BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA
FACULTAD DE CIENCIAS DE LA ELECTRÓNICA
MAESTRÍA EN INGENIERÍA ELECTRÓNICA, OPCIÓN INSTRUMENTACIÓN ELECTRÓNICA

Tesis de Maestría:

"ALGORITMO DE VISIÓN BASADO EN APARIENCIA PARA NAVEGACIÓN AUTÓNOMA CON CAPACIDAD DE EVADIR OBSTÁCULOS"

Presenta:

PEDRO ALEJANDRO DE JESÚS GONZÁLEZ

Asesores:

Dr. Salvador Eugenio Ayala Raggi

M.C. Ricardo Álvarez González

El alumno | El asesor | El asesor

Puebla-México, Diciembre de 2014

A mi familia.

Resumen

En este trabajo se programaron y compararon dos enfoques distintos de navegación basada en apariencia, para que un robot móvil de dos ruedas fuera capaz de navegar autónomamente una ruta previamente entrenada. La ruta entrenada se construyo con imágenes del entorno tomadas cada treinta centímetros a lo largo de un ambiente estructurado en el interior de un edificio. En ambos casos se comparan las imágenes que el robot adquiere durante la navegación, con las imágenes guardadas durante el entrenamiento, para así determinar la ruta a seguir. El primer enfoque para lograr la navegación autónoma fue usando comparación de plantillas, haciendo la comparativa de similitud entre imágenes con el coeficiente de correlación. El segundo enfoque fue obteniendo los descriptores SURF de las imágenes, relacionando los descriptores correspondientes con la correlación cruzada normalizada y agrupando los puntos de interés con una variante del algoritmo RANSAC. El algoritmo que se programó también permite que el robot sea capaz de evadir obstáculos estáticos que no estaban durante la fase de entrenamiento y regresar a la ruta. Para esta etapa se uso la información obtenida de los sensores de rango con los que cuenta el robot y el procesamiento de imágenes para comenzar y terminar la evasión. Para el inicio de la evasión se usa un método de segmentación para determinar el espacio obstruido por el obstáculo y para el retorno a la ruta se usan de nuevo los descriptores SURF. Se hicieron pruebas en las que se cambiaron condiciones del ambiente entre el entrenamiento y la navegación autónoma, Pudiéndose observar que el uso de los descriptores SURF permite una mejor navegación a pesar de tener condiciones distintas entre el entrenamiento y la navegación autónoma.

Agradecimientos

En primer lugar, a mi asesor de tesis, Dr. Salvador Eugenio Ayala Raggi, al que tengo un gran respeto. Le agradezco por haberme brindado su apoyo en este proyecto, por su paciencia, por su valiosa dirección y amistad.

A mis compañeros y amigos de la MIE, sin un orden en particular, Lupita, Areli, José Luis, Pepe, Eliseo y Tomé, porque sin el equipo que formamos, no hubiera logrado esta meta.

A todos ustedes, mi mayor reconocimiento y gratitud!!

Para la realización de este trabajo se contó con con el apoyo de:

- Consejo Nacional de Ciencia y Tecnología, *CONACYT*, para realizar estudios de Maestría, con número de Becario: 274983.

Índice general

Índice de figuras **xi**

1 Introducción **1**
1.1 Objetivos 4
1.2 Organización de la Tesis 5

2 Generalidades y Estado del Arte **9**
2.1 Navegación Basada en Visión Artificial 9
2.1.1 Navegación en interiores 9
2.2 Enfoques Usados para Navegación por Visión 11
2.2.1 Visión Omnidireccional 11
2.2.2 Visión Estereoscópica 11
2.2.3 Visión Monocular 12
2.3 Evasión de Obstáculos 12
2.3.1 Algoritmo Bug 13
2.3.2 Vector Field histogram (VFH) 13
2.3.3 Otros algoritmos 14
2.4 Odometría 15
2.5 Dilatación 15
2.6 Erosión 16
2.7 Coeficiente de Correlación 17
2.7.1 Interpretación del coeficiente de correlación 18
2.7.2 Coeficiente de correlación entre dos imágenes 19
2.7.3 Algoritmo SURF 20

