
Expediente del alumno
alumno

Av. Tecnológico s/n esq. Mariano Escobedo, Col. Centro, C.P. 76000, Querétaro, Qro., México
Campus Centro Tel. 01(442) 2274400 Fax: 01(442)2169931
Campus Norte 01(442) 2435554
www.itq.edu.mx

Santiago de Querétaro, Qro. 27 de enero de 2020.

La que suscribe, egresada de la MAESTRÍA en INGENIERIA; de manera libre y voluntaria autorizo al Centro de Información del Instituto Tecnológico de Querétaro a difundir la obra de mi autoría con el Título del trabajo DISEÑO E IMPLEMENTACIÓN DE UNA INTERFAZ GRÁFICA PARA GENERACIÓN DE TRAYECTORIAS DE UN ROBOT MÓVIL DE CONFIGURACIÓN DIFRENCIAL. Para fines académicos, científicos y tecnológicos, mediante formato CD-ROM o digital, desde Internet, Intranet y en general cualquier formato conocido o por conocer.

Dicha obra estará disponible al estudiantado de esta Institución a partir del 31 de enero de 2020, fecha en la cual se puede difundir la obra.

Postulante: ALICIA GUADALUPE MIJANGOS CONTERAS

No. de Control: M17142642

Correo electrónico:alimcontreras09@gmail.com

Título de la obra: DISEÑO E IMPLEMENTACIÓN DE UNA INTERFAZ GRÁFICA PARA GENERACIÓN DE TRAYECTORIAS DE UN ROBOT MÓVIL DE CONFIGURACIÓN DIFRENCIAL

Área del conocimiento: INGENIERIA

Palabras clave de la obra: robot móvil, interfaz gráfica, modelado cinemático, linealización exacta, sistema embebido

Alicia Mijangos Contreras

NOMBRE Y FIRMA

Agradecimientos

A mis padres: Quienes me dieron la vida y me dieron todo sin esperar nada. Por su interminable apoyo, confianza y comprensión en todo momento de mi vida. Porque gracias a ustedes he realizado una de mis mejores metas. Les agradezco los esfuerzos que realizaron para que pudiera tener una buena educación, los consejos y todo el amor que me han brindado.

A mis hermanos: Por todas sus palabras de aliento y buenos deseos que siempre me dan, por todos los momentos que hemos pasado juntos y porque a pesar de todo siempre me han dado su apoyo y cariño.

A mi familia: A todos los demás miembros de mi familia, porque a pesar de la distancia siempre han estado conmigo, por los buenos deseos que siempre me brindan y por todos los momentos que hemos compartido.

Al Mtro. Omar Cervantes: Por su apoyo, consejos, sugerencias y críticas para la culminación de este trabajo. Por la confianza que me brindó para realizar este proyecto y por la paciencia que me tuvo por más de dos años.

A mis amigos: Por qué siempre me han escuchado y aconsejado en los buenos y malos momentos y por estar siempre a mi lado a pesar de la distancia.

A mis amigos y compañeros del ITQ: Por brindarme su amistad y apoyarme en los buenos y malos momentos que pasamos en estos dos años. Les deseo lo mejor y espero que podamos mantenernos unidos sin importar la distancia.

Al CONACYT: Por su apoyo económico para realizar mis estudios de Maestría.

Resumen

El presente proyecto propone el diseño e implementación de un sistema de control de robots móviles mediante una interfaz gráfica capaz de generar trayectorias de manera dinámica. Su finalidad es proveer un sistema completo que permita la realización de pruebas de seguimiento de trayectorias mediante una arquitectura modular y que no se encuentre restringido a un solo sistema robótico. Este trabajo considera un sistema robótico móvil de configuración diferencial capaz de seguir las trayectorias diseñadas mediante la interfaz con un error cercano a cero. Las leyes de control empleadas para su validación son a través del método de retroalimentación por linealización exacta

Abstract

Following project proposes the design and implementation of a mobile robotic control system by means of a graphical user interface (GUI) that is capable of dynamic trajectories generation. Its final purpose is to provide a complete system that allows path following testing, with a modular architectural design and not restricted to a single robotic system. This work considers a differential robotic system able to track the desired trajectories designed through the graphical interface with a near to zero error. Control laws applied for this process are called exact linearization feedback.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1. INTRODUCCIÓN

El desarrollo de la robótica ha atendido considerablemente el campo de los robots manipuladores debido a su fuerte presencia en el campo industrial. Sin embargo, a medida que la tecnología avanza, los robots móviles, especialmente los robots móviles con ruedas (RMR), empezaron a ganar un amplio terreno dada la reducción de la complejidad en su construcción. La robótica ha logrado impulsar el desarrollo económico, científico y social de la humanidad. Como consecuencia, el control de este tipo de robots comenzó a ser el tema de múltiples estudios de investigación, sobre todo en aquellas basadas en sistemas autónomos que hacen posible que la supervisión humana sea la mínima posible, tal es el caso de aquellos enfocados en el seguimiento de trayectorias.

Actualmente, los robots móviles han adquirido una importante relevancia por su variedad de aplicaciones en diversos sectores productivos para desarrollar tareas de exploración, transporte y recolección de productos, entre otras. Este tipo de labores requieren que el robot posea una unidad de procesamiento de datos con velocidad de respuesta relativamente elevada, destinados a la adquisición de las variables de posición, así como la aplicación de controladores discretos y algoritmos de control implementados en sistemas embebidos. Además, debe poseer un sistema robusto capaz de controlar su trayectoria durante la navegación, así como un sistema de monitoreo, supervisión y control en tiempo real. El objetivo central del proyecto es desarrollar un sistema integral que permita la reconfiguración de trayectorias para un robot móvil con un sistema de control preciso y eficiente, que sea de arquitectura abierta para permitir el posterior desarrollo de aplicaciones específicas.

Trabajos como los realizados en el año 2015 en el Centro de Investigación y de Estudios Avanzados del IPN titulado: *"Control de formación líder-seguidor por backstepping para un robot tipo uniciclo y un cuadricóptero" (*M.A. Vallejo-Alarcón, R. Castro-Linares, M. Velasco-Villa, 2015) continúan en esta línea de investigación demostrando la interacción de vehículos tanto terrestres como aéreos en el seguimiento conjunto de trayectorias. Las aplicaciones que conducen a este tipo de trabajos son muy extensas: vigilancia, gestión de desastres naturales, operaciones militares, aplicaciones en el sector agrícola, entre otras.

1.1 Planteamiento del Problema

Actualmente, un amplio número de investigadores, profesionistas, docentes, estudiantes y demás integrantes de la comunidad tecnológica a lo largo del mundo se encuentran desarrollando trabajos de investigación y proyectos tecnológicos con el fin de resolver problemas de diferentes índoles aplicando robots móviles autónomos aéreos y terrestres. Invariablemente, siempre es necesaria la incorporación de una interfaz que permita el intercambio de información, la recolección de datos que conduzcan a la correcta validación del sistema y, si es el caso, el rediseño y mejoramiento de los equipos propuestos.

El estudio de los vehículos autónomos, tanto terrestres como aéreos, posee una extensa bibliografía que les concede de una amplia caracterización; sin embargo, estos robots se encuentran sujetos a la predefinición de las trayectorias, lo que vuelve complicado las pruebas en línea o en campo. Esta tesis se enfoca en la realización de una interfaz gráfica de usuario que permita la generación de trayectorias en línea, de acuerdo a las posibilidades y limitantes del espacio de trabajo.

1.2 Justificación

En la actualidad existen sistemas comerciales que funcionan como interfaces de usuario para el uso de robots móviles; sin embargo, estos presentan diversos inconvenientes entre los cuales destacan:

- Son aplicables a un solo tipo de vehículo (terrestre o aéreo).
- Su uso se limita a la aplicación para la cual ha sido diseñada.
- No funcionan con otros sistemas fuera del que acompaña su producto comercial.
- Son de arquitectura cerrada.
- Frecuentemente se requiere la compra de licencias para modificar el software preestablecido de fábrica.
- No es posible la modificación del hardware.
- Generalmente no permiten la validación de los sistemas propios.

El desarrollo del presente proyecto permite contar con un sistema flexible, que sea aplicable a diferentes tipos de robots móviles y que, además, sea de arquitectura abierta, lo que fomenta el desarrollo y validación de nuevos proyectos de base tecnológica para su implementación en los sectores productivos y sociales.

Por otro lado, la realización de este proyecto apoya la formación de profesionistas de alto rendimiento mediante el desarrollo y la programación de sistemas de control y de algoritmos para vehículos autónomos.

Los sistemas de arquitectura abierta como el propuesto a través de este proyecto, es decir, sistemas programados con software de libre acceso y que utilizan hardware genérico, permiten modificaciones en el diseño original de tal manera que se puedan reutilizar acorde a diversas aplicaciones específicas. Esto representa una ventaja económica ante los sistemas comerciales disponibles en el mercado cuya arquitectura de software-hardware (SW/HW) es completamente cerrada y prácticamente imposible de modificar o mejorar.

El sistema al ser diseñado de manera modular, permite la migración entre sistemas de hardware de manera transparente, pudiendo cambiar de un vehículo terrestre a uno aéreo o inclusive submarino sin mayores complicaciones. La interfaz gráfica será, por lo tanto, portable en diversas plataformas

1.3 Objetivos

Objetivo General:

Diseñar y construir un sistema de control de seguimiento de trayectoria para un robot móvil de configuración diferencial mediante interfaz gráfica en computadora.

Objetivos Específicos:

- Modelar y simular un robot móvil de configuración diferencial mediante un programa computacional.
- Construir un robot móvil autónomo que permita la validación del sistema propuesto.
- Diseñar la interfaz gráfica mediante una computadora para la generación de trayectorias.
- Implementar el algoritmo de control en el sistema embebido del robot móvil.

CAPÍTULO 2. MARCO TEÓRICO

2.1 Componentes principales de un robot móvil

Los robots móviles son dispositivos electromecánicos capaces de moverse en un espacio determinado y generalmente están asociados a tareas repetitivas o posiblemente dañinas al ser humano. En los últimos años la investigación sobre robots móviles está adquiriendo gran notoriedad, en parte debido a la disminución en costos del hardware necesario para su construcción.

Los principales componentes de un robot móvil son:

- Sensores externos: que captan una percepción del entorno: visión, tacto, audición, proximidad, etc.
- Sensores internos: que miden el estado de la estructura mecánica: dirección, desplazamiento, velocidad, etc.
- Actuadores: Sistemas electromecánicos que generan fuerza y par para el movimiento: Sistemas eléctricos, mecánicos o neumáticos.
- Unidad de Procesamiento: En este sistema se implementan los algoritmos de control y navegación mediante la programación de un dispositivo embebido que utilizan los sensores como entradas y los actuadores como salidas. Esto define el comportamiento del robot.
- Sistemas de control que aseguran el funcionamiento correcto de los movimientos (bucles de control), planificación (trayectorias), etc.

Encoders

Dado que este proyecto de tesis se centra en la generación y seguimiento de trayectorias, el uso de encoders es una parte fundamental en la construcción del sistema robótico. Un encoder óptico es un mecanismo que se utiliza para calcular la posición, velocidad y aceleración de un eje. Este dispositivo optoelectrónico convierte el movimiento angular de su eje en un tren de pulsos digitales. Su principio de funcionamiento consiste en la detección de un haz de luz generado por un emisor de

luz que atraviesa las ranuras de un disco a través de un fotoreceptor, tal como se muestra en la figura 1.

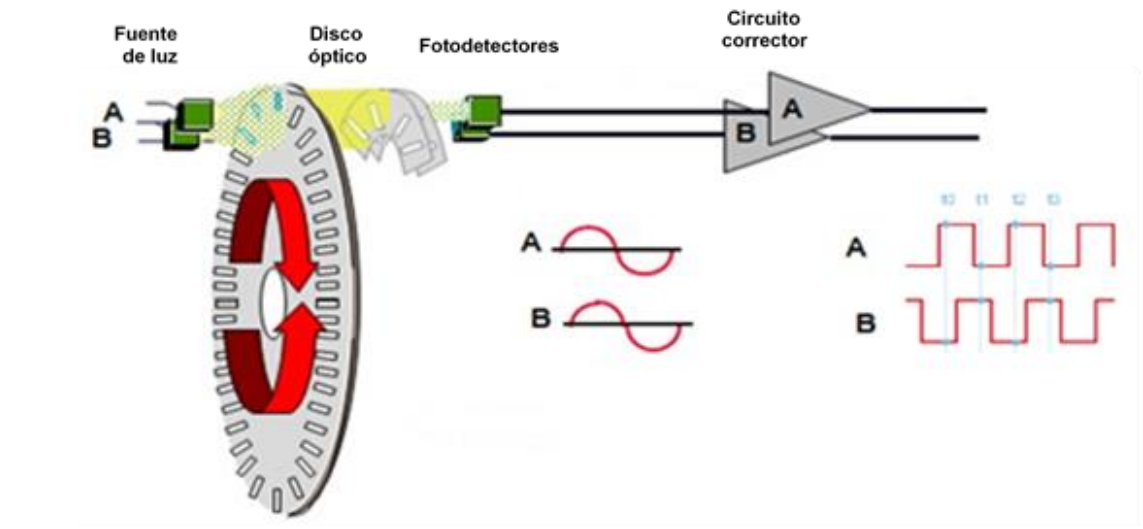


Figura 1 Encoder Óptico Incremental, fuente: Parker 2019

Cuando el fotoreceptor detecta el haz de luz se obtiene un "1" lógico y cuando no se detecta se obtiene un "0" lógico. De esta manera el encoder generará un tren de pulsos en función al movimiento angular transmitido a su eje. Los encoders proporcionan dos ondas cuadradas y desfasadas entre sí en 90º, los cuales llamaremos canal A y canal B, según se muestra en la figura 1. Con la lectura de un solo canal (canal A) se dispone de la información necesaria para calcular únicamente la velocidad de rotación, mientras que si se capta también la señal B es posible discriminar el sentido de rotación en base a la secuencia de datos que producen ambas señales. Como medida del desplazamiento tangencial asociado al eje del encoder, se acopla una rueda de tal manera que el movimiento producido por la

rueda se transmite al eje del encoder generando *ticks* (pulsos) positivos y negativos en función al sentido de giro.

Interfaces Gráficas de Usuario

Las interfaces gráficas de usuario, conocidas también como GUI (del inglés *graphical user interface*), es un tipo de programa que actúa de interfaz de usuario, utilizando un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz. Su principal uso consiste en proporcionar un entorno visual sencillo para permitir la comunicación con el sistema operativo de una máquina o computador.

Habitualmente las acciones que ejecutan un control o comando sobre el sistema informático se realizan mediante manipulación directa, para facilitar la interacción del usuario con la computadora. Las interfaces graficas de usuario surgen como evolución de las interfaces de línea de comandos que se usaban para operar los primeros sistemas operativos y es pieza fundamental en un entorno gráfico. Como ejemplos de interfaz gráfica de usuario cabe citar los entornos de escritorio *Windows*, el *X-Windows* de *GNU/Linux* o el de *Mac OS X, Aqua*.

En el contexto del proceso de interacción persona-computadora, la interfaz gráfica de usuario es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.

La integración de un sistema de alta complejidad, como la de un robot que tiene un campo de acción en constante cambio, se convierte en una tarea que requiere la capacidad de una pronta respuesta a estas variaciones, como comandar la velocidad deseada, cambiar las trayectorias, monitoreo, etc. Estas acciones se ven facilitadas

por la inclusión de un sistema que provea una interfaz gráfica que permita la pronta modificación de estas variables (o despliegue de las mismas).

Visión Artificial

La visión artificial tiene como objetivo generar descripciones inteligentes y útiles de escenas y secuencias visuales, así como de los objetos que aparecen en ellas, mediante la realización de operaciones sobre imágenes y vídeos.

Entre las tareas más típicas de visión artificial están:

- Detección, reconocimiento, identificación y seguimiento de objetos
- Análisis de movimiento
- Reconstrucción de escenas
- Restauración de imágenes
- Calibración de cámaras
- Visión estéreo

La visión artificial consiste en la captación de imágenes en línea mediante cámaras basadas en matrices de sensores sensibles a la luz (CCD o CMOS), el posterior tratamiento de las mismas mediante técnicas de análisis de imagen y la actuación sobre el proceso.

Generalmente la visión artificial se divide en las siguientes áreas:

***Captación**:* Proceso necesario para obtener una imagen de calidad que no altere las condiciones reales agregando errores considerables al sistema.

Pre procesamiento: Métodos para quitar o reducir características no deseadas en la imagen como el ruido.

Segmentación: Dividir una imagen en objetos que sean de interés.

Descripción: Obtención de características de un objeto como su forma, tamaño, color con el fin de diferenciarlo de otros objetos.

Reconocimiento: Proceso a través del cual se identifica un objeto dentro de una escena.

Interpretación: Asociar un significado a un conjunto de objetos reconocidos.

2.2 Clasificación de los robots móviles terrestres

Los robots móviles son dispositivos electromecánicos capaces de desplazarse en un espacio de trabajo con cierto grado de autonomía (Vázquez, 2005). En cuanto a los sistemas robóticos móviles, el estudio de estos se ha visto incrementado debido a su utilidad y flexibilidad en tareas tales como la exploración, búsqueda y vigilancia.

Los robots móviles se dividen en tres tipos de acuerdo al entorno en que se desplazan, los cuales son: terrestres, acuáticos y aéreos. Los robots móviles terrestres son los más utilizados en la actualidad y se pueden clasificar por el tipo de locomoción:

Robot móvil - orugas: Las orugas se construyen uniendo mediante una cadena que rodea las llantas, las ruedas delanteras y traseras, cuyo fin es aumentar la superficie de contacto con el suelo y conseguir una mayor tracción. Las orugas permiten rebasar mayores obstáculos que solamente usando ruedas e incluso subir escaleras. Como desventaja, presenta la gran cantidad de energía que necesita el robot en los giros. El giro se realiza rotando las cadenas en sentidos opuestos de modo que las fuerzas inversas hacen girar el robot.

Robot móvil - por patas: son sistemas complejos, inestables y difíciles de controlar. Son utilizados cuando la tarea a la que se destina el robot requiere de movilidad. Los diseñadores de este tipo de robots han intentado imitar las distintas formas de

desplazamiento con que la naturaleza ha dotado a los insectos y animales, incluidos los seres humanos. Al dotar de movimiento con patas a un robot, debemos tomar en cuenta su posición y velocidad, pero también debemos asegurar que el robot permanezca en equilibrio y no caiga, usando solamente el movimiento en las articulaciones mediante motores. En robots bípedos, el desplazamiento requiere necesariamente mantener el equilibrio, lo que conlleva inestabilidad en el movimiento.

Robot móvil - por ruedas: Son los sistemas más utilizados por su simplicidad, eficiencia y relativamente más fáciles de controlar. Para que un robot que use ruedas pueda moverse simplemente hay que suministrar energía al eje de las ruedas motrices. Además, con este tipo de locomoción, el robot puede desplazar mayor peso que usando patas. En cuanto a las desventajas, los robots con ruedas están limitados a terrenos planos y las ruedas no permiten pasar por obstáculos grandes. En concreto, cualquier objeto que tenga más altura que el radio de la rueda, no podrá ser librado por esta. Al diseñar el robot, se debe decidir cuál va a ser la disposición de las ruedas y cuáles van a ser las motrices, es decir las que proporcionan el movimiento.

2.2.1 Configuración de los robots móviles por ruedas

De acuerdo al tipo de ruedas y al arreglo de las mismas, se pueden identificar configuraciones comunes dentro de la robótica móvil: Configuración diferencial, Síncrono, Triciclo, Ackerman o tipo automóvil.

Robot móvil con configuración diferencial. Se le considera la configuración de accionamiento más simple para un robot móvil con ruedas (Gómez, 2011). Es usado en pequeños y baratos robots destinados a tareas en interiores. Un robot móvil con configuración diferencial consiste en dos ruedas montadas sobre ejes colineales, cada una controlada por un motor diferente.

Bajo esta configuración, para que cada rueda produzca movimiento, el robot debe rotar alrededor de un punto ubicado en el eje de las ruedas denominado centro instantáneo de curvatura o ICC por sus siglas en inglés (*instantaneous center of curvature*) variando la velocidad relativa de ambas ruedas, el punto de rotación puede ser modificado, provocando que el robot describa diferentes trayectorias. Es importante notar que, si la velocidad en ambas ruedas es la misma, el radio de la trayectoria descrita respecto al centro instantáneo de curvatura o ICC tiende a infinito, por lo que el robot realizará una trayectoria recta. Si las velocidades son iguales en magnitud, pero contrarias en sentido, el radio de la trayectoria descrita respecto al ICC es cero y el robot girará sobre un eje ubicado en medio de ambas ruedas del robot.

Robot móvil síncrono. En un robot móvil síncrono todas las ruedas giran y producen impulso al unísono, todas las ruedas apuntan en la misma dirección y giran al mismo tiempo. Estos vehículos controlan la dirección de las ruedas y la velocidad a la que giran. En esta configuración, cada rueda es capaz tanto de producir un impulso como de modificar su orientación.

Configuraciones típicas incluyen un arreglo de tres ruedas situadas en los vértices de un triángulo equilátero. Una rueda direccionable, es aquella que permite controlar la orientación de su eje de rotación.

Una solución común en robots de este tipo es utilizar dos motores independientes, uno para producir desplazamiento y otro para rotar las ruedas.

Robot móvil tipo triciclo. Un robot tipo triciclo clásico tiene tres ruedas, las dos traseras y la delantera que incorpora capacidad de orientación e impulso. El movimiento del robot es controlado por la rotación y giro de la llanta delantera.

Robot móvil con configuración *Ackerman*. Este tipo de configuración se puede observar en la mayoría de los automóviles. En el modelo de dirección *Ackerman*, cada rueda delantera de dirección rota en brazos separados con el objetivo de permitir tener diferentes ángulos en cada rueda respecto al CIR. La rueda interna debe girar un ángulo mayor que la exterior y la rueda interior recorre una distancia menor que la externa. La dirección *Ackerma*n es el mecanismo preferido para vehículos grandes de los cuales se espera rueden sobre caminos existentes o sirvan para trasladar grandes cantidades de carga.

2.2.2 Propiedades cinemáticas de los robots móviles con ruedas

(González Villela, 2012) hace mención de tres propiedades cinemáticas de movimiento de un robot móvil.

Grado de movilidad $\boldsymbol{r_m}$*.* Define el número de variables que se pueden controlar y, como consecuencia, los grados instantáneos de libertad que el robot puede manipular desde sus entradas sin orientar ninguna de sus ruedas.

Grado de manejabilidad $\boldsymbol{r_s}$*.*Define el número de ruedas orientables centradas que pueden ser controladas.

Grado de maniobrabilidad $\boldsymbol{r_M}$*.* Define el total de grados de libertad que el robot puede manipular.

$$r_M = r_m + r_s$$

Ecuación 1 Grados de Libertad

Tabla 1 Configuraciones de robots móviles con ruedas

		$Tipo\ (r_m, r_s)$				
		(3,0)	(2,0)	(2,1)	(1,1)	(1,2)
Grado	r_m	3	2	2	1	1
	r_s	0	0	1	1	2
	r_M	3	2	3	1	3

Como consecuencia de las definiciones anteriores, los robots móviles con ruedas son clasificados en cinco configuraciones no singulares, de acuerdo al grado de movilidad (grados instantáneos de libertad), a su grado de manejabilidad (número de grados de direccionabilidad) y a la suma de ambos, definida como el grado de maniobrabilidad (total de grados de libertad). El grado de maniobrabilidad por sí mismo no define el tipo de robot móvil, debido a que dos robots con el mismo $\boldsymbol{r_M}$ pero diferente $\boldsymbol{r_m}$ no son equivalentes. Los valores $\boldsymbol{r_m}$ y $\boldsymbol{r_s}$ son necesarios para definir el tipo de robot como tipo ($\boldsymbol{r_m}$, $\boldsymbol{r_s}$).

Tipo (3,0). Estos robots no tienen ruedas fijas ni centradas, únicamente tienen ruedas orientables no centradas o suecas. Estos robots son llamados omnidireccionales debido a que tienen movilidad total en el plano, lo que significa que se pueden mover en cualquier instante de tiempo en cualquier dirección y con cualquier orientación.

Tipo (2,0). Estos robots no tienen ruedas centradas, tienen una o varias ruedas fijas en el mismo eje.

Tipo (2,1). Estos robots no tienen ruedas fijas y tienen al menos una rueda centrada.

Tipo (1,1). Estos robots tienen una o varias ruedas fijas en un único eje. Asimismo, tienen una o varias ruedas centradas con la condición de que el centro de una de ella no esté localizado en el eje de las ruedas fijas.

Tipo (1,2). Estos robots no tienen ruedas fijas y tienen al menos dos ruedas centradas. Cabe señalar que el único robot omnidireccional es el tipo (3,0), debido a que su $\boldsymbol{r_m}$ = 3, mientras que los otros 4 mencionados tienen un $\boldsymbol{r_m}$ < 3.

En este proyecto se consideran robots móviles propulsados por ruedas, que poseen dos grados de movilidad con 2 variables a controlar (movimiento lineal y rotacional) y cero grados de manejabilidad, ya que no tiene ruedas orientables centradas que pueden ser controladas de ahí su denominación de robot móvil tipo (2,0).

Este robot móvil posee una configuración diferencial, ya que cuenta con dos ruedas situadas diametralmente opuestas en un eje perpendicular a la dirección del robot. Cada una de las ruedas esta acoplada a un motor independiente, esto permite tener velocidades distintas que generan un ángulo de rotación, este ángulo está en función de las velocidades de cada uno de los motores. Para que el robot se desplace hacia adelante las velocidades de ambos motores deberán ser iguales escalarmente y tener el mismo sentido. Si la velocidad del motor del lado derecho es mayor a la velocidad del motor del lado izquierdo, el robot se desplazará hacia la izquierda. Si la velocidad del motor del lado izquierdo es mayor a la velocidad del motor del lado derecho, el robot se desplazará hacia la derecha. Una de las ventajas de la configuración diferencial radica en que el robot puede girar sobre su propio eje en sentido horario o anti horario. Para realizarlo, las velocidades de ambos motores tienen que ser iguales, pero de sentidos opuestos.

CAPITULO 3. ESTADO DEL ARTE

En cuanto a trabajos que involucran el desarrollo de interfaces gráficas de usuario es conveniente tomar en consideración la tesis de posgrado titulada: "*Coordinación de estructuras mecánicas virtuales"* (González, 2013) realizado en el Centro de Investigación y Estudios Avanzados (CINVESTAV). Este trabajo cuenta con una interfaz gráfica capaz de proveer trayectorias definidas mediante modelos matemáticos.

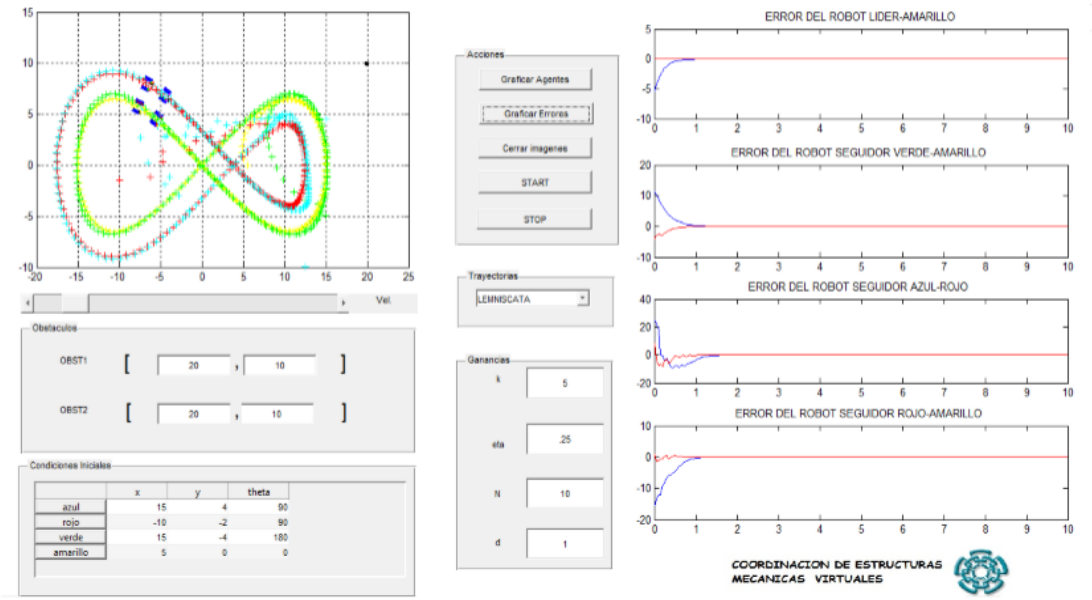


Figura 2 Interfaz de control de tesis de posgrado "Estructuras Mecánicas Virtuales"

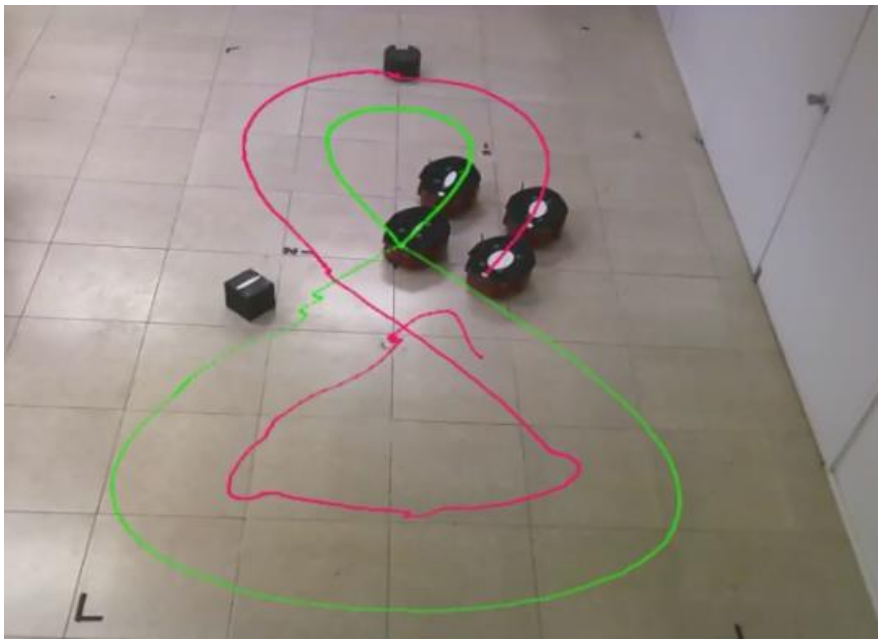


Figura 3 Trayectorias de tesis de posgrado "Estructuras Mecánicas Virtuales"

Otro trabajo previo que involucra interfaces de usuario en robots móviles es la tesis de posgrado: *"Diseño e implementación de un robot móvil con control de trayectoria mediante principios odométricos"* (Arellano, 2015), en el que el usuario, a través de una interfaz gráfica modifica las coordenadas, lo que permite al robot móvil reconfigurar su trayectoria en sitio.

Figura 4 Interfaz gráfica de usuario de tesis de posgrado (Arrellano, 2015)

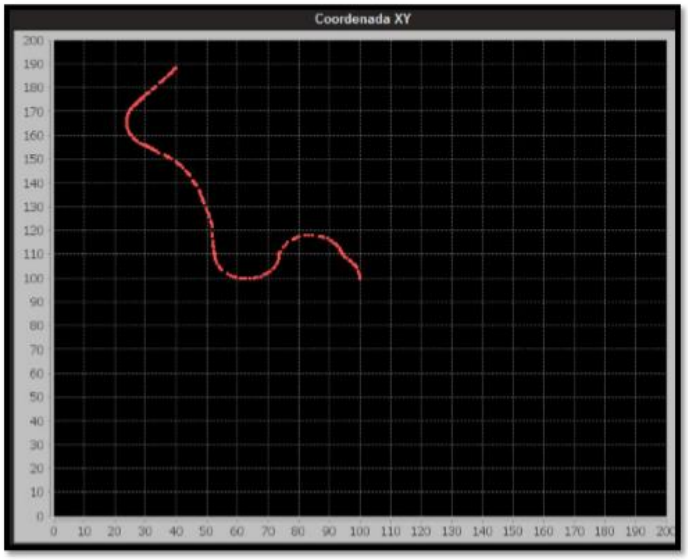


Figura 5 Gráfica del seguimiento de trayectoria de tesis de posgrado (Arrellano, 2015)

Finalmente, como un producto comercial ya disponible se encuentra la empresa *Marvelmind*, que a través de su sistema *indoor GPS*, permite al usuario mediante su interfaz generar trayectorias en un espacio caracterizado a través de *beacons*, que proveen un marco de referencia y el rastreo de objetos de manera inmediata. Esta funcionalidad es aplicable al campo de la robótica, con lo que se pueden realizar sistemas de seguimientos de trayectoria con un error de ±2 cm.

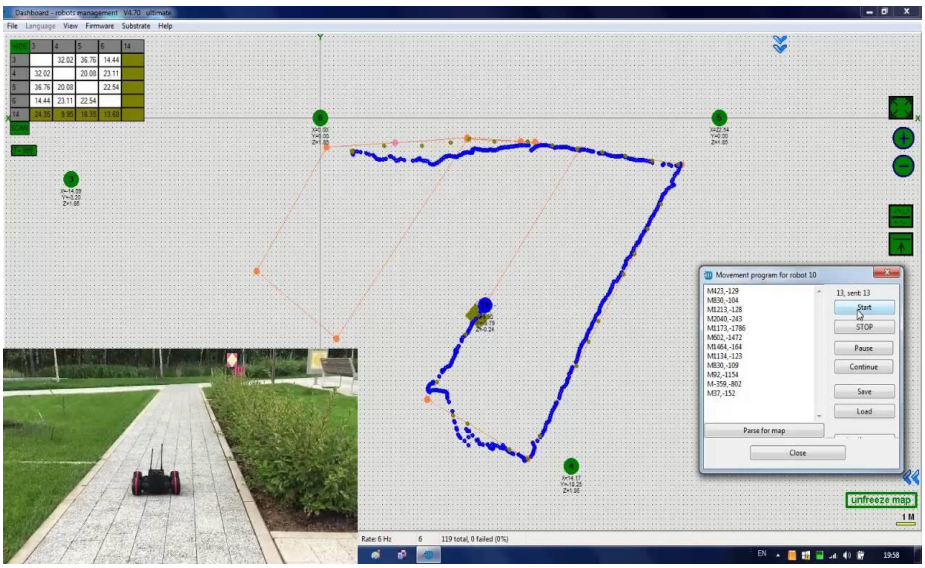


Figura 6 Interfaz gráfica de usuario de la empresa *MarvelMind*

CAPÍTULO 4. METODOLOGÍA

Para la realización de este proyecto se utiliza la metodología basada en el modelo “V”, que es aplicada para trabajos relacionados con software, este método representa la secuencia de pasos en el desarrollo del ciclo de vida de un proyecto.

Se describen las actividades y resultados que deben producirse durante el desarrollo del sistema. Para fines prácticos se clasifica en 5 etapas: Análisis, Diseño, Desarrollo e Implementación, Pruebas y Resultados, como se muestra en la figura 7, el lado izquierdo de la “V” representa la descomposición de las necesidades, y la creación de las especificaciones del sistema. El lado derecho de la “V” representa la integración de las piezas y su verificación. “V” significa «verificación y validación». Es muy similar al modelo de la cascada clásico ya que es muy rígido y contiene una gran cantidad de iteraciones.

La unión mediante líneas discontinuas entre las fases de la parte izquierda y las pruebas de la derecha representa una doble información. Por un lado, sirve para indicar en qué fase de desarrollo se deben definir las pruebas correspondientes. Por otro sirve para saber a qué fase de desarrollo hay que volver si se encuentran fallos en las pruebas correspondientes.