2.7.3.1 Detección 21
2.7.3.2 Imagen Integral 21
2.7.4 Algoritmo RANSAC 21

3 Diseño e Implementación del Algoritmo 23
3.1 Características Físicas del Sistema 23
3.2 Programación de la Interfaz 24
3.3 Diseño del Algoritmo 25
3.3.1 Comparación de Plantillas usando el Coeficiente de Correlación 27
3.3.2 Comparación de descriptores SURF usando RANSAC . . 28
3.4 Evasión de Obstáculos 28

4 Resultados 37
4.1 Pruebas de Navegación con Modificación de Parámetros 37

5 Conclusiones y trabajo futuro 41
5.1 Conclusiones de las pruebas realizadas 41
5.2 Publicaciones y ponencias resultado del trabajo de investigación . 41
5.3 Trabajo futuro 42

Bibliografía 43

Índice de figuras

1.1 Metodología del desarrollo del trabajo de tesis 6
1.2 Bloques del sistema implementado 6

2.1 Diagrama polar. 14
2.2 Ejemplos de vehículos que ocupan odometría a) Vehículo guiado por señal de radiofrecuencia. b)Vehículo con visión estéreo, IMU y Sensor de rango laser. c)Vehículo con IMU y GPS. 15
2.3 Proceso de dilatación en una imagen, a) imagen original, b) imagen binaria y c) imagen dilatada. 16
2.4 Efecto de la erosión a) Imagen con puntos blancos dispersos. b)Imagen erosionada sin puntos 17
2.5 Ejemplos de diagramas de dispersión con diferentes valores de coeficiente de correlación r. 18
2.6 Coeficiente de correlación entre imágenes idénticas 19
2.7 Coeficiente de correlación entre imágenes similares 20
2.8 Coeficiente de correlación entre imágenes distintas 20
2.9 Algoritmo RANSAC 22

3.1 Robot utilizado para la ejecución del algoritmo. 24
3.2 Interfaz usada para controlar el vehículo X80SV 25
3.3 Forma en la que se genera el entrenamiento de la ruta 26
3.4 Ilustración del modo para determinar la amplitud del giro. 27

3.5 Los puntos marcados con una cruz corresponden a la imagen actual y los puntos marcados con un circulo corresponden a la imagen de la memoria, La linea que los une es la distancia que indica los grados que se deben girar. 28
3.6 Rango de visión de los sensores integrados en la plataforma . . . 29
3.7 Imagen con rastros de entorno 30
3.8 Imagen de la que se obtiene el ancho 30
3.9 Imagen con rastros de entorno 31
3.10 Ilustración del modo en que se determina la trayectoria para regresar a la ruta. 33
3.11 Ilustración del modo en que se usan los sensores para definir el avance 34
3.12 Diagrama de flujo de pasos para evasión completa 36

4.1 Gráfica de errores de posición en cinco corridas 38
4.2 Gráfica de errores de posición en cinco corridas 38
4.3 Gráfica de errores de posición en cinco corridas 38
4.4 Gráfica de errores de posición durante una trayectoria que inicio con un error de posición de 20 cm para ambos métodos 38
4.5 Efecto en el uso de descriptores SURF de un objeto que modifica el entorno 39
4.6 Efecto en el uso de comparación de plantillas de un objeto que modifica el entorno 40

CAPÍTULO

1

Introducción

La idea de emplear un robot para realizar una tarea específica, en lugar de un ser humano ha sido fascinante y ha generado gran interés entre los investigadores en de distintos campos durante el pasado reciente.

La robótica abarca un amplio espectro de tecnologías en las que la inteligencia computacional es incorporada en las máquinas físicas, como la creación de sistemas con capacidades muy por encima de los componentes básicos. Tales sistemas robóticos pueden realizar tareas que son inalcanzables por las máquinas convencionales, o incluso por los seres humanos que usen herramientas convencionales.

Los sistemas robóticos se pueden emplear para una variedad de tareas que van desde la realización de la cirugía médica [1] a la tarea de montaje de un coche o a la tarea de atravesar en un entorno urbano [2] . Una de las principales habilidades a las que aspira este tipo de sistemas para llevar a cabo las tareas mencionadas es moverse por sí mismos, es decir, Áutonomía'. Los robots móviles autónomos son máquinas que se mueven por si solos, ya sea en el suelo, en el aire / espacio o bajo el agua. Estos vehículos son generalmente no tripulados, en el sentido de que los seres humanos no están a bordo. Las máquinas se mueven por sí mismas, con sensores y recursos computacionales a bordo para guiar su movimiento.