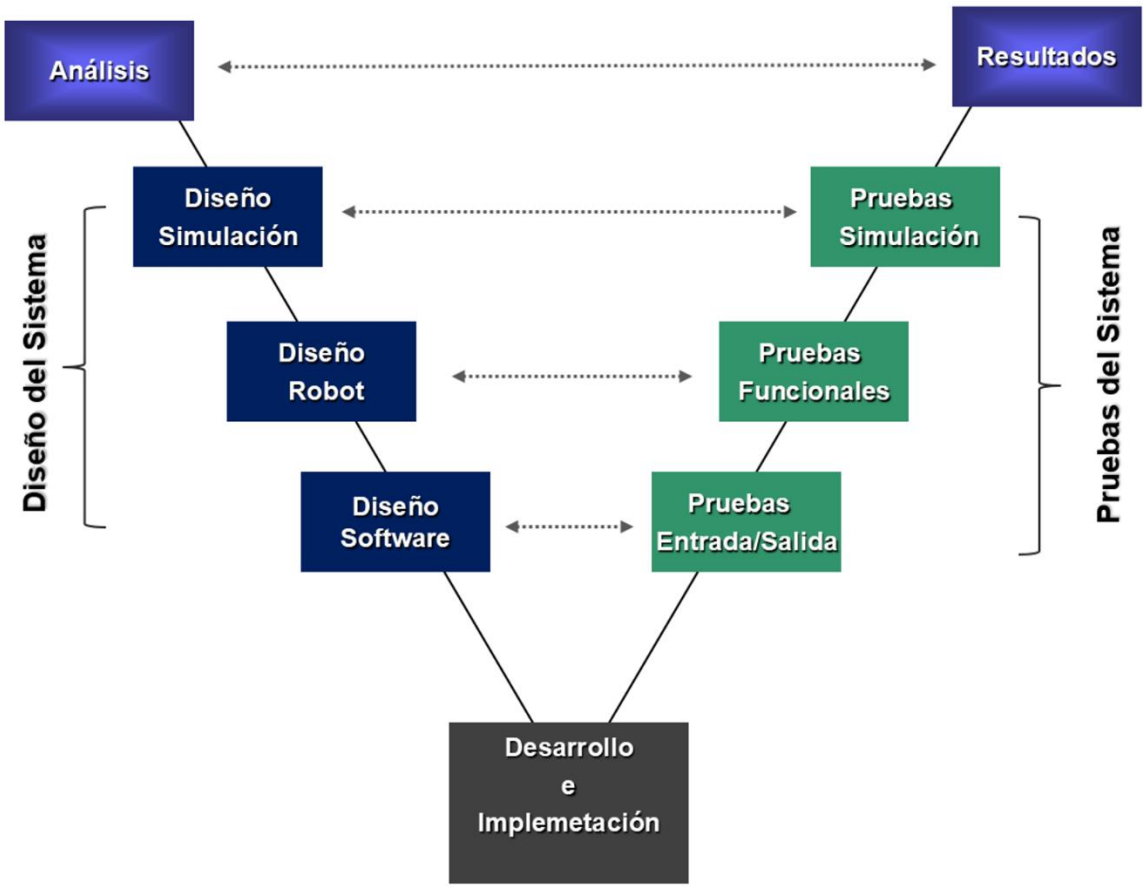


Figura 7 Descripción en etapas de metodología general en" V"

4.1 Análisis

Durante esta fase se precisaron los alcances del proyecto a construir de acuerdo a los objetivos específicos, esto requiere de establecer las necesidades que necesitan ser cubiertas por el sistema propuesto y determinar que herramientas comerciales o de desarrollo interno serán necesarias para su realización

Requerimientos, restricciones y delimitaciones del sistema

El sistema es de arquitectura abierta y modificable, la interfaz gráfica de usuario es desarrollada en ambiente .NET lo que permite su portabilidad en sistemas Windows, las trayectorias a seguir por el robot móvil serán diseñadas en dicha interfaz gráfica.

El área de tránsito del robot móvil se asume en un terreno plano sin deslizamiento. La trayectoria es generada en un espacio conocido y sin obstáculos. El área de trabajo utilizada en este proyecto es de 3*4 metros.

Visual Basic .NET es el sistema de programación elegido para el diseño de la interfaz gráfica de usuario. Las ventajas de este ambiente de codificación permite usar con suma facilidad la plataforma de los sistemas Windows dado que tiene acceso prácticamente total a la API (*application programming interface)* de Windows incluidas librerías actuales, este lenguaje es compatible con *Microsoft Office*, ya que existe una versión integrada en las aplicaciones de *Office*, versiones tanto como *Windows* y *Mac* que permite programar macros para extender y automatizar funcionalidades en documentos, tiene una implementación de la POO (Programación Orientada a Objetos), la cual es un paradigma de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas de computadora.; es fácilmente extensible mediante librerías DLL (*dynamic-link librar*y) y componentes *Activex* de otros lenguajes, este tipo de lenguaje es extendido, permite el tratamiento de mensaje de Windows, es excelente para cálculos intensivos del CPU, como por ejemplo operaciones matemáticas.

El tipo de robot utilizado para validar la interfaz gráfica de usuario es un robot de configuración diferencial tipo (2,0) debido a su simplicidad. Este tipo de robots pueden ser descritos matemáticamente mediante modelos cinemáticos que simplifican los cálculos realizados, dado que en este tipo de análisis la posición, velocidad y aceleración de cada uno de los elementos del robot son calculadas sin considerar las fuerzas que causan el movimiento. El robot no tendrá sensores externos que apoyen en el seguimiento de la trayectoria, esta acción será soportada exclusivamente por los encoders ópticos para retroalimentación de la velocidad.

Previo a la construcción del robot móvil, el sistema será simulado mediante el programa *Simulink/Matlab,* partiendo de un modelo cinemático (posiciones y velocidades). Este tipo de modelos representa ventajas debido a la simplificación de los cálculos necesarios para obtener la salida del sistema, en comparación con modelos dinámicos, en los que las variables a medir pueden ser desconocidas o requieren de constantes mediciones. Esta simplificación del modelo matemático también permite utilizar un control menos complejo, como por ejemplo la linealización exacta, que no requiere de variables no lineales en el procesamiento.

La selección de la plataforma en la cual se desarrolla la interfaz gráfica está asociada al ambiente que nos permita el reuso eficiente de librerías gráficas para reducir considerablemente los tiempos de desarrollo. Sistemas basados en el *framework* .NET permiten cubrir estos requisitos del sistema.

4.2 Diseño

En esta etapa se delimita la arquitectura software/hardware de la interfaz de usuario. Las especificaciones técnicas consideradas proveen al sistema de las entradas requeridas para su implementación y validación.

A continuación, se describe el diseño electrónico, mecánico y computacional en base al modelo matemático de un robot móvil de configuración diferencial y a su respectiva simulación numérica. Debido a que es un sistema compuesto de diversas disciplinas, la metodología de diseño se divide en las siguientes etapas:

Diseño y simulación: Permite comprobar los supuestos matemáticos del sistema y hacer ajustes en los modelos antes de la implementación real.

Diseño del robot móvil: De acuerdo al análisis de requisitos, el sistema debe contar con las entradas y salidas necesarias para poder cumplir con el funcionamiento esperado. Estas entradas y salidas se definen, como puertos de comunicación y control del sistema robótico.

Diseño de Software: Finalmente, los algoritmos desarrollados serán quienes dicten el comportamiento del sistema y su respuesta a los estímulos presentes, como las posiciones y velocidades deseadas. Por lo tanto, el software deberá ser capaz de leer puertos de entrada, generar señales de control proporcionales al error de posición, recalcular los nuevos valores si las condiciones actuales cambian, etc.

4.2.1. Modelado y Simulación Numérica

Un modelo matemático consiste en una ecuación o un grupo de ecuaciones que representa un sistema físico, muchos fenómenos físicos pueden describirse mediante modelos matemáticos, estos modelos pueden basarse en teorías o leyes científicas que han resistido la prueba del tiempo, otros tienen una base empírica y se obtiene a base de observaciones y experimentos.

Con la finalidad de entender el comportamiento de los sistemas es necesario obtener modelos matemáticos que los representen. Un modelo matemático es una réplica de las relaciones de entrada y salida o entre entradas y salidas. Las relaciones reales entre la entrada y la salida de un sistema se sustituyen por expresiones matemáticas. Para diseñar el modelo de un sistema se debe empezar a partir de una predicción de su funcionamiento antes que el sistema pueda diseñarse en detalle. Normalmente el modelo matemático se trata de una serie de ecuaciones diferenciales que describen el comportamiento del sistema (modelo teórico).

Durante la primera etapa de este proyecto de tesis, el modelado matemático y la simulación son útiles para probar condiciones que podrían resultar difíciles de

reproducir solamente con prototipos de hardware, especialmente en la primera fase del proceso de diseño, cuando es posible que no esté disponible el hardware. La iteración entre el modelado matemático y la simulación mejora la calidad del diseño del sistema en una etapa temprana y reduce así el número de errores descubiertos más adelante en el proceso de diseño.

Para realizar el modelado del robot móvil se tomaron en cuenta ciertas hipótesis que generalizan el comportamiento del mismo, por ejemplo, se asume que el robot se desplaza en una superficie plana idealmente sin rozamiento, también se toman los ejes de las ruedas como perpendiculares al suelo por donde se desplaza; por último el robot se debe mover únicamente por las fuerzas ejercidas por el movimiento rotacional de las ruedas, el robot es considerado como un mecanismo sólido, rígido y sin partes flexibles, pero se deben tener en cuentas las restricciones no holonómicas del sistema. Es decir, el robot puede desplazarse hacia atrás o adelante, pero no puede trasladarse a los lados sin que exista una fuerza externa que obligue el movimiento.

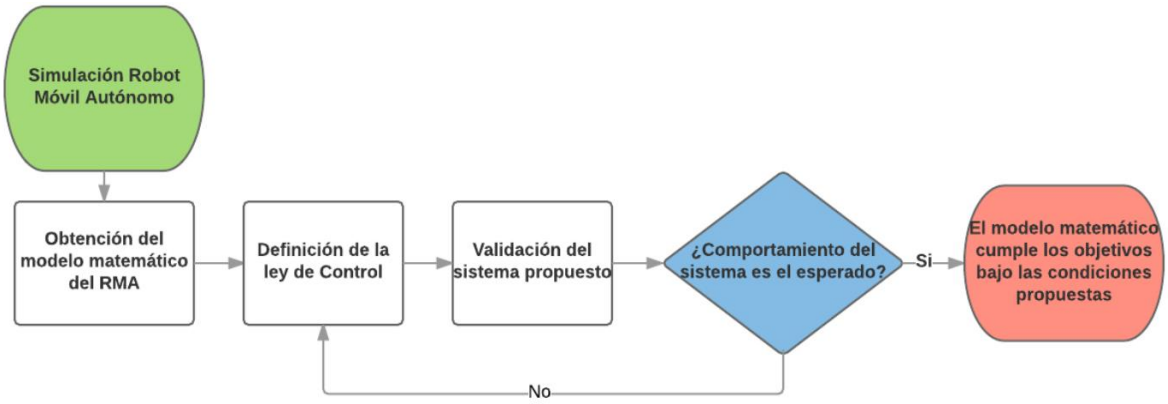


Figura 8 Metodología de simulación robot móvil

Para que un sistema robótico pueda realizar una tarea de manera óptima es necesario contar con una serie de postulados matemáticos que permitan conocer la

posición y orientación del robot y en caso de ser necesario, de los objetos que este tenga que manipular.

La forma más intuitiva de localizar espacialmente los puntos de un cuerpo es el sistema cartesiano, que se define mediante ejes perpendiculares entre sí con un origen definido. Para localizar un punto dentro de este sistema se emplean las siguientes coordenadas:

- Coordenadas cartesianas: un punto a vendrá expresado por las componentes (x, y) correspondientes al sistema OXY . El punto tiene asociado un vector $\vec{p}$ (x, y) que va desde el origen O del sistema OXY hasta el punto a . Esto sería válido para el caso de dos dimensiones. Si se está trabajando con un sistema de tres dimensiones el vector $\vec{p}$ estará definido por las componentes (x, y, z) con respecto al sistema $OXYZ$.

- Coordenadas polares y cilíndricas: se pueden emplear las coordenadas polares $p(r, \theta)$, en las que representa la distancia del punto al origen y θ *r*epresenta el ángulo del vector $\vec{p}$ con el eje OX. Mediante las coordenadas cilíndricas $p(r, \theta, z)$ se define un punto en tres dimensiones análogamente al caso de las coordenadas polares. La componente z representa la proyección sobre el eje OZ del vector $\vec{p}$.

- Coordenadas esféricas: se utilizan para localizar un vector en un espacio tridimensional. El vector $\vec{p}$ tendrá como coordenadas esféricas (r, θ, φ), donde la componente r es la distancia desde el punto hasta el origen, θ es el ángulo formado por la proyección del vector $\vec{p}$ sobre el plano OXY con el eje OX y la componente φ es el ángulo formado por el vector $\vec{p}$ con el eje OZ.

Para un sistema robótico móvil, es necesario además de la posición, definir la orientación con respecto a un sistema de referencia. El sistema de posicionamiento utilizado en este modelado será mediante coordenadas cartesianas.

La cinemática del robot estudia el movimiento del mismo respecto a un sistema de referencia, pero sin atender a las fuerzas que lo producen (Cortés, Castañeda, Benítez y Díaz, 2015). El problema cinemático, en el caso de los robots móviles, consiste en encontrar una matriz homogénea de transformación T que relacione la posición y orientación del extremo del robot respecto de un sistema de referencia fijo localizado en su base.

La representación y desarrollo del modelo cinemático se muestra en la figura 9.

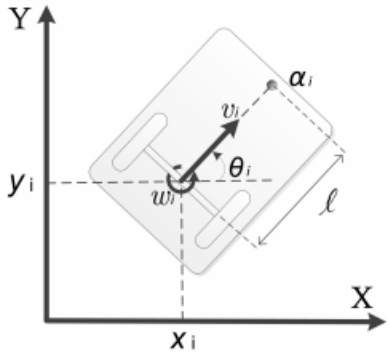


Figura 9 Representación del modelo cinemático de robot móvil

A partir de este diagrama se desarrolla el modelo matemático que se utilizara en nuestro sistema de control, tal como se muestra en la ecuación 2.

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} cos\,\theta_i & 0 \\ sin\,\theta_i & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \end{bmatrix}$$

Ecuación 2 Representación del modelo cinemático

Donde v_i y ω_i son la velocidad longitudinal y angular, respectivamente, del punto medio del eje de las ruedas robot. El sistema es no lineal entonces no puede ser

estabilizado por leyes de control continuas e invariantes en el tiempo. Por esta razón, se estudia la cinemática de un punto α_i que esta fuera del eje de las ruedas del robot. Las coordenadas del punto α_i están representadas por la ecuación 3:

$$\alpha_i = \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} = \begin{bmatrix} x_i + l\cos\theta_i, \\ y_i + l\sin\theta_i \end{bmatrix}$$

Ecuación 3 Representación de la coordenadas del punto αi

Donde ℓ es la distancia del centro del eje de las ruedas al punto frontal que se desea controlar y la cinemática de las nuevas coordenadas queda definida de la siguiente forma:

$$\dot{\alpha_\iota} = \begin{bmatrix} \dot{x}_i - \dot{\theta}_i l\cos\theta_i \\ \dot{y}_\iota + \dot{\theta}_i\, l\sin\theta_i \end{bmatrix}$$

Ecuación 4 Cinemática punto alpha

Sustituyendo en la ecuación anterior:

$$\dot{\alpha_\iota} = \begin{bmatrix} v_i\cos\theta_i - \omega_i l\sin\theta_i \\ v_i\sin\theta_i + \omega_i\, l\cos\theta_i \end{bmatrix} = A_i(\theta_i) \begin{bmatrix} v_i \\ \omega_i \end{bmatrix}$$

Ecuación 5 Modelo Cinemático basado en punto alpha

Donde:

$$A_i(\theta_i) = \begin{bmatrix} \cos(\theta_i) - l\sin\theta_i \\ \sin(\theta_i) \quad l\cos\theta_i \end{bmatrix}$$

Ecuación 6 Matriz de desacoplamiento

A_i Es llamada matriz de desacoplamiento, la cual es no singular dado que:

$$\det\left(A_i(\theta_i)\right) = l \neq 0$$

$$det\left(A_i\left(\theta_i\right)\right) = l \neq 0$$

Y su inversa está dada por:

$$A_i^{-1}(\theta_i) = \begin{bmatrix} cos(\theta_i) & sin(\theta_i) \\ -\frac{sin(\theta_i)}{l} & \frac{cos(\theta_i)}{l} \end{bmatrix}$$

Ecuación 7 Matriz Inversa de Desacoplamiento

Esto permite el control de posición del punto αᵢ para robots tipo uniciclo. Definiendo una variable auxiliar de control $u_i = (u_{i1}, u_{i2})^T$

$$\begin{bmatrix} v_i \\ \omega_i \end{bmatrix} = A_i^{-1}(\theta_i) \begin{bmatrix} v_{i1} \\ \omega_{i2} \end{bmatrix}$$

Ecuación 8 Modelo Cinemático Inverso

Se observa que el modelo cinemático de un robot móvil resulta en un sistema matricial no singular, que puede ser resuelto mediante al seleccionar un punto dentro de la geometría del robot móvil que comparta la direccionalidad del vector de velocidad lineal. Al seleccionar este punto auxiliar las nuevas ecuaciones son resolubles en términos de la velocidad lineal y angular, que era lo que se pretendía en primer lugar. Esta matriz de desacoplamiento sirve para proponer un control auxiliar que estará definido por la dinámica de la trayectoria a seguir.

La linealización Exacta o linealización Entrada/Salida (E/S), es una técnica madura para sistemas no lineales afines. Esta técnica cancela los términos no lineales de la

planta, usando una retroalimentación de estado no lineal, y produce una dinámica E/S lineal para la planta (Aranda, 2006). La linealización exacta ha sido utilizada ampliamente para el control de las trayectorias de un sistema no lineal y se ha implementado con éxito en el control de sistemas robóticos, funcionando satisfactoriamente en un amplio rango de actuación. El diseño del controlador se genera bajo el supuesto que conforme el tiempo avanza, la posición frontal del robot alcanza cada uno de los puntos deseados de una trayectoria deseada. Esto se expresa de la siguiente manera:

$$\lim_{t\to\infty}\left(\alpha_1 - m\,(t)\right) = 0$$

Ecuación 9 Error de posición conforme avanza el tiempo

Donde α_1 es la posición del punto frontal del robot móvil y $m(t) = [mx(t), my(t)]^T$ es la trayectoria deseada que es continuamente diferenciable. La variable auxiliar de control utilizada es:

$$u_1 = \dot{m}(t) - k_1\left(\alpha_1 - m(t)\right)$$

Ecuación 10 Señal de Control

A partir de esta variable auxiliar obtenemos la ley de control para la marcha:

$$\begin{bmatrix} v_1 \\ \omega_1 \end{bmatrix} A_1^{-1}\,(\theta_1)\,[\dot{m} - k_m\,(\alpha_1 - m)]$$

Ecuación 11 Control a través de velocidades

La convergencia a la trayectoria deseada se puede mostrar de la siguiente manera:

$$e_1 = \alpha_1 - m(t)$$

$$\dot{e}_1 = \dot{\alpha}_1 - \dot{m}(t) = u_1 - \dot{m}(t) = -k_1\left(e_1(t)\right)$$

Ecuación 12 Convergencia del Error

Por lo tanto, el error converge a cero exponencialmente.

Actualmente, los sistemas de simulación son una herramienta fundamental en el diseño y análisis de una gran variedad de procesos que involucren sistemas mecánicos en movimiento. El programa utilizado en este proyecto es Matlab® mediante su herramienta grafica a bloques Simulink®. Ambos sistemas permiten simplificar el complejo análisis matemático de un sistema robótico móvil.

Como herramienta de simulación Matlab/Simulink permite tener una interfaz simple pero muy compleja para realizar simulación de sistemas numéricos. Además, posee librerías de integración con la mayoría de las tarjetas de desarrollo.

En el modelo cinemático, las variables de control son u_1 y u_2. Sin embargo, la tracción y cambio de dirección del robot se obtienen por diferencia de velocidades entre las ruedas del robot, es decir, gracias a la diferencia de velocidades angulares derecha e izquierda, ω_d y ω_i. A continuación se analiza la relación de u_1 y u_2 con las velocidades angulares ω_d y ω_i . Para esto es necesario hacer algunas consideraciones pertinentes, las cuales son: que las ruedas motrices del robot son lo suficientemente rígidas (sin deformación) y que giran sin deslizarse sobre la superficie en la cual se mueven, y puesto que se empleara sólo el modelo cinemático, se considera también que el espacio de trabajo es perfectamente plano (sin inclinaciones), sólo existe un punto de contacto entre la rueda y el plano. De manera ideal la velocidad lineal v y la velocidad angular ω pueden ser consideradas como variables de control del sistema. Sin embargo, desde un punto de vista más

real, es necesario considerar que dichas variables son funciones de las velocidades angulares de las ruedas del robot.

Sean ω_d y ω_i las velocidades angulares de las ruedas derecha e izquierda del vehículo móvil, respectivamente. Entonces es posible mostrar que v y ω están relacionadas con ω_d y ω_i por medio de la siguiente transformación.

$$v = \frac{(\omega_d + \omega_i)r}{2}$$

$$\omega = \frac{(\omega_d + \omega_i)r}{2L}$$

Ecuación 13 Relación de velocidades

Estas ecuaciones pueden escribirse de forma matricial:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = T \begin{bmatrix} \omega_d \\ \omega_i \end{bmatrix}$$

Ecuación 14 Matriz de transformación

Donde:

$$T = \frac{r}{2} \begin{bmatrix} 1 & 1 \\ 1/L & -1/L \end{bmatrix}$$

Ecuación 15 Transformación T

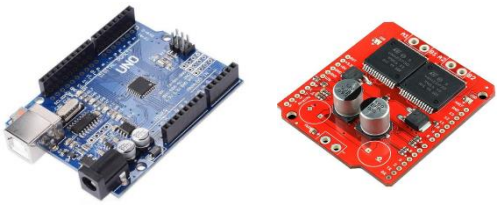
Puesto que la transformación T es no singular para todo r > 0, L > 0, cualquier valor de las velocidades v y ω pueden ser obtenidos mediante una adecuada selección de las variables ω_d y ω_i .

Se observa que, aunque los cálculos para el seguimiento de trayectoria están basados en las velocidades lineal y angular, es necesario traducir estas cantidades a magnitudes correspondientes a las velocidades angulares de cada una de las llantas. Esto eventualmente se traduce a un control de la velocidad de las llantas de manera individual a través de un sistema digital.

4.2.2 Diseño del robot móvil.

Descripción General

En esta etapa se analizan los aspectos funcionales para la construcción del robot móvil, como el tipo de locomoción, tipo de motores (motores de cc, servomotores, motores paso a paso), tipo de sensores y colocación de los mismos, principio de funcionamiento del algoritmo de control del robot, tipos de acción de control, etc., así como el ensamblaje de todos los elementos necesarios (motores, ruedas, tarjetas electrónicas) en el espacio disponible. Los elementos utilizados fueron: un sistema Arduino como tarjeta de control, el sistema de comunicación se compone de un módulo bluetooth HC05, la tarjeta controladora de motores es un *shield* basado en el chip VNH2SP30 diseñado por *Sparkfun.*



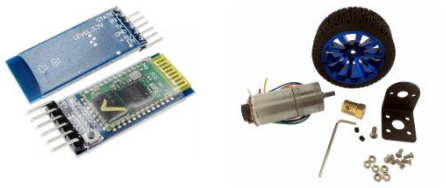


Figura 10 Componentes generales del robot móvil

Criterios de selección de elementos de elementos

Debido a la diversidad de dispositivos electrónicos que se relacionan con el diseño del robot móvil, la elección de los elementos puede resultar compleja. Sin embargo, se agrupan las principales características comunes de acuerdo a los objetivos de este proyecto para considerar para su elección, estas son:

Capacidad de los datos en el sistema: Velocidad de reacción y espacio suficiente para albergar los algoritmos desarrollados.

Eficiencia del sistema: Medida de la efectividad con la cual el sistema obtiene la respuesta deseada.

Proyección del sistema técnica y económica: Posibilidad de reusabilidad y modificación del sistema propuesto.

Capacidad de los datos en el sistema

Las capacidades del sistema se relacionan con sus especificaciones técnicas. Las especificaciones asociadas con el criterio de capacidad de los datos son:

- Memoria RAM de la tarjeta microcontroladora
- Métodos de comunicación
- Tasa de transmisión
- Herramientas de software disponibles

Eficiencia del sistema

La eficiencia del sistema se relaciona con la minimización de los costos, tanto económicos, de tiempo y técnicos. Las características relacionadas con el criterio de eficiencia del sistema son:

- Reducción del tiempo y del diseño de la ingeniería del proyecto
- Reducción de los costos del sistema
- Eficiencia de la comunicación de la red
- Reducción y simplificación del diseño del robot

Proyección del Sistema

Se refiere a las posibilidades de ampliación y actualización en el sistema actual, lo cual se traduce en un ahorro de costos futuros. Las características asociadas con el criterio de proyección técnica y económica del sistema son: ampliación del sistema y sistema abierto

De acuerdo a las características mencionadas anteriormente se realiza la búsqueda y selección de los dispositivos electrónicos teniendo en cuenta los requerimientos y

tratando de cubrir la definición dada por la intersección de los componentes mostrados en la figura 11.

Figura 11: Iteración de las características principales de los criterios de selección

La siguiente tabla muestra el criterio de selección de la tarjeta controladora:

Tabla 2 Criterio de selección de tarjeta controladora

Tarjeta Controladora	**Características Deseadas**	**Arduino Uno**	**Arduino Mega**	**Raspberry Pi**
Memoria	5 Kb estimados	***32 K**	256 K	SD Card,
Puertos	2	***2**	6	40 configurables
Pines de uso	4	***14**	54	40
Puertos PWM	2	***6**	15	1
Puertos Seriales	1	***1**	3	1
Costo		***$150**	$250	$1300

La siguiente tabla muestra el criterio de selección del Sistema de comunicación:

Tabla 3 Criterio de selección de sistema de comunicación

Sistema de Comunicación	**Características Deseadas**	**HC05**	**Tarjeta RF 433 MHz**	**Xbee**
Interfaz Serial	UART, 9600	***Nativo, max**	2400	Nativo,
Bidireccionalidad	Duplex	***Full duplex**	No	Full
Alcance	5 m	***10 m**	200 m	1000 m
Consumo de corriente	Menor a100mA	***40 mA, consumo**	40mA	50 mA
Costo		***$80**	$50	$600

Debido a la selección del motor JGA25-370 6V con Encoder, el análisis del sistema de potencia fue en base a las corrientes de funcionamiento: 3.2A corriente pico y 130 mA sin carga.

La siguiente tabla muestra el criterio de selección del Sistema de potencia:

Tabla 4 Criterio de selección del sistema de potencia

Sistema de Potencia	**Características deseadas**	**Adafruit Shield L293 v2**	**VNH2SP30 Dual Monster Motor Driver**
Numero de	2	2	***2**
Corriente Pico	3.2 A	0.6 A	***30 A**
Corriente Nominal	130 mA	600 mA	***6 A**
Control PWM	2	2	***2**
Control de	Si	Si	***Si**
Costo	Menor precio	$100	***$350**

En donde de acuerdo a las características deseadas la opción seleccionada es: VNH2SP30 Dual Monster**.**

Se observa, que, en el caso de la tarjeta controladora y el sistema de comunicaciones, la mayoría de las opciones cumple o excede los requisitos deseados, por lo que la decisión final es de carácter de costos. No así en el caso del

sistema de potencia, en el cual las características mínimas deseadas corresponden a la selección más costosa.

Especificaciones de componentes electrónicos

A continuación, se describen las características más importantes de los componentes electrónicos y mecánicos utilizados.

Tabla 5 Especificaciones Generales del Robot Móvil

Características	
Motores	Motorreductor JGA25-370
Controlador de Motores	Monster Motor Shield
Módulo de comunicación inalámbrica	HC-05 Módulo Bluetooth
Tarjeta controladora	Arduino uno
Largo	20 cm
Ancho	16 cm
Alto	15 cm
Distancia entre las ruedas	12 cm
Radio de las ruedas	3 cm

Motorreductor JGA25-370 6V con Encoder – 210RPM

Cuenta con un motor DC de 6V de alta potencia, una caja de engranes de 210RPM, con un eje de salida en forma de “D” de 4mm de diámetro. Tiene un decodificador en cuadratura o encoder con efecto hall que facilita la lectura de los pulsos por revolución. Es utilizado en diversas aplicaciones por su alta velocidad, es controlable el sentido de giro del eje al cambiar la polaridad entre sus dos conectores.

Tabla 6 Especificaciones Motorreductor JGA25-370

Características	
Voltaje nominal	12V
Voltaje de funcionamiento	6V-18V
Velocidad sin carga a 6V	210 RPM
Corriente de funcionamiento a 6V	480mA
Torque	0.95Kg-cm
Relación de engranaje:	35:1
Peso	85g
Tamaño del reductor	19mm

Monster Motor Shield

Esta es una versión mejorada del *Motor Driver Shield Ardumoto*. Para este *monster shield* se ha reemplazado el puente L298 H con un par de drivers de motor *full-bridge* VNH2SP30. También se ha mejorado el circuito de soporte por lo cual esta placa es capaz de manejar un par de motores de alta-corriente. El VIN y motor out están preparados para los terminales *screw* de 5mm, haciendo fácil conectar cables de gran grosor.

Cuando se utiliza en aplicaciones de alta corriente mayores a 14A, es necesario mejorar la resistencia térmica con un disipador de calor o ventilador y soldar los cables directamente a la placa en lugar de utilizar un terminal de tornillo.

Tabla 7 Especificaciones Motor Shield

Características	
Voltaje máximo	16V
Capacidad máxima de corriente	30 A
Capacidad continua de corriente	14A
Frecuencia de PWM máxima	20 Khz
Basado en circuito integrado	VNH2SP30
Pines analógicos	A2 y A3

Tarjeta controladora - Arduino uno

El Arduino es un sistema embebido, es decir, una combinación de componentes integrados en una sola placa que le dan funcionalidad a un circuito electrónico. Los sistemas embebidos como el Arduino están construidos en base a un microcontrolador, el cual almacena y ejecuta un algoritmo de programación.

Es una placa 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 USB, conector *jack* de alimentación, terminales para conexión ICSP y un botón de reseteo. Tiene toda la electrónica necesaria para que el microcontrolador opere, simplemente hay que conectarlo a la energía por el puerto USB o con un transformador AC-DC.

Tabla 8 Especificaciones Arduino Uno

Características	
Microcontrolador	ATmega328
Voltaje Operativo	5V
Voltaje de Entrada	7-12V
Pines de entradas/salidas digital	14 pines
Pines de entradas análogas	6 pines
Memoria Flash	32 KB (ATmega328)
SRAM	2KB (ATmega328)
EEPROM	1KB (ATmega328)

HC-05 Módulo Bluetooth maestro - esclavo

Este módulo permite establecer un enlace serial entre un microcontrolador y una PC u otro dispositivo habilitado con bluetooth (como un teléfono móvil). Tiene un header estándar de 0.1 pulgadas para alimentación y las señales del puerto serie. El módulo contiene un servicio de puerto serie (RFCOMM), creando un enlace de datos transparente entre una PC, Celular, Tablet o cualquier dispositivo con bluetooth maestro y el microcontrolador. La salida del módulo es una señal serial asíncrona a 9600 baudios que puede ser recibida e interpretada fácilmente por cualquier microcontrolador. Funciona como dispositivo maestro y esclavo, configurable mediante comandos AT, distancia de hasta 10 metros en condiciones óptimas del ambiente

Tabla 9 Especificaciones HC-05 Módulo Bluetooth

Características	
Frecuencia	2.4GHz
Modulación	GFSK
Alcance	5m a 10m
Sensibilidad	<-84dBm
Voltaje de	3.6- 6V
Consumo de corriente	50mA
Potencia de emisión	<4dBm, Clase2
Dimensiones	1.7cm x 4cm aprox)

Tabla 10 Especificaciones Fuente de Alimentación

Características	
Voltaje	5-18V
Peso	230g
tamaño	10*3.1*3.3
Tipo de conector	XT60

Una vez seleccionados los componentes, se dispuso a seguir la metodología mostrado en el diagrama de la figura 9. Herramientas de diseño 3D como *Google SketchUp,* permiten a su vez tener una idea conceptual de la distribución de los componentes. Entre sus ventajas, es que su versión gratuita permite usar herramientas avanzadas para crear y editar sólidos, escalar modelos, rotación de objetos, creación de textos 3D, además se puede trabajar con componentes reutilizables para el diseño de formas más complejas.