Ademas de lo anterior, estos vehículos también se emplean en tareas rutinarias que se producen en espacios y ambientes donde la movilidad de la máquina puede reemplazar con eficacia la presencia humana directa. Por ejemplo, en el cultivo a gran escala [3] y minería [4]. Finalmente las aplicaciones de vehículos robóticos también incluyen la asistencia personal, en las tareas domésticas y en el entretenimiento. Por ejemplo, una silla de ruedas para proporcionar movilidad a los discapacitados [5].

La Navegación Robótica Móvil se conoce como la capacidad de un robot para actuar sobre la base de su conocimiento y valores de sus sensores con el fin de alcanzar sus metas lo más eficiente y más fiable posible. En el primera instancia, puede parecer una tarea aparentemente trivial navegar un robot en comparación con la tarea de la cirugía o la fabricación de automóviles. Sin embargo las últimas tareas son cuidadosamente marcadas y formadas de tal manera que son en gran parte una aplicación de posicionamiento de alta precisión para una herramienta muy especializada. Mientras que en la navegación autónoma, el problema es que no hay alta precisión alrededor, no hay bases de datos disponibles acerca de lo que son los objetos en el mundo y el plano de planta. Además, el medio ambiente puede ser desconocido (con obstáculos), puede haber gente que se mueve alrededor, aparte de la presencia de objetos deformables, como las plantas, los juguetes, etc.

Tratar con un ambiente tan variables plantea una gran cantidad de desafíos para un sistema móvil.

Aunque el sensado por visión provee un flujo extremadamente rico de información que potencialmente sería excepcional para el control autónomo y el planeamiento de rutas. La tarea de interpretación es de gran complejidad. Además de que la velocidad de movimiento trae un reto en cuanto a la velocidad de procesamiento.

La navegación autónoma fusionando sensores y visión artificial es de las más efectivas, ya que tener más información del entorno, facilita las tareas y el procesamiento. Kim, Keller y Lattimer [6],Propusieron un algoritmo de navegación que usa fusión de sensores y visión artificial, para lograra que una plataforma robótica busque fuego en unas instalaciones cerradas. Wo (et al.)[7] desarrollaron un vehículo que transporta personas lesionadas de una lugar a otro fusionando sensores ultrasónicos, GPS, IMU, y una cámara estéreo. Trabajos como el de Jean y

Wang [8] también buscan que un robot patrulle una instalación, sin embargo a pesar de fusionar sensores y contar con una cámara, no aborda el tema de la evasión de obstáculos.

En este trabajo se programaron y compararon dos métodos para la navegación autónoma usando vision artificial basada en apariencia y se hicieron pruebas para la evasion de obstáculos con cada uno de los métodos.

Planteamiento del problema y Contribuciones

La navegación que no se basa en mapas geométricos tiene el principal inconveniente de tener una alta incertidumbre en cuanto a la posición y orientación del robot, del mismo modo, la navegación basada en apariencia no proporciona información para hacer ajustes precisos de posición, lo que provoca que durante el recorrido sea muy fácil acumular error en cada movimiento y a la larga provoca una estimación de localización errónea.

En la navegación basada en apariencia con vista monocular se tiene información reducida del entorno, porque además de no proveer información explícita de la posición precisa, se tiene una vista reducida del mundo circundante y no se posee información de objetos que pudieran aparecer en la ruta, lo que provoca tener un bajo rango de certidumbre para determinar la posición y la orientación correctas. Sin embargo esa información es suficiente para conocer muchos de los aspectos necesarios para evadir obstáculos como puede ser el tamaño, la forma, la textura, etc.

Para ayudar a tener una mayor certidumbre en cuanto a la posición y la ubicación de obstáculos, se usan sensores de rango infrarrojos y ultrasónicos.