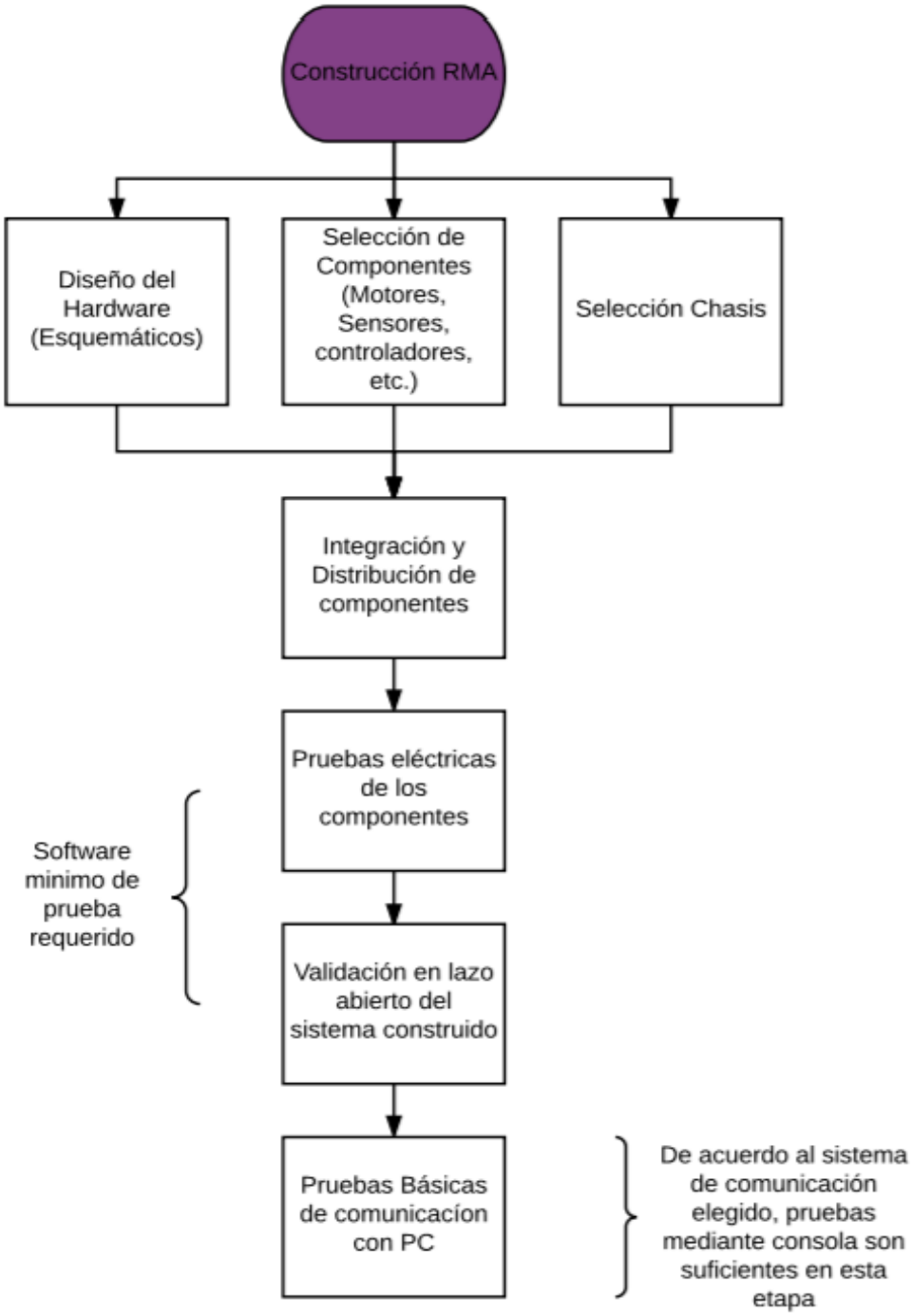


Figura 12 Metodología construcción robot móvil

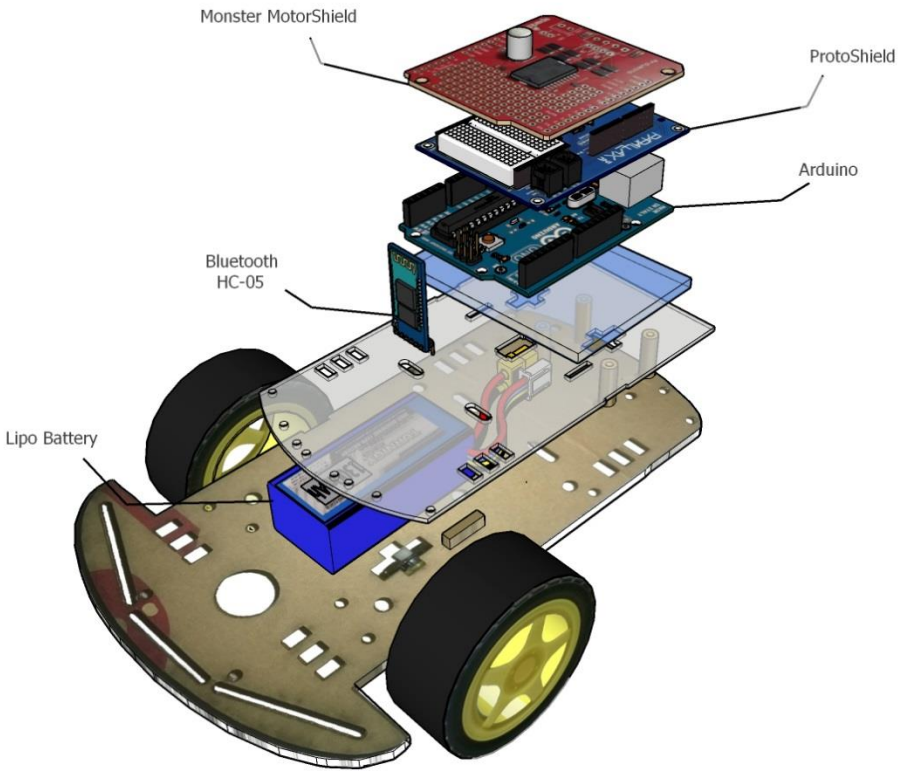


Figura 13 Diseño de robot móvil - Vista *exploded*

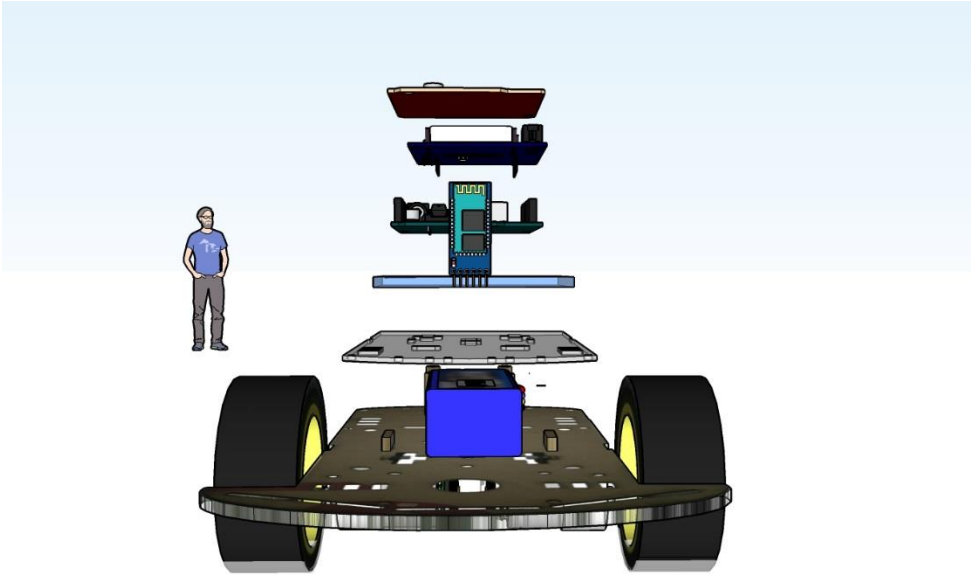


Figura 14 Diseño de robot móvil - Vista Frontal

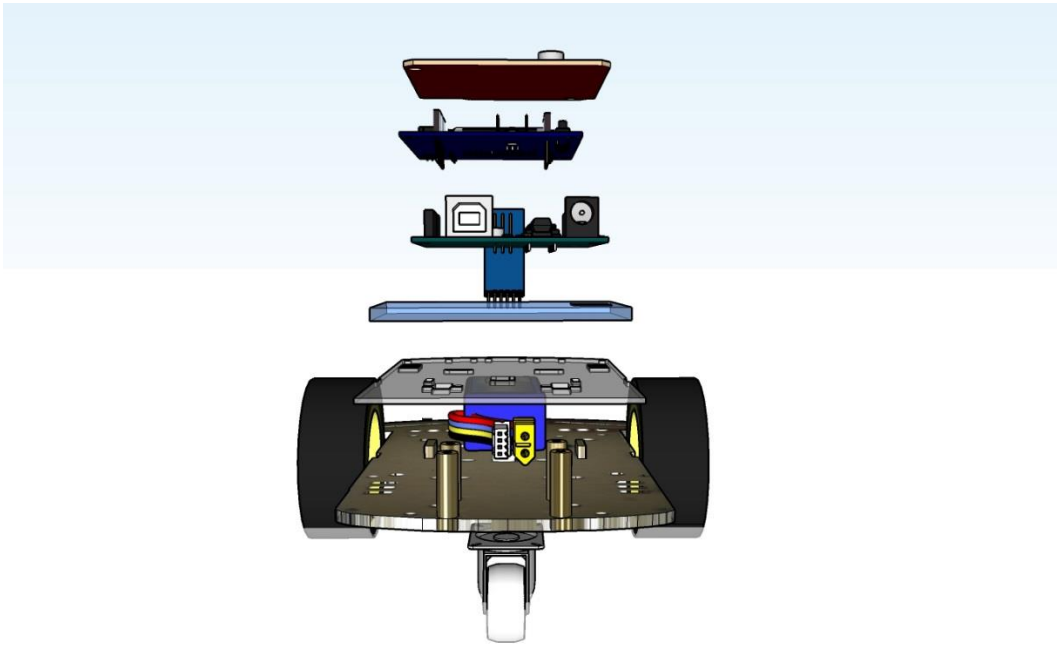


Figura 15 Diseño de robot móvil - Vista Posterior

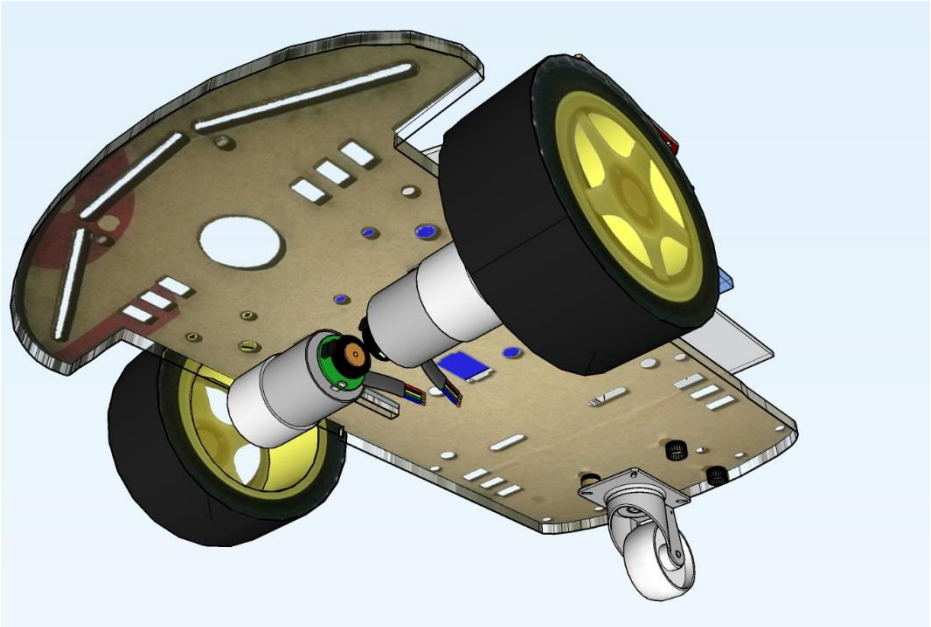


Figura 16 Diseño de robot móvil - Vista Inferior

4.2.3. Diseño de Software

4.2.3.1. Diseño de Software embebido

El diagrama de secuencia mostrado en la figura 17 representa, a alto nivel, los componentes envueltos en el desarrollo de la arquitectura, indica la interacción de los sistemas de software tanto de la interfaz gráfica, como del sistema embebido. Cabe señalar que cada subsistema tiene su propia implementación y arquitectura modular.

Debido a la facilidad de portabilidad, el ambiente seleccionado para la elaboración de la interfaz fue la suite Visual Basic .NET 2015.

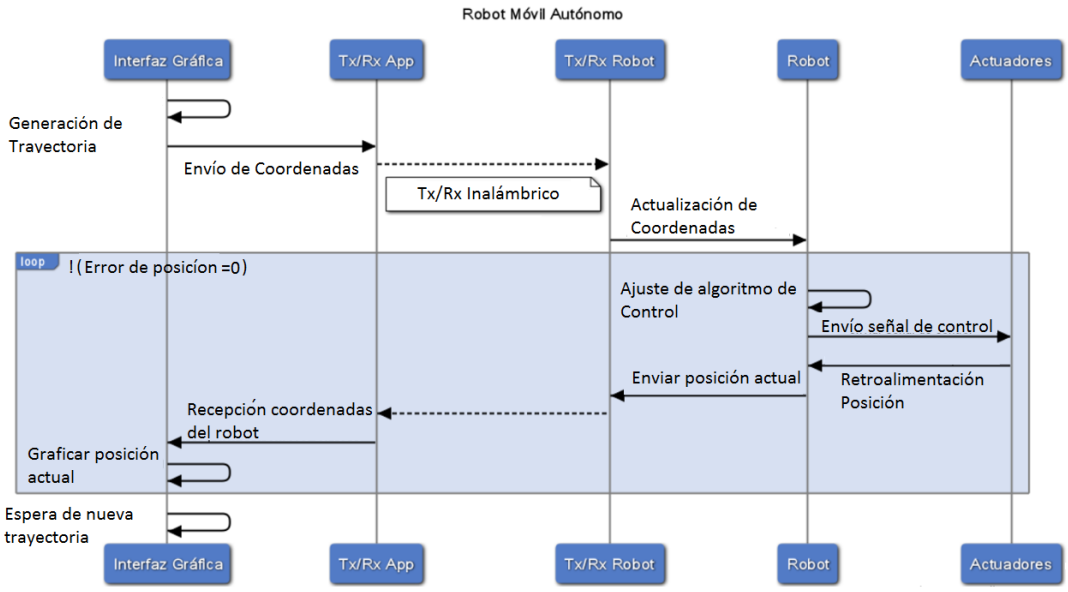


Figura 17 Metodología diseño de software

4.2.3.2 Diseño de Software de Visión Artificial

El procesamiento digital de imágenes (PDI) se encarga de procesar las imágenes del mundo real de manera digital por medio de un computador. Es un tema amplio, en el

que se incluyen estudios de física, matemáticas, ingeniería eléctrica, computación. Estudia los fundamentos conceptuales de la adquisición y despliegue de imágenes y con detalle los fundamentos teóricos y algorítmicos del procesamiento como tal. Tiene, además, como objetivo mejorar el aspecto de las imágenes y hacer más evidentes en ellas ciertos detalles que se desean hacer notar.

La Visión Artificial (o visión computacional) puede ser definida como los procesos de obtención, caracterización e interpretación de información de imágenes tomadas de un mundo tridimensional. Estos procesos pueden ser subdivididos en seis áreas principales y están agrupados de acuerdo a la complicación y delicadeza que lleva su implementación (Granados y Velásquez, 2015).

La captura o adquisición es el proceso a través del cual se obtiene una imagen digital utilizando un dispositivo de captura como una cámara digital, video cámara, escáner, satélite, etc.

El preprocesamiento incluye técnicas tales como la reducción del ruido, realce del contraste, realce de ciertos detalles, o características de la imagen.

La segmentación es el proceso que divide una imagen en objetos que sean de nuestro interés de estudio.

La descripción es el proceso que obtiene características convenientes para diferenciar un tipo de objeto de otro, como: la forma, el tamaño, área, etc.

El reconocimiento es el proceso que identifica los objetos, como, por ejemplo: una llave, un tornillo, moneda, coche, etc.

La interpretación es el proceso que asocia un significado a un conjunto de objetos reconocidos (llaves, tornillos, herramientas, etc.) y trata de emular la cognición.

MATLAB dispone también de una amplia gama de programas de soporte especializados, denominados *Toolboxes*, que extienden significativamente el número de funciones incorporadas en el programa principal. Estos *Toolboxes* cubren casi todas las áreas principales en el mundo de la ingeniería y la simulación, destacando entre ellos: procesamiento de imágenes, procesamiento de señales, control robusto, estadística, análisis financiero, matemática simbólica, redes neuronales, lógica difusa, identificación de sistemas, simulación de sistemas dinámicos, Simulink, etc. Para este caso de visión artificial, el *Image Processing Toolbox* proporciona a Matlab un conjunto de funciones que amplían las capacidades del producto para realizar desarrollo de aplicaciones y de nuevos algoritmos en el campo del procesamiento y análisis de imágenes. Algunas de las funciones más importantes son:

- Análisis de imágenes y estadística.
- Diseño de filtros y recuperación de imágenes.
- Mejora de imágenes.
- Operaciones morfológicas.
- Definición de mapas de colores y modificación gráfica.
- Operaciones geométricas.
- Transformación de imágenes.

El presente proyecto de tesis consiste en la detección de objetos a color, utilizando las imágenes obtenidas en tiempo real con una cámara web, permitiendo realizar el procesamiento de imágenes para marcar cada uno de los objetos con su respectivo centroide (coordenadas de un punto central promedio), con el objetivo de indicar de manera visual y física que ha detectado algún color especificado. La implementación de este sistema está sujeto a ambientes controlados en donde los objetos a detectar posean matices con nitidez alta.

En la figura 18 se muestra el diagrama de flujo representativo del proceso de captura de imágenes del video en tiempo real y su correspondiente tratado de fotogramas.

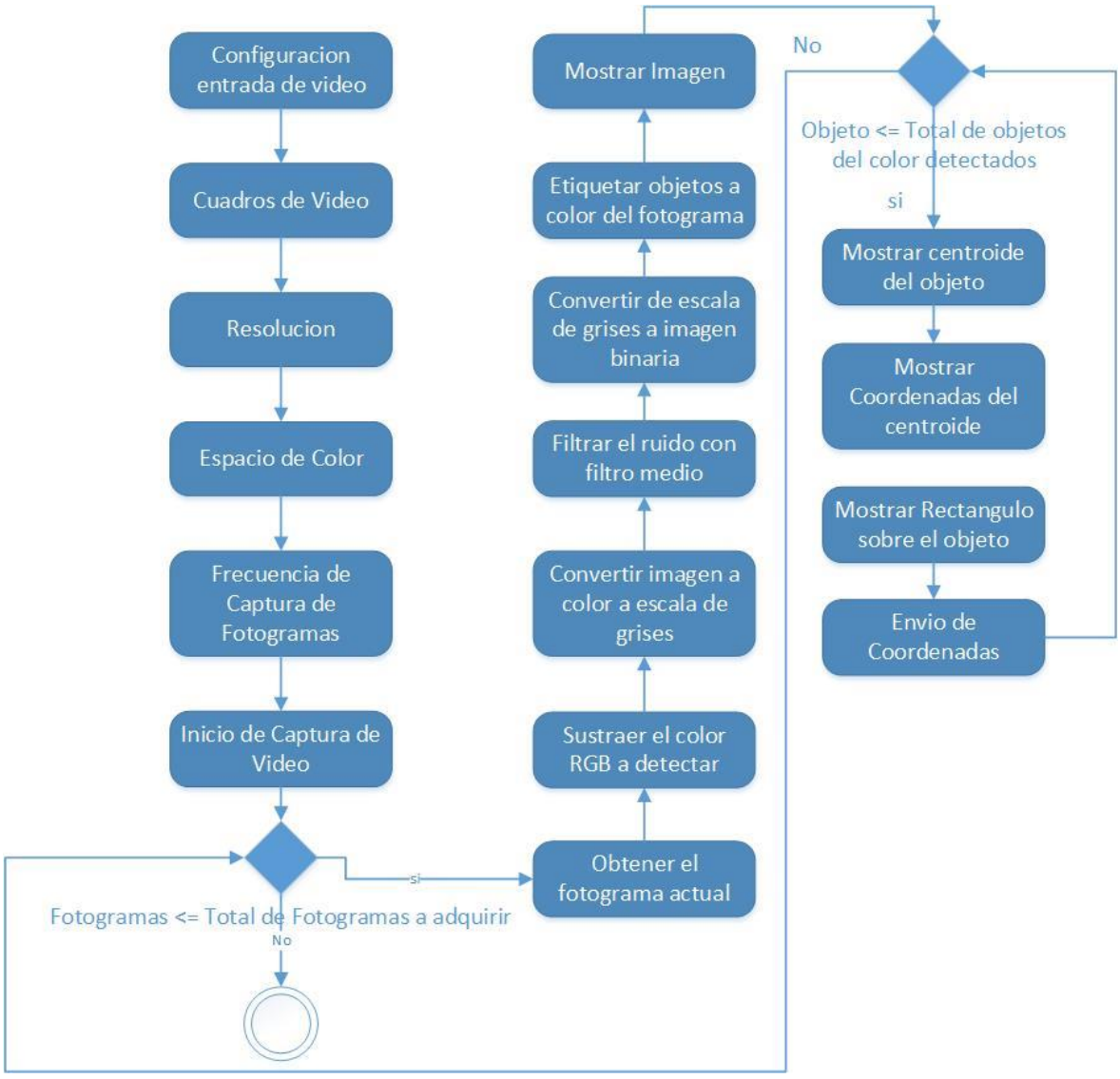


Figura 18 Proceso detección de color

Al poder detectar estas coordenadas mediante estas herramientas de visión artificial, se puede tener una representación prácticamente en tiempo real del comportamiento del modelo cinemático, además que el *workspace* de Matlab nos permite observar los valores deseados mientras el sistema se encuentra en movimiento.

4.3 Implementación y desarrollo

La integración de todos los componentes de la etapa del diseño del proyecto tiene como resultado final un robot móvil cuya trayectoria de transito podrá ser modificada a través de una interfaz gráfica.

Un diagrama de la integración completa del sistema se muestra en la figura 19. El proceso inicia con el usuario diseñando la trayectoria deseada (1), la interfaz tiene la capacidad de enviar las coordenadas a través del puerto serial o de generar un archivo de valores separados por coma (.csv), en ambas formas los datos corresponden a los puntos generados en la interfaz gráfica. Estas dos opciones permiten hacer un procesamiento embebido (en el microcontrolador del robot) o centralizado (en una computadora de uso general) de manera respectiva. Por la complejidad de los cálculos matriciales, el tipo de procesamiento seleccionado es el centralizado, esto significa que las coordenadas se guardan en un archivo, el cual es leído posteriormente mediante un comando de Matlab (2). Una vez obtenidos los datos, estos se vectorizan y se guardan en arreglos matriciales, los pasos siguientes consisten en inicializar los puertos hardware que se utilizaran como sistemas de detección (cámara web) y comunicación (puerto serial) con el robot móvil (3). Una vez inicializado el sistema de visión artificial, los algoritmos de detección (**¡Error! No se encuentra el origen de la referencia.**2) permiten localizar al robot en el área de trabajo designada (3.1). Posteriormente, una vez obtenidos los datos actuales de la posición del robot, estos sirven como la retroalimentación del sistema de control (3.2). Una vez calculados los valores de velocidad angular de cada una de las llantas, estos son enviados a través del puerto serial (4) y transmitidos de manera inalámbrica mediante bluetooth hacia el robot móvil (5).

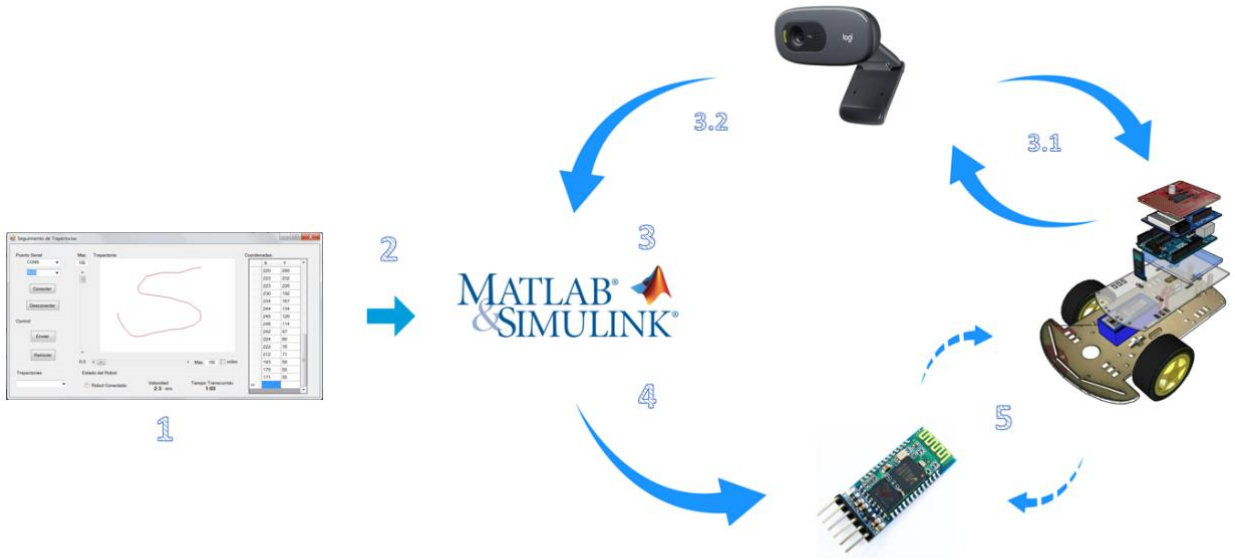


Figura 19 Proceso completo de generación de trayectorias a través de GUI

La integración de todos los componentes de la etapa del diseño del proyecto tiene como resultado final un robot móvil cuya trayectoria de transito podrá ser modificada a través de una interfaz gráfica. En la figura 20 se muestra el diseño de la interfaz gráfica con sus respectivos comandos.

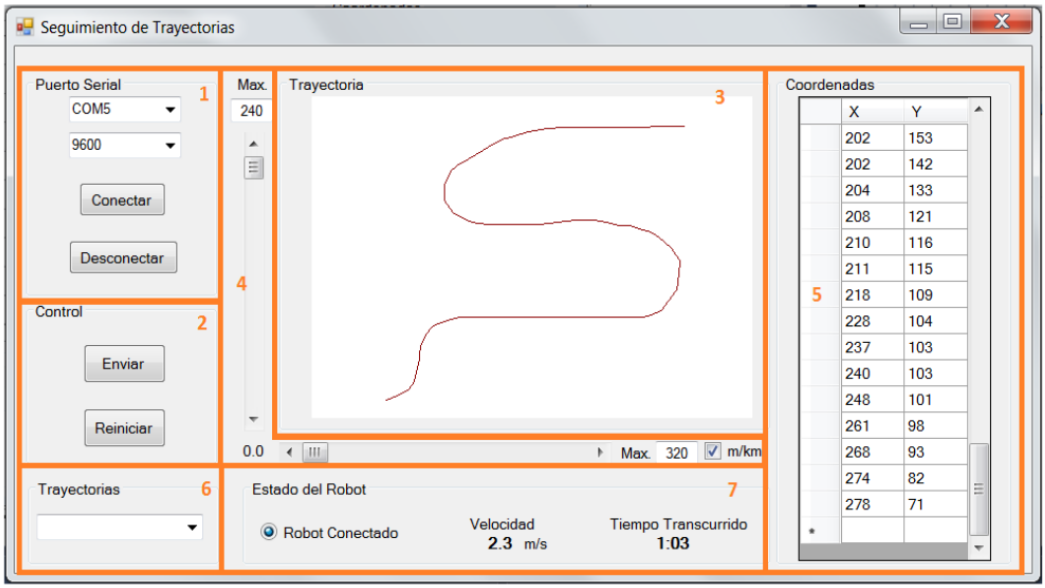


Figura 20 Diseño final de interfaz gráfica de usuario

1. Panel de Conexión
2. Panel para envió y reinicio de datos
3. Panel grafico para diseño de la interfaz
4. Ajuste de unidades y escala de la trayectoria diseñada
5. Tabla con cada una de las coordenadas
6. Trayectorias predefinidas
7. Estado y métricas del robot

El primer paso en MATLAB, para utilizar una cámara web en la detección de objetos en tiempo real es, asignar el formato de la cámara, al igual que la captura de los cuadros que corren en tiempo real y su resolución (en este caso YUY2 COM 620X 480 pixeles).

En este punto ya se cuenta con una imagen (fotograma) procesada en el cual se encuentran detectados solamente los objetos del color que fue configurado a detectar. Continúa con el marcado de cada uno de los objetos detectados anteriormente (proporcionados por el método de *regionprops)*, donde se localizan su centroide, las coordenadas del objeto (base a su centroide) en la imagen y se marca en un rectángulo del mismo color en el cual fue configurado a detectar.

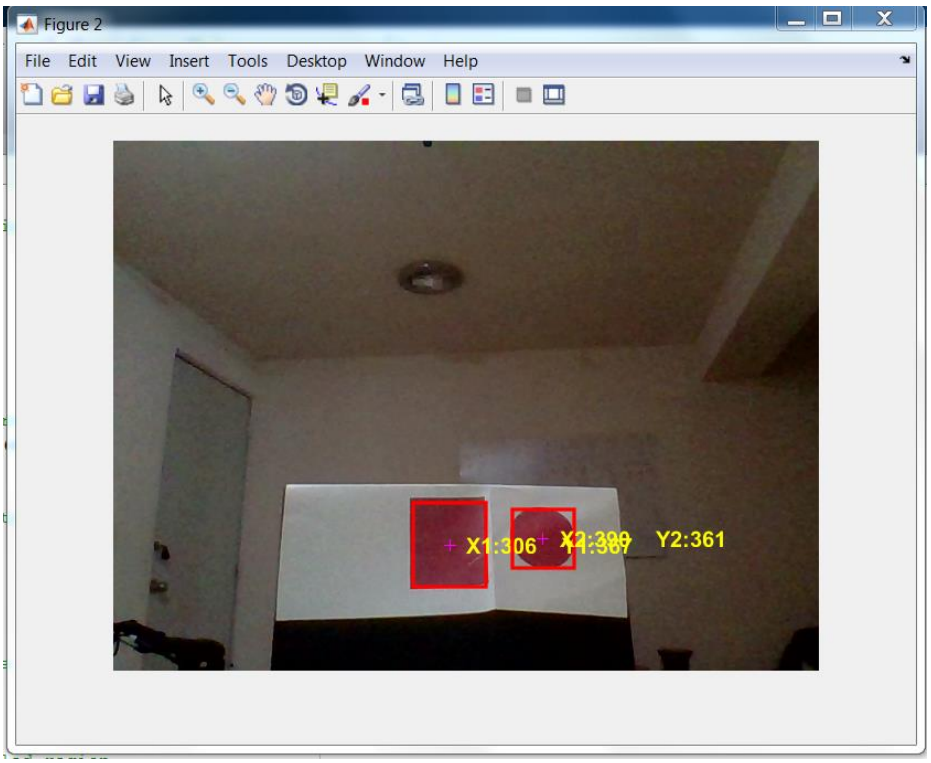


Figura 21 Detección de color

4.4 Pruebas

4.4.1 Pruebas de Simulación

Simulink® es otro de los productos desarrollados por *MathWorks*. Consiste en un entorno de programación grafica para el modelado, simulación y análisis de sistemas dinámicos. Simulink se integra con MATLAB de manera que se pueden aprovechar de manera conjunta y bidireccional las características de ambos entornos. Por sus características, el uso de Simulink está muy extendido en aplicaciones relacionadas con el procesamiento digital de señales, la ingeniería de control, la robótica o el diseño basado en modelos entre otras. Los modelos Simulink consisten en diagramas de bloques. Cada bloque es tratado como una caja negra que realiza alguna operación. Simulink proporciona numerosos bloques predefinidos que realizan una gran variedad de funciones diferentes. Además, en un modelo es posible incluir subsistemas dentro de bloques, creando de este modo estructuras jerárquicas de bloques anidados y facilitando la creación de modelos complejos. El comportamiento dinámico de los modelos generados puede simularse y analizarse.

Simulink no sólo utiliza bloques para definir el comportamiento del sistema. También se pueden incorporar algoritmos descritos mediante el lenguaje propio de MATLAB o en otros lenguajes de programación, como C o C++. Es posible incluso generar código directamente a partir del modelo creado en Simulink. Por otro lado, Simulink también permite conectar el modelo con hardware con el fin de realizar pruebas en tiempo real.

El siguiente modelo es utilizado para simular mediante la herramienta Simulink el sistema robótico. El sistema, que se observa en la figura 22, es un sistema en lazo cerrado que contiene las partes descritas previamente.

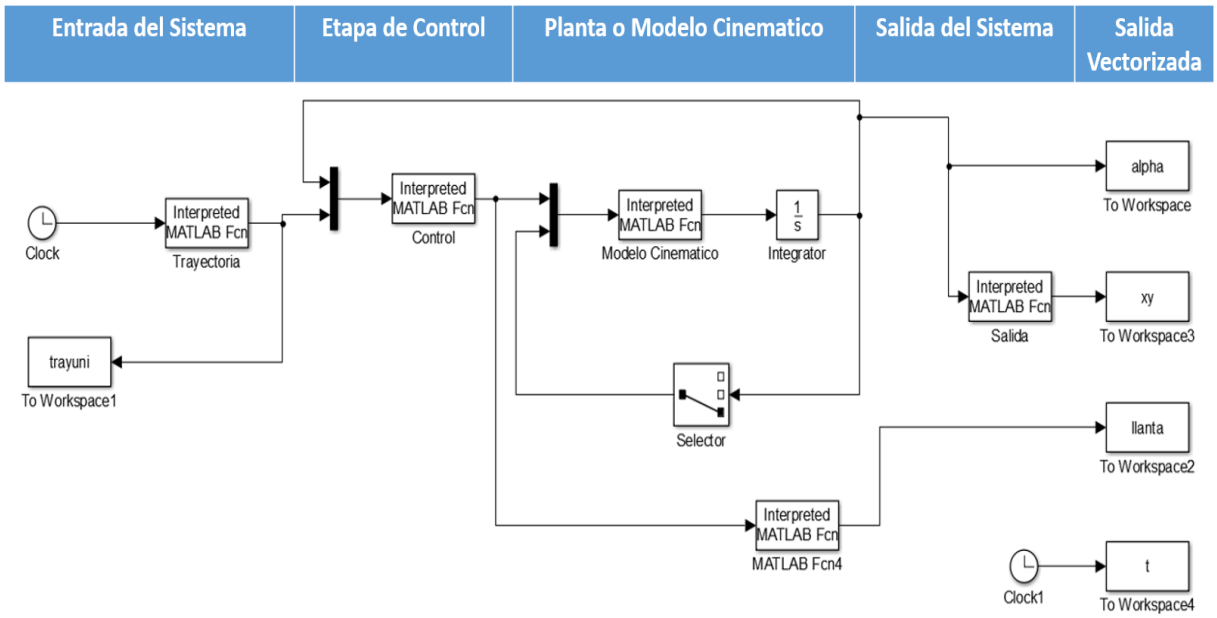
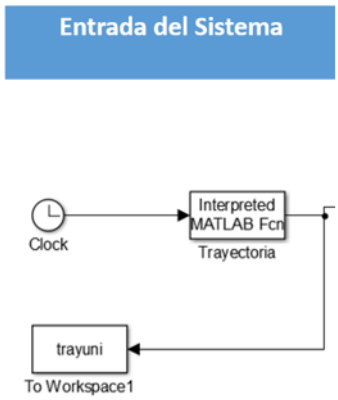


Figura 22 Bloques modelo cinemático

La entrada a este sistema se conforma a través de un vector de datos que contiene las coordenadas previamente generadas mediante la interfaz gráfica. La simulación consiste en resolver el sistema mediante los algoritmos de punto fijo de Simulink, la prueba corre durante 10 segundos a un periodo de muestreo de 0.01 segundos. Al término de la prueba, los resultados vectorizados son enviados al worskpace de Matlab, un script toma estos datos calculados y los procesa para generar resultados gráficos y entendibles para el usuario.

4.4.1.1 Entrada del sistema

En esta primera etapa de la simulación se encuentra definida la trayectoria seleccionada, los datos en este caso son una representación parametrizada de una ecuación o conjunto de ecuaciones que generan coordenadas en un sistema vectorial.

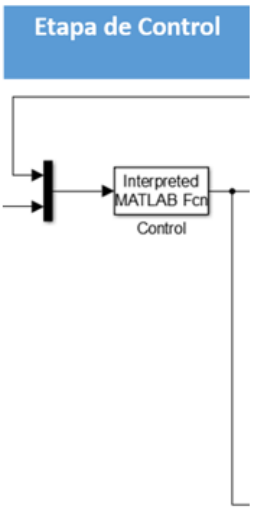



```
function xy=tray_uni(u)

T=u(1);

thdp = 0;
if (T <= 3.33)
xd =9+ 33.03*T;
yd =11+ 88.58*T;
xdp = 33.03;
ydp = 88.58;
thd = pi*(69.55)/180;
elseif (T <= 6.66)

xd = 119 + 33.03*T/2;
yd = 306 - 88.58*T/2;
xdp =33.03*.5;
ydp =-.5*88.58;
thd = pi*(-71.27)/180;
elseif(T <= 9.99)

xd = 229 - 33.03*(2*T/3);
yd =11;
xdp =-33.03*(2/3);
ydp =0;
thd = pi*(179)/180;
else
xd = T;
yd =1;
xdp =1;
ydp =0;


end

xy(1)=xd;
xy(2)=yd;
xy(3)=xdp;
xy(4)=ydp;
xy(5)=thd;
xy(6)=thdp;
```

Figura 23 Entrada y lógica del sistema

4.4.1.2 Sistema de Control

La figura 24 muestra la simulación del bloque y la codificación del sistema de control definido en la ecuación 10. Se observa que la señal de control es nuevamente un sistema vectorizado.

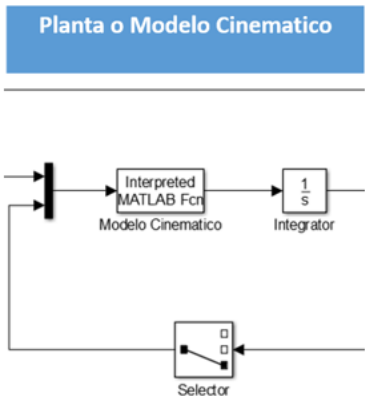



```
function Velocidad=control_uni(u)
x=u(1);
y=u(2);
th=u(3);
xd=u(4);
yd=u(5);
xdp=u(6);
ydp=u(7);
thd=u(8);
thdp=u(9);

k1=2;
k2=2;

l=0.16;

A=[cos(th) -l*sin(th);
    sin(th) l*cos(th)];

u1 = xdp-k1*(x-xd);
u2 = ydp-k2*(y-yd);

U=[u1;u2];

V=(inv(A))*U;


Velocidad(1)=V(1);
Velocidad(2)=V(2);
```

Figura 24 Simulación del bloque del sistema de Control

4.4.1.3 Modelo Cinemático

Las velocidades y la orientación son las entradas del modelo cinemático. A través de ecuaciones matriciales, estas operaciones dan como resultado la cinemática del punto frontal.

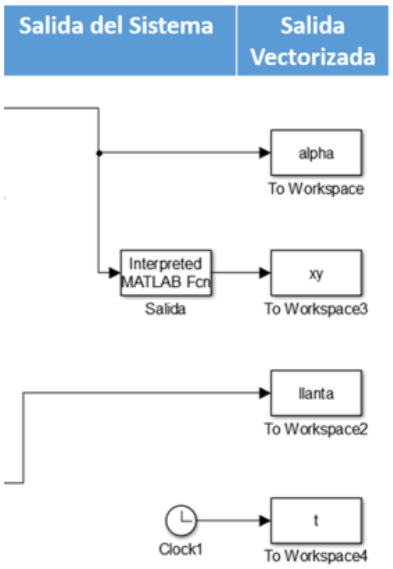



```
function sal=plant_uni(u)

v=u(1);
w=u(2);
th=u(3);
r=.05; % radio de las llantas

l=0.16;
A=[cos(th) -l*sin(th);
    sin(th) l*cos(th)];

alphap=A*[v;
          w ];

thp=w;

sal(1)=alphap(1);
sal(2)=alphap(2);
sal(3)=thp;
```

Figura 25 Simulación y codificación modelo cinemático

4.4.1.4 Salida del Sistema

Para facilitar el tratamiento de los datos, el resultado de la simulación es enviado al *workspace* de Matlab. Esto permite generar *scripts* que permitan manejar los datos y realizar operaciones posteriores, tales como gráficas y simulaciones animadas.




```
function toworkspace=alphas_uni(u)


a1=u(1);
a2=u(2);


l=0.5;

th=u(3);


x=a1-l*cos(th);
y=a2-l*sin(th);


toworkspace(1)=x;
toworkspace(2)=y;
toworkspace(3)=th;
```

```
function llanta=vel_llantas(u)
v=u(1);
w=u(2);

r=.05; % radio de las llantas

l=0.16;


W=(1/r)*([1 l;1 -l]*[v;w]);
llanta(1)=W(1);
llanta(2)=W(2);
```

Figura 26 Simulación y codificación de la salida del sistema

4.4.2 Pruebas Funcionales

Una de las partes fundamentales de este proyecto de tesis es la realización del trazado de las trayectorias deseadas mediante la interfaz gráfica. El algoritmo diseñado se puede observar en la figura 27.