Esta forma de abordar el problema se basa en la premisa de que una persona podría fácilmente guiar remotamente a un robot móvil dentro en un espacio que le es familiar, ya que como ser humano es capaz de discernir entre objetos móviles e inmóviles, inferir profundidad de acuerdo a los objetos presentes en la imagines, definir la mejor ruta para circular y saber cuando ya se logro alcanzar un objetivo. Todo eso a pesar de no contar con un mapa geométrico y de contar únicamente con una vista parcial del entorno.

Motivación

El campo de investigación relacionado con la Navegación Autónoma es actual y está abierto para ser abordado desde distintas y diversas perspectivas. Como pueden ser la fusión de sensores y el procesamiento de imágenes.

1.1 Objetivos

Para el trabajo de tesis se fijaron los siguientes objetivos:

Objetivo General

* Diseñar un algoritmo de navegación por visión, que permita a un robot móvil de dos ruedas navegar autónomamente y evadir obstáculos

Objetivos Específicos

1. Diseñar un algoritmo de visión que dote a una plataforma de dos ruedas con la capacidad de llevar a cavo un recorrido autónomo de un punto a otro siguiendo una ruta previamente entrenada.

2. Diseñar un algoritmo de navegación, con la fusión información de sensores ultrasónicos e infrarrojos para detectar y evadir obstáculos colocados en la ruta.

Solución Propuesta

El modo de navegación que se implemento consta de dos etapas. La de entrenamiento, en la que el vehículo almacena las imágenes del entorno por donde navegue y la segunda etapa que es la de navegación autónoma, en la que se comparan las imágenes almacenadas, con las imágenes que el vehículo toma durante su recorrido. En esta segunda etapa, se determina una nueva ruta y una nueva posición

cuando el vehículo esta en presencia de un obstáculo detectado con los sensores de la plataforma móvil.

Jong-Hann y Jheng-Liang en [8], proponen una forma de navegar en la que busca un rastro que indique la posición de la pared de forma visual y hace un seguimiento de este rastro para realizar tareas de patrullaje, el algoritmo propuesto puede ser implementado para patrullaje sin la necesidad de extraer características del entorno. Similarmente a lo planteado por JM. Maja et. al. en [9] se hace una segmentación de la imagen actual, pera determinar la posición y el tamaño de los obstáculos.

Diagrama a bloques de la metodología empleada

El diagrama de flujo que se sigue para alcanzar los objetivos del proyecto se muestra en la Figura 1.1.

Diagrama a bloques del sistema implementado

El diagrama de bloques que se implemento para probar el algoritmos se ve en la Figura 1.2.

1.2 Organización de la Tesis

Este documento de tesis se compone de cinco capítulos, donde abordan los fundamentos y el desarrollo del algoritmo de navegación y evasion de obstáculos, usando vision artificial basada en apariencia y sensores de rango.

El documento está organizado de la siguiente manera. El **Capítulo 2** comienza con la descripción de las características principales del sistema y del enfoque seleccionado para diseñar el algoritmo. En la segunda mitad de este capitulo se explica la teoría de los conceptos usados para el procesamiento de las imágenes en las distintas etapas del algoritmo

En el **Capítulo 3** se explican los pasos que se siguieron para el diseño e implementación del algoritmo final.

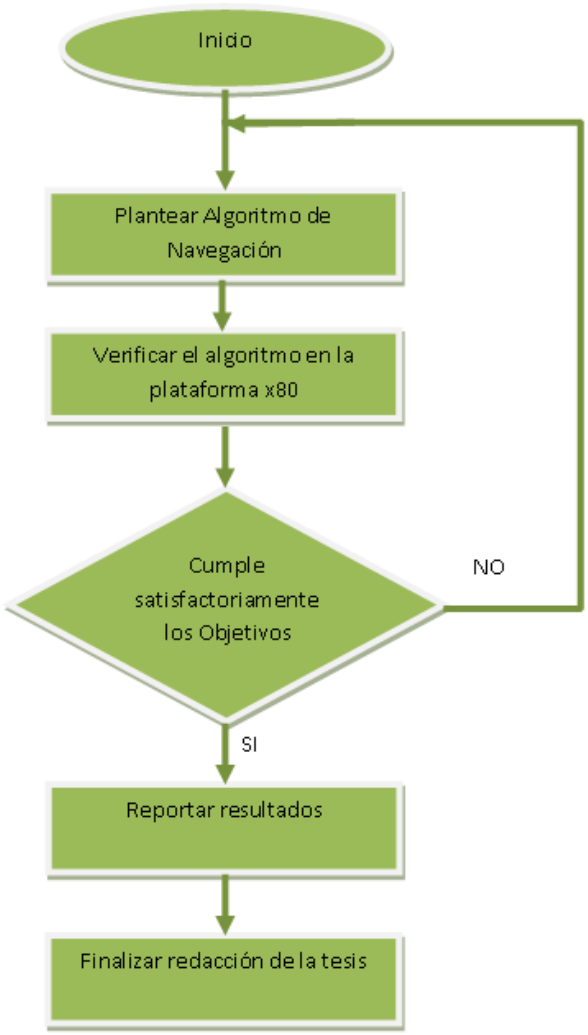

Figura 1.1: Metodología del desarrollo del trabajo de tesis

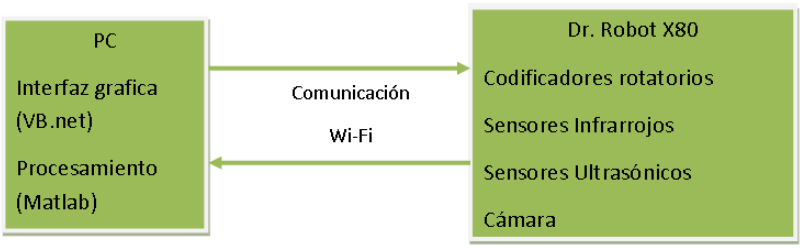

Figura 1.2: Bloques del sistema implementado

En el **Capítulo 4** se analizan y explican las pruebas realizadas tanto para la navegación como para la evasion y las condiciones finales del algoritmo diseñado.

Finalmente, en el **Capítulo 5** se presentan las conclusiones observadas y propuestas de posibles investigaciones que se podrían realizar para lograr una navegación basada únicamente en vision artificial.

CAPÍTULO

2

Generalidades y Estado del Arte

En este capítulo se da una introducción y un panorama general de los sistemas de antena inteligente. Se presenta el concepto, las formas de implementar, los componentes principales de este tipo de sistemas y se introduce la notación a emplear.

2.1 Navegación Basada en Visión Artificial

Durante más de treinta años esta área ha sido estudiada y desarrollada en dos frentes distintos: Navegación basada en visión para interiores y navegación basada en visión para exteriores.

2.1.1 Navegación en interiores

Desde los primeros trabajos fue evidente que era necesario indicarle a la computadora lo que se supone vería. Algunos de los primeros trabajos se basaron principalmente en la geometría del espacio, en algunos otros se usaron modelos generados por computadora, en otros modelos topológicos del ambiente e incluso secuencias de imágenes. Cuando las secuencias de imágenes fueron usadas para

la representación del espacio, las imágenes tomadas durante la navegación fueron usadas para compararlas con lo que el robot se supone que tendría que ver [10].

Los siguientes esfuerzos fueron encaminados en tres grupos principales:

- Navegación basada en mapas: Son sistemas que dependen de mapas geométricos o topológicos del entorno creados por el usuario.

- Navegación basada en construcción de mapas: Son sistemas que usan sensores para construir su propio modelo del entorno y usan este modelo para la navegación.

- Navegación sin mapas: Estos son sistemas que no usan una representación explicita del entorno, si no que reconoce los objetos en el ambiente y en base a eso genera su movimiento.

Dentro de la navegación basada en mapas, hay procesos computacionales que se dividen en cuatro [11]

- Adquirir información de sensores: Cuando es únicamente navegación por visión esto se refiere tomar imágenes y digitalizarlas.

- Detectar marcas: Esto se refiere a extraer bodes, suavizar, filtrar y segmentar regiones basado en las diferencias de color, de profundidad o de movimiento.

- Establecer coincidencias entre lo observado y lo esperado: En esta etapa el sistema trata de identificar las marcas observadas comparándolas con las que tiene en la base de datos.

- Calcular posición una vez que se detectaron las coincidencias: El sistema necesita calcular la posición en función de las marcas observadas y la posición de estas en la base de datos.