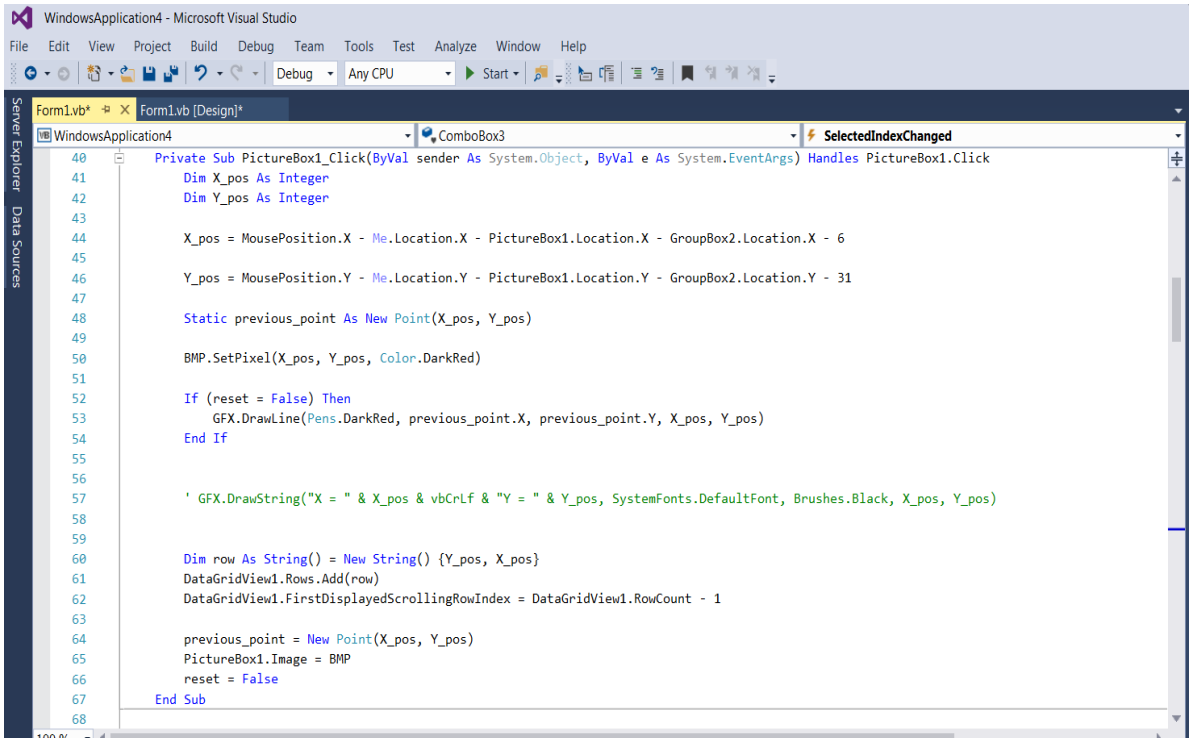


Figura 27 Algoritmo de trazado de trayectorias

El diseño de este algoritmo consiste en un análisis incremental de los puntos previos seleccionados mediante el puntero del mouse. Este algoritmo permite caracterizar los pixeles seleccionados en un panel gráfico y unir estos puntos mediante líneas. Posteriormente, estas recolecciones de puntos son convertidos en coordenadas que serán utilizadas como las entradas de nuestro sistema robótico.

Las pruebas funcionales son ejecutadas con el sistema diseñado mediante retroalimentación de la posición por visión artificial programada en Matlab. El algoritmo de control es programado para generar las velocidades angulares de cada

una de las llantas ω_d , ω_i. El sistema robótico diseñado mediante un control proporcional (retroalimentación por encoder óptico) ajusta la velocidad deseada.

4.5 Resultados de las pruebas

4.5.1 Simulación

Las imágenes de las siguientes figuras corresponden a los resultados de una simulación del comportamiento del robot al seguir una trayectoria de tipo triangular.

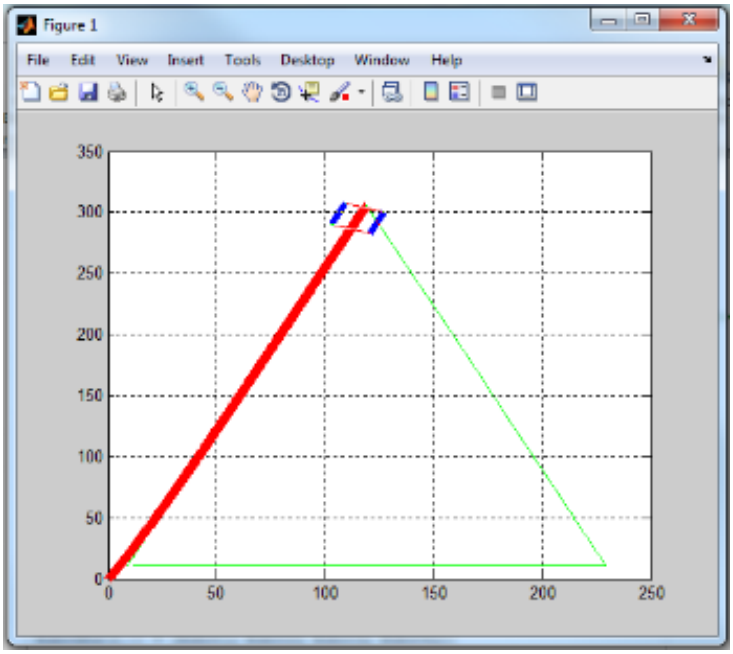


Figura 28 Transición de seguimiento de trayectoria 1

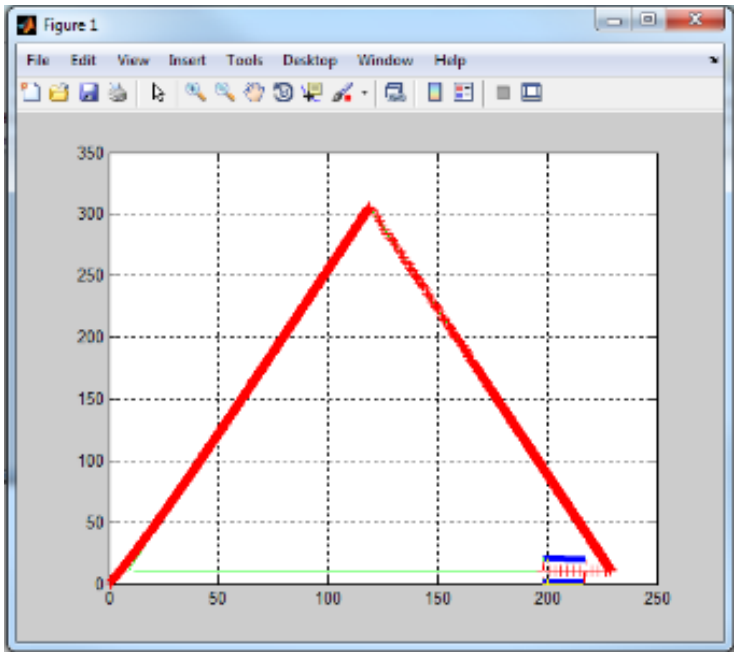


Figura 29 Transición de seguimiento de trayectoria 2

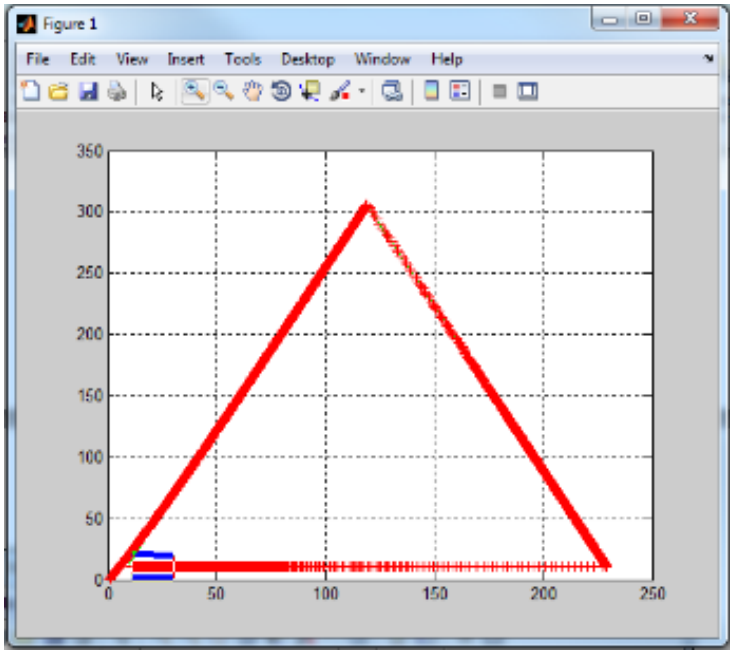


Figura 30 Transición de seguimiento de trayectoria 3

Las siguientes graficas muestran el comportamiento de las variables de interés, el punto central, la orientación, el punto alfa y finalmente los errores de posición respectivamente.

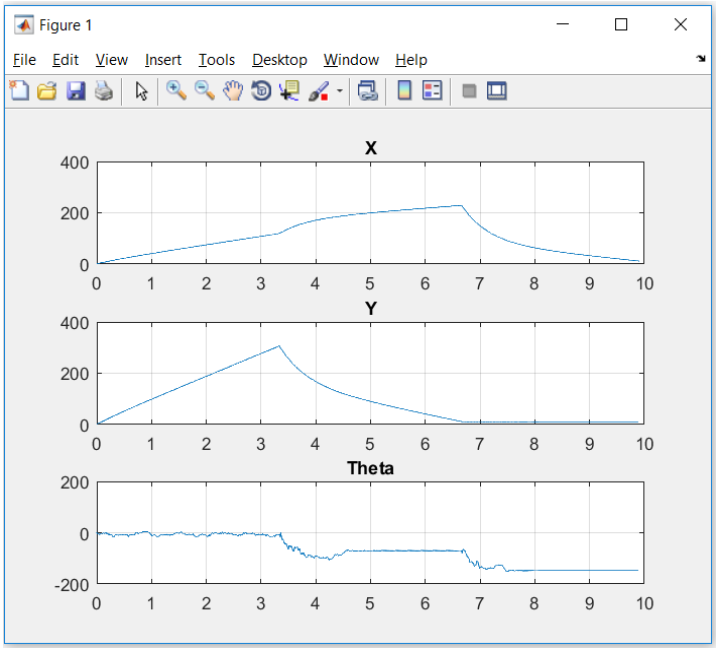


Figura 31 Comportamiento del punto central del robot

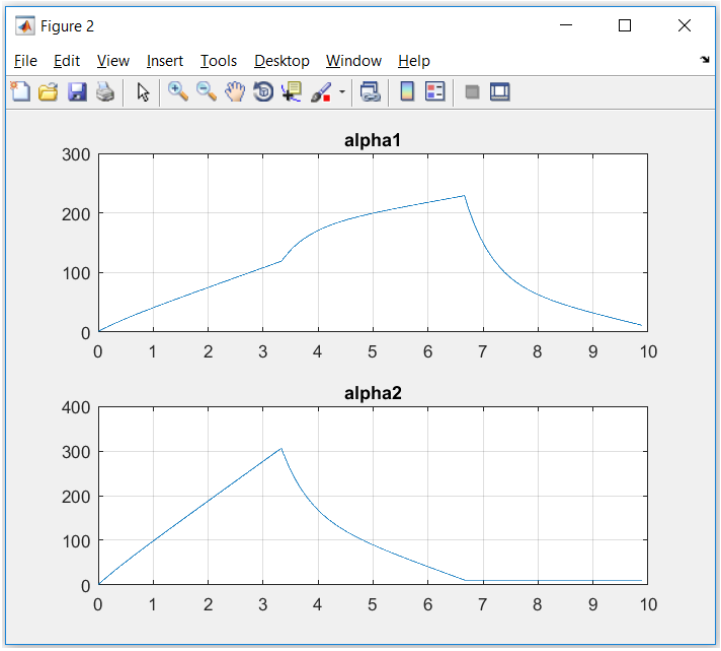


Figura 32 Comportamiento del punto frontal del robot

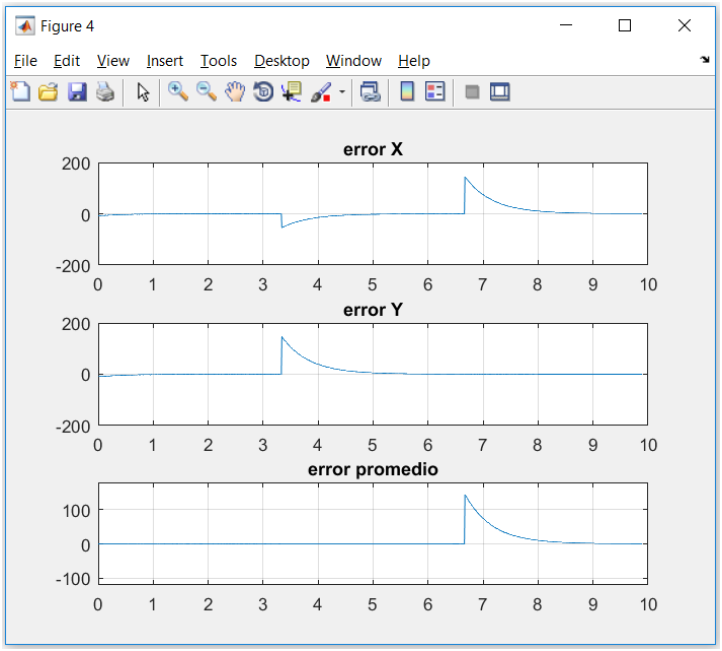


Figura 33 Comportamiento de los errores de posición

De la figura 33, se observa que el error, tanto en los vectores x y y como el error promedio cuadrático, mantienen un valor cercano al cero aun considerando las variaciones que contiene la trayectoria.

4.5.2 Pruebas funcionales

La figura 34 muestra el comportamiento del sistema robótico al seguir una trayectoria semicircular (marcador verde). Considerando el desempeño mínimo de los encoders ópticos, los resultados demuestran que el control por linealización exacta provee resultados satisfactorios, inclusive cuando el sistema a controlar posee componentes no optimizados.

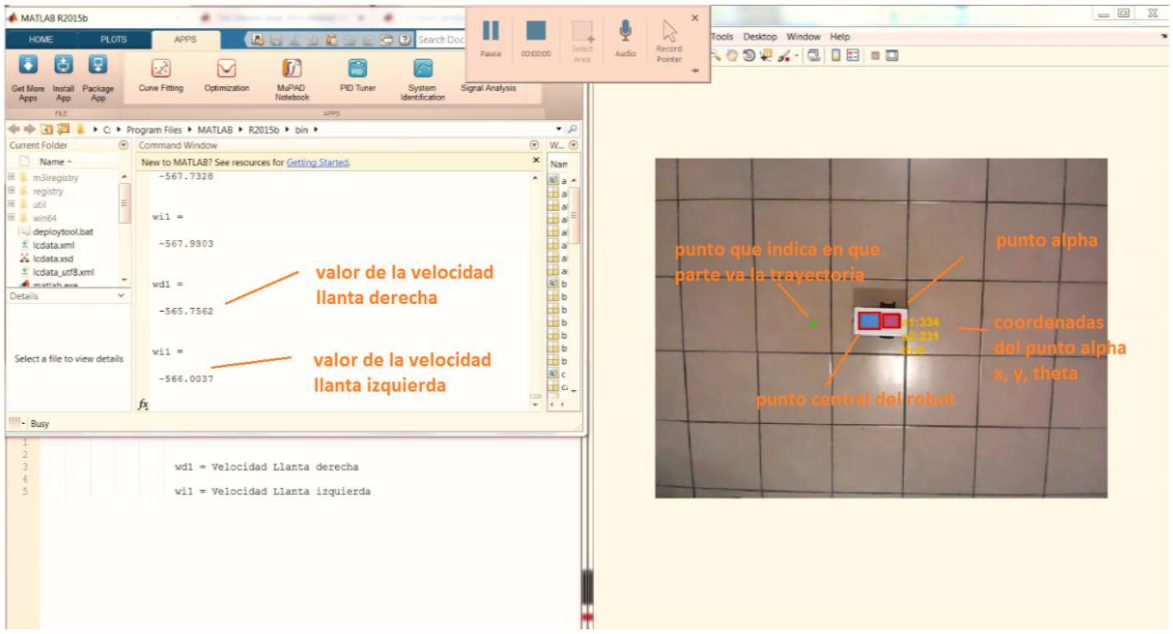


Figura 34 Seguimiento de trayectoria del sistema robótico 1

En la figura anterior se puede observar el inicio de nuestra prueba, mediante el *workspa*ce de Matlab se observa el comportamiento de las variables de interés, en este caso las velocidades angulares de cada una de las llantas. La magnitud y el signo de estas variables dependerán del valor de la señal de error en ese instante de tiempo. Para este primer caso, el valor de ambas velocidades son lo suficientemente grandes, lo que implicara una respuesta rápida (aceleración) del sistema robótico, el signo indica la dirección de giro.