2.2 Enfoques Usados para Navegación por Visión

2.2.1 Visión Omnidireccional

La visión omnidireccional aplicada a robots móviles consta de una cámara que apunta hacia una lente en forma de parábola y con esto logra obtener una imagen del medio circundante al punto donde fue tomada la fotografía.

Este tipo de visión es usada por Taha y Arif Mat-Jizat para detectar marcas artificiales en ambientes controlados con poca iluminación [12], En trabajos como el de Zhang, Cao y Lou usan una vista omnidireccional y un enfoque basado en apariencia para navegar [13]. En el caso de Wang, Zhao y Li usan análisis de componentes principales y estas imágenes para crear mapas de navegación [14].

Las ventajas de usar un sistema así es que se tiene una visión más amplia del entorno, lo que implica mayor información para navegar. En principio se pudiera pensar que si se toman imágenes con una cámara monocular y se van añadiendo conforme se gira la cámara, se lograría el mismo efecto, pero el hecho es que cuando se toman imágenes separadas, la iluminación de la imagen va cambiando entre cada foto y esto provoca que al unirse las fotografías, la imagen resultante sea drásticamente distinta a una omnidireccional. Otro aspecto importante es que tomando una sola fotografía omnidireccional, el entrenamiento de la ruta seria más rápida en comparación con tomar fotografías, girar y unir.

Las desventajas radican en el tiempo de procesamiento, al tener que hacer una transformación de la imagen para verla como un plano y el costo del sensor omnidireccional.

2.2.2 Visión Estereoscópica

De manera natural, al igual que nuestro sistema auditivo, nuestro mecanismo de visión es estéreo, es decir, somos capaces de apreciar, a través de la visión binocular, las diferentes distancias y volúmenes en el entorno que nos rodea. Nuestros ojos, debido a su separación, obtienen dos imágenes con pequeñas diferencias entre ellas, a lo que denominamos disparidad.

En esta aproximación de visión artificial, se busca emular lo que nuestro cerebro hace al procesar las diferencias entre ambas imágenes e interpretarlas de forma que

percibimos la sensación de profundidad, lejanía o cercanía de los objetos que nos rodean.

Chitra (et al.) usan dos cámaras para obtener la información tridimensional de las imágenes que analizan [15]. Schauwecker y Zell usan dos pares de cámaras como sensores para dotar a un cuadrirotor con la capacidad de navegar [16]. Chiang (et al.) usa un sistema estereoscópico empotrado en un robot humanoide que juega futbol, el sistema se usa para medir la distancia del robot al balón y del robot a las marcas de la cancha [17].

2.2.3 Visión Monocular

Esta aproximación de visión artificial es la más barata de implementar debido a que únicamente se usa una sola cámara para obtener las fotografías, esta cámara puede ser de baja resolución y costo, lo que la hace fácil de obtener. Sin embargo, es la que presenta mayores retos en cuanto a la extracción de la información. Debido a que únicamente se tiene una vista parcial del espacio circundante, creando una limitante en cuanto a la creación del mapa del recorrido a seguir. Otra desventaja es que para obtener información sobre la profundidad de los objetos de las imágenes, se tiene que agregar más fotografías y mas procesamiento de las imágenes.

En el trabajo de Tsai (et al.) se uso una cámara y dos haz laser verdes para calcular las distancias de distintos objetos, dependiendo de la separación de los haces de luz [18]. Yamada, Lin y Cheng desarrollaron un algoritmo en el que con una sola cámara se logra detectar obstáculos y navegar [19].

De los métodos anteriores se selecciono el último, principalmente porque la plataforma con la que se cuenta tiene integrados seis sensores ultrasónicos y siete sensores infrarrojos de rango para obtener información de la distancia de los objetos en las imágenes. Así como una cámara que permite tomar imagines que cubren aproximadamente 150° del perímetro circundante.

2.3 Evasión de Obstáculos

La evasión de obstáculos se enfoca en cambiar la trayectoria del robot de acuerdo a la información de sus sensores durante su movimiento. El movimiento resultante

del robot, es una función de las lecturas actuales de sus sensores y la posición deseada. Los siguientes algoritmos dependen en mayor o menor grado de la existencia de un mapa global y en el conocimiento preciso del robot de su posición respecto al mapa. A pesar de sus diferencias estos algoritmos pueden considerarse de evasión de obstáculos, porque las lecturas de los sensores juegan un rol importante en la futura trayectoria del robot.

2.3.1 Algoritmo Bug

Quizá es el más simple de los algoritmos [20, 21] , El nombre da una idea de cómo funciona, básicamente consiste en seguir el contorno de cada obstáculo y así atravesarlo.

Algoritmo Bug 1, primero rodea completamente el primer obstáculo y se posiciona en el punto con la distancia más corta hacia la meta. Esta aproximación es muy ineficiente, pero garantiza que el robot alcance la meta.

Algoritmo Bug 2, el robot comienza siguiendo el contorno del obstáculo, y comienza a avanzar cuando es capaz de moverse directo a la meta, en general este algoritmo tiene menor recorrido que el primero, sin embargo pueden haber situaciones en las que este algoritmo no alcance la meta

Algoritmo Bug Tangente [22], Este algoritmo agrega sensores de rango y una representación del entorno denominada gráfica tangente local (LTG), no solo se mueve más eficientemente el robot, si no también puede tomar atajos cuando rodea los obstáculos.

2.3.2 Vector Field histogram (VFH)

Borenstein y Koren crearon el algoritmo VFH y después se extendió a VFH+ [23], y VFH* [24],. Esta técnica consiste en que a diferencia del algoritmo Bug, esta no depende únicamente de la lectura actual de los sensores. Creando un mapa local del entorno alrededor del robot que tiene lecturas recientes. Para la evasión de obstáculos VFH genera un histograma polar donde en eje x representa el ángulo en el cual fue encontrado el obstáculo y el eje y, representa la probabilidad P de que realmente haya un obstáculo. De este histograma, se calcula la dirección de desplazamiento. Primero se identifican las aberturas por las que puede pasar el vehículo

como se indica en la Figura2.1, después se analizan las aberturas candidatas y se elije la que tenga una menor función de costo G. La función G tiene 3 términos.

G= a* Dirección del objetivo+ b* Orientación de las llantas + c* Dirección previa.

Dirección del objetivo = alineación del robot con la meta. Orientación de las llantas = diferencia entre la nueva dirección y la orientación actual de las llantas. Dirección previa = diferencia entre la dirección seleccionad previamente y la nueva dirección. Los parámetros a, b y c son seleccionados para ajustar el comportamiento del robot.

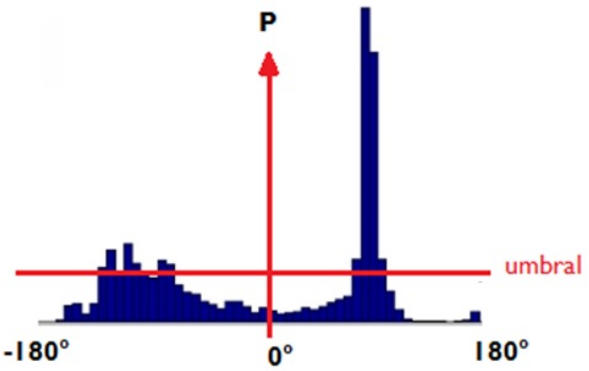

Figura 2.1: Diagrama polar.

2.3.3 Otros algoritmos

Existen muchos mas algoritmos de evasión de obstáculos que implican modelos dinámicos de las plataformas móviles como son el método de cercanía [25], El Autonomous Systems Lab (ASL)[26], la aproximación Schlegel [27], la aproximación de ventana dinámica [28], el método de curvatura de velocidad [29], la técnica de la banda elástica [30] y muchas variantes de estas. También hay propuestas con lógica difusa [31].

2.4 Odometría

En cuanto a la odometría, es posible observar en trabajos recientes, que es una herramienta muy usada en navegación, ubicación y mapeo en robots móviles. Sin embargo tiene limitantes fuertes que están relacionados con el terreno sobre el que se circula y relacionados con la arquitectura del sistema. Se usa como apoyo de muchos más sistemas para lograr una navegación efectiva en los distintos ambientes para los que fueron construidas las plataformas robóticas.

En la Figura2.2 se muestran imágenes de trabajos publicados en la IEEE, en el inciso a se usa como apoyo a un sistema de detección de posición con respecto a transmisores de radiofrecuencia [32], en el b se usa como apoyo en sistemas que cuentan con cámaras 3D [33] y en el c en sistemas con unidades de medición inerciales y de posicionamiento global GPS [34]. Con estas observaciones es evidente que aunque presenta muchos errores, es básico para una navegación funcional.

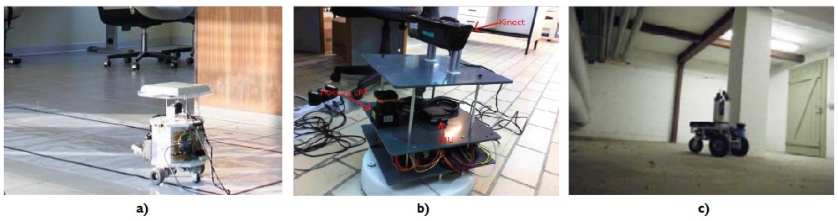

Figura 2.2: Ejemplos de vehículos que ocupan odometría a) Vehículo guiado por señal de radiofrecuencia. b)Vehículo con visión estéreo, IMU y Sensor de rango laser. c)Vehículo con IMU y GPS.

2.5 Dilatación

La dilatación es una operación es una operación morfológica correspondiente a la idea intuitiva del crecimiento, o de añadir una capa de píxeles a la estructura de la imagen. La manera específica en la que es controlado el crecimiento, es por medio de una figura referida como estructura de referencia. Esta operación es definida en la notación de conjuntos de puntos como:

$$I \oplus H = \{(x', y') = (x + i, y + j) | (x', y') \in P_I(i, j) \in P_n\} \tag{2.1}$$

Como indica la expresión 2.1, el conjunto de puntos que constituyen a la dilatación de una imagen I y la estructura de referencia H, queda definido por todas las posibles combinaciones de los pares de coordenadas de los conjuntos de puntos P_I y P_H. Se podría también interpretar a la operación de dilatación, como el resultado de añadir a los pixeles 1 de la imagen (P_I) la forma correspondiente a la estructura de referencia. Un ejemplo del proceso de dilatación en una imagen se ilustra en la Figura 2.3.

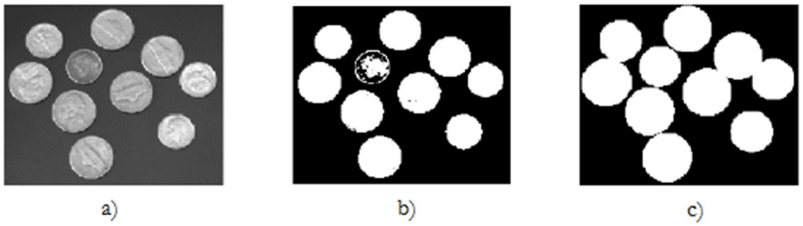

Figura 2.3: Proceso de dilatación en una imagen, a) imagen original, b) imagen binaria y c) imagen dilatada.

En la Figura 2.3, en la imagen binaria b) se observa que quedan pixeles o pequeños grupos de ellos que sabemos que deberían pertenecer al objeto pero que por diferentes motivos no quedaron identificados. Adicionalmente aparecen pixeles que sabemos que no deberían haber aparecido. Para incorporar estos pixeles al objeto conviene hacer a dicha imagen el proceso de dilatación, tal y como se observa en c).

2.6 Erosión

La operación cuasi-inversa de la operación de dilatación es la erosión, la cual es expresada en la notación de conjunto de puntos es definida como:

$$I \ominus H = \{(x', y') = (x' + i, y' + j) | (x', y') \in P_I(i, j), \forall (i, j) \in P_H\} \qquad (2.2)$$

La expresión 2.2, dice que para cada punto (x', y') de la imagen, el resultado se compone por los puntos (x', y') para los cuales todos los posibles valores $(x' + i, y' + j)$ se encuentran en I. La figura 2.4 muestra un ejemplo de la erosión aplicada a una imagen con ruido sal y pimienta.

Figura 2.4: Efecto de la erosión a) Imagen con puntos blancos dispersos. b)Imagen erosionada sin puntos

2.7 Coeficiente de Correlación

El coeficiente de correlación entre