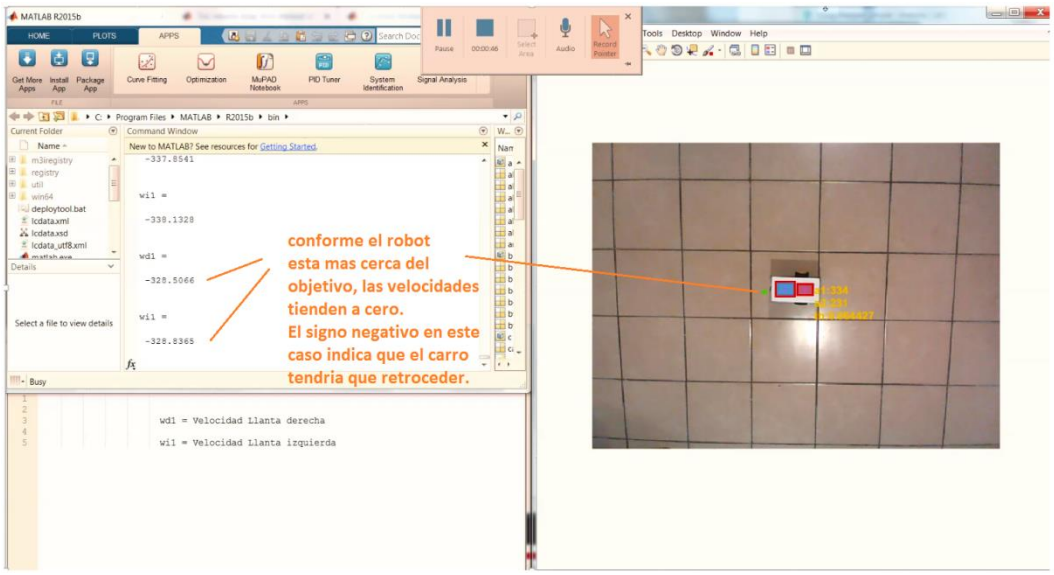


Figura 35 Seguimiento de trayectoria del sistema robótico 2

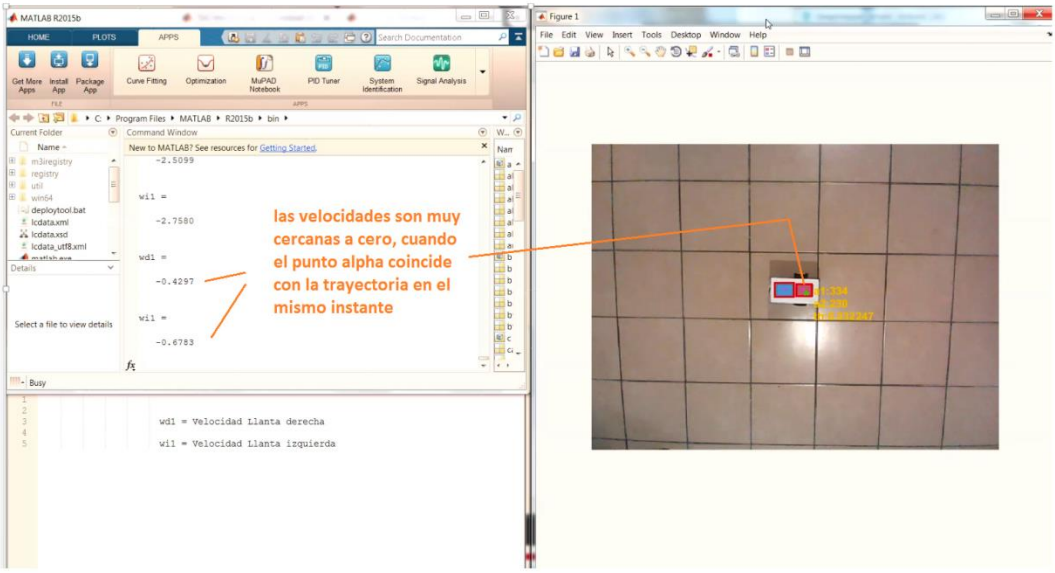


Figura 36 Seguimiento de trayectoria del sistema robótico 3

Figura 36 muestra el comportamiento del sistema cuando el error es próximo a cero, en este caso las velocidades angulares son tan pequeñas que el sistema robótico estará en un estado momentáneo de aparente reposo. Figura 37, el caso en que nuevamente, al ser un sistema de seguimiento, el nuevo punto de referencia habrá cambiado, por lo tanto, el sistema de control realizara un reajuste de las variables, y así mismo de las nuevas velocidades. En este caso, el signo de las velocidades angulares ha cambiado, en consecuencia también el sentido del movimiento.

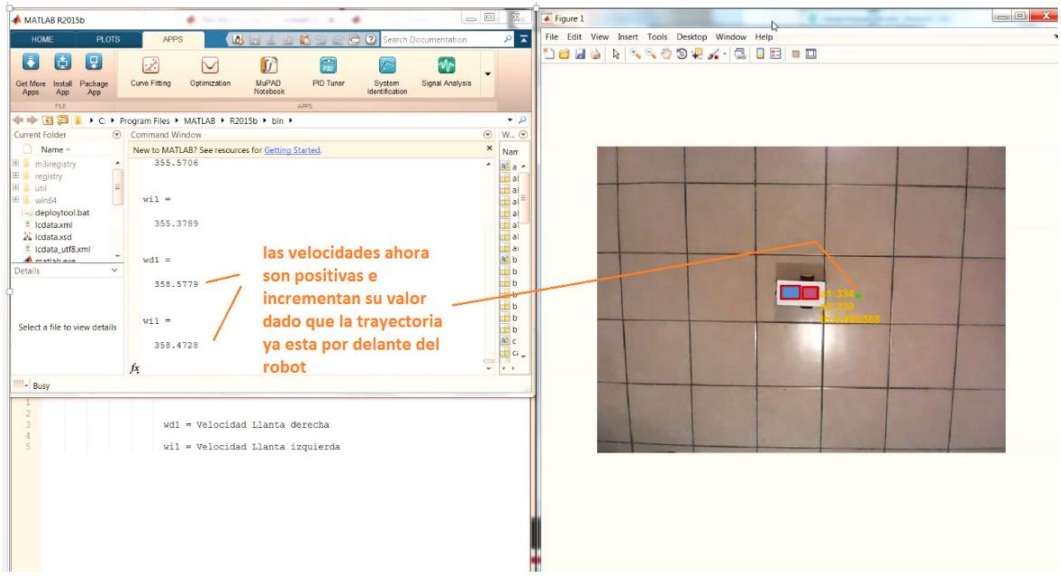


Figura 37Seguimiento de trayectoria del sistema robótico 4

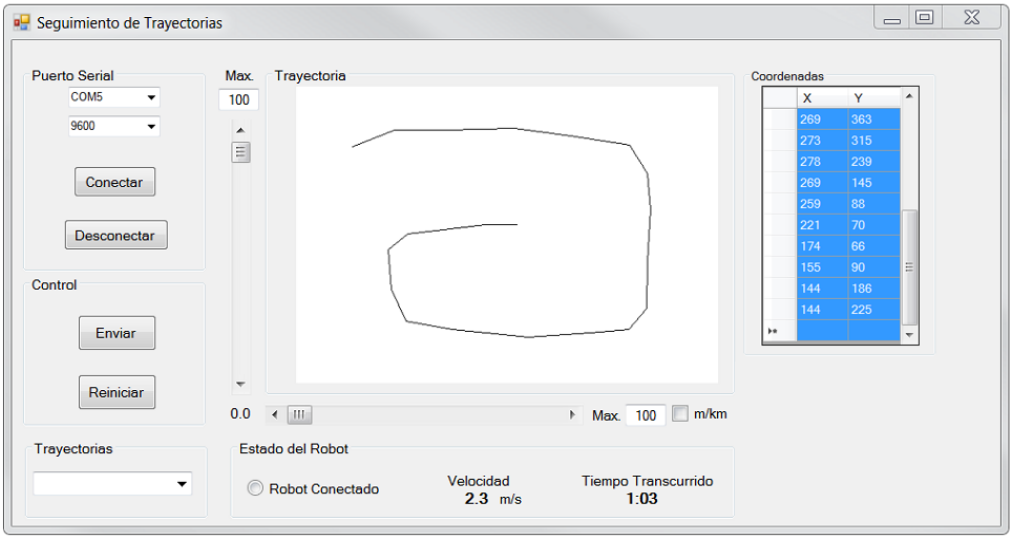


Figura 38 Generación de trayectoria en interfaz gráfica de usuario

Figura 39 Seguimiento de trayectoria visión artificial

CAPÍTULO 5. CONCLUSIONES

Se diseñó e implemento una interfaz gráfica de usuario capaz de generar trayectorias la cual se validó por un sistema robótico, en relación al control del sistema mecánico, la complejidad fue inherente debido a la dinámica propia del sistema, inclusive al tener un controlador robusto, estas no linealidades del mismo sistema producen variaciones, que aunque mínimas a través del tiempo generan un error que tiende a incrementar conforme el sistema pasa más tiempo en uso. El predecir estas variaciones a través de un modelo matemático, hace el sistema mucho más complejo de controlar.

El objetivo principal de este proyecto se obtuvo con éxito al realizar una interfaz gráfica para cualquier tipo de robot móvil, siendo de bajo costo, cuya característica principal es su portabilidad entre cualquier sistema robótico. En este trabajo se mostró el comportamiento de un sistema robótico de configuración diferencial, en el que inclusive, los sistemas mecánico y electrónico presentan deficiencias, el control trata de compensar estas variaciones para un seguimiento de la trayectoria que resulte en un error mínimo en estado estacionario. La siguiente etapa de este proyecto consistirá en utilizar un sistema de mejor resolución que arroje datos de mayor exactitud en la medición de las velocidades a las que el sistema se encuentra sometido, recordemos que el modelo matemático utilizado es cinemático, por lo que una caracterización/medición apropiada de estas variables resultara en un mejor comportamiento del sistema

REFERENCIAS

[1] M. Velasco-Villa P. Niño-Suarez y E. Aranda-Bricaire. Discretización exacta de un robot móvil con retardo de transporte. Ciencia e Ingeniería Neogranadina, 1 Bogotá, Colombia:45-53, 2006.

[2] Mario Alberto Gómez Rodríguez, Fernando Pech May.- Modelo odométrico diferencial de robots móviles. Cinvestav Tamaulipas.Enero 2011

[3] Rafael Vázquez Pérez. Conceptos Básicos de Locomoción. Instituto tecnológico de chihuaha.2005

[4] U. Cortés, A. Castañeda, A. Benítez, A. Díaz. Control de Movimiento de un Robot Móvil Tipo Diferencial. Artículo. Congreso Nacional de Control Automático, AMCA 2015.

[5] M. Granados-Contreras1 , J. G. Velásquez-Aguilar2 , A. Ramírez-Agundis1 , Outmane Oubram. Sistema de control de robot móvil con sensor de visión integrado, basado en FPGA, para fines educativos y de investigación. Artículo. Congreso Nacional de Control Automático, AMCA 2015.

[6] José Sánchez Bautista. Diseño y construcción de un robot móvil contra incendios. Benemérita Universidad Autónoma de Puebla. Junio 2013.

[7] R. Silva-Ortigoza, G. Silva-Ortigoza, V. M. Hernández-Guzmán, V. R. Barrientos-Sotelo, J. M. Albarrán-Jiménez y V. M. Silva-García .Trajectory Tracking in a

Mobile Robot without Using Velocity Measurements for Control of Wheels. IEEE Latinoamerica. Diciembre 2008.

[8] Evectronics system (2004 – 20019), Alicante-España, recuperado de: http://eventronic.es/13-robots-de-limpieza-de-conductos

[9] Marvelmind Robotics (2017-2019), El Camino Real #109-365 Sunnyvale CA EU, recuperado de: https://marvelmind.com/products/

11. APÉNDICE

Interfaz Gráfica Código Fuente

```
Imports System.IO
Imports System.IO.Ports
Imports System.Threading

Public Class Form1
    Dim myPort As Array
    Dim reset As Boolean = False

    Delegate Sub SetTextCallback(ByVal [text] As String) 'Added to prevent threading
        errors during receiveing of data
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
        MyBase.Load

        myPort = IO.Ports.SerialPort.GetPortNames()
        ComboBox1.Items.AddRange(myPort)

        GFX.FillRectangle(Brushes.White, 0, 0, PictureBox1.Width, PictureBox1.Height)
        'GFX.DrawLine(Pens.Red, 10, 10, 50, 50)
        ' GFX.DrawString("Robomatics", SystemFonts.DefaultFont, Brushes.Black, 60,
          60)
        PictureBox1.Image = BMP

        DataGridView1.ColumnCount = 2
        DataGridView1.Columns(0).Name = "X"
        DataGridView1.Columns(1).Name = "Y"

    End Sub

    Dim BMP As New Drawing.Bitmap(640, 480)
```

```
Dim GFX As Graphics = Graphics.FromImage(BMP)
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles Button1.Click
  GFX.FillRectangle(Brushes.White, 0, 0, PictureBox1.Width, PictureBox1.Height)
  'GFX.DrawLine(Pens.Red, 10, 10, 50, 50)
  ' GFX.DrawString("Robomatics", SystemFonts.DefaultFont, Brushes.Black, 60,
    60)
  PictureBox1.Image = BMP

  DataGridView1.Rows.Clear()
  RichTextBox1.Clear()
  RichTextBox1.Update()

  reset = True
End Sub
Private Sub PictureBox1_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles PictureBox1.Click
  Dim X_pos As Integer
  Dim Y_pos As Integer

  X_pos = MousePosition.X - Me.Location.X - PictureBox1.Location.X -
    GroupBox2.Location.X - 6

  Y_pos = MousePosition.Y - Me.Location.Y - PictureBox1.Location.Y -
    GroupBox2.Location.Y - 31

  Static previous_point As New Point(X_pos, Y_pos)

  BMP.SetPixel(X_pos, Y_pos, Color.DarkRed)

  If (reset = False) Then
```

```
            GFX.DrawLine(Pens.DarkRed, previous_point.X, previous_point.Y, X_pos,
        Y_pos)
    End If


    ' GFX.DrawString("X = " & X_pos & vbCrLf & "Y = " & Y_pos,
        SystemFonts.DefaultFont, Brushes.Black, X_pos, Y_pos)


    Dim row As String() = New String() {Y_pos, X_pos}
    DataGridView1.Rows.Add(row)
    DataGridView1.FirstDisplayedScrollingRowIndex = DataGridView1.RowCount -
        1

    previous_point = New Point(X_pos, Y_pos)
    PictureBox1.Image = BMP
    reset = False
End Sub

Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
        Button2.Click
    Dim output_buffer As String = " "
    Dim data_size As Integer = 0


    For Each row As DataGridViewRow In DataGridView1.Rows
        If Not row.IsNewRow Then
            '       MessageBox.Show(row.Cells(0).Value.ToString      &      ","      &
        row.Cells(1).Value.ToString)
            output_buffer += row.Cells(0).Value.ToString.PadLeft(4, "0"c) & "," &
        row.Cells(1).Value.ToString.PadLeft(4, "0"c) & ","
```

```
        End If
    Next
    data_size = DataGridView1.Rows.Count - 1
    TextBox1.Text = "0001," + data_size.ToString().PadLeft(4, "0"c) + "," +
      output_buffer



    ' SerialPort1.Write(TextBox1.Text & System.Environment.NewLine)



End Sub



Private Sub Button4_Click(sender As Object, e As EventArgs) Handles
      Button4.Click
    SerialPort1.PortName = ComboBox1.Text
    SerialPort1.BaudRate = ComboBox2.Text
    SerialPort1.Open()
End Sub



Private Sub Form1_MouseMove(ByVal sender As Object, ByVal e As
      System.Windows.Forms.MouseEventArgs) Handles MyBase.MouseMove
```

```
    Label1.Text = "X = " & e.X & vbCrLf & "Y = " & e.Y

    'Static last As New Point

    'If e.Button = System.Windows.Forms.MouseButtons.Left Then
    '    Me.CreateGraphics.DrawLine(Pens.Black, last.X, last.Y, e.X, e.Y)

    '    If e.Button = System.Windows.Forms.MouseButtons.Right Then
    '        Dim lbl As Label = New Label     'Create your Label
    '        lbl.Location = New Point(e.X, e.Y) 'Set Label Location

    '        lbl.Text = "X = " & e.X & vbCrLf & "Y = " & e.Y
    '        Me.Controls.Add(lbl)

    '    End If

    '  End If



    '   last = e.Location
End Sub

Private Sub SerialPort1_DataReceived(sender As System.Object, e As
      System.IO.Ports.SerialDataReceivedEventArgs) Handles
      SerialPort1.DataReceived
    ReceivedText(SerialPort1.ReadExisting())
End Sub

Private Sub ReceivedText(ByVal [text] As String) 'input from ReadExisting
```

```
    If Me.RichTextBox1.InvokeRequired Then
        Dim x As New SetTextCallback(AddressOf ReceivedText)
        Me.Invoke(x, New Object() {(text)})
    Else
        Me.RichTextBox1.Text &= [text] 'append text
    End If
End Sub

Private Sub Button3_Click(sender As Object, e As EventArgs)
    RichTextBox1.Clear()
    RichTextBox1.Update()

End Sub

Private Sub ComboBox3_SelectedIndexChanged(sender As Object, e As
      EventArgs) Handles ComboBox3.SelectedIndexChanged
    Dim VARX As New ArrayList
    Dim VARY As New ArrayList
    Dim COEFICIENTE As Single = 2
    Dim SENXCOEF As Single = 1
    Dim SENXEXP As Single = 1
    Dim SENTOEXP As Single = 1
    Dim PASO As Single = 0.1
    Dim ESCALA As Single = 30

    VARX.Clear()
    VARY.Clear()

    For I = -5 To 10 Step CSng(PASO)
        VARX.Add((I * ESCALA))
```

```
    Next

    For I = -5 To 10 Step PASO
        Try
            VARY.Add(-120 + COEFICIENTE * Math.Sin(Math.Pow(SENXCOEF *
      Math.Pow(I, SENXEXP), SENTOEXP)) * ESCALA)
        Catch ex As Exception
        End Try
    Next

    For I = 0 To VARX.Count - 2
        Try
            GFX.DrawLine(Pens.Blue, CSng(VARX(I)), CSng(VARY(I)) * -1,
      CSng(VARX(I + 1)), CSng(VARY(I + 1)) * -1)

        Catch ex As Exception

        End Try

    Next


End Sub

Private Sub CheckBox1_CheckedChanged(sender As Object, e As EventArgs)
      Handles CheckBox1.CheckedChanged

End Sub

Private Sub Label5_Click(sender As Object, e As EventArgs) Handles Label5.Click
```

```
    End Sub

    Private Sub Label9_Click(sender As Object, e As EventArgs) Handles Label9.Click

    End Sub
End Class
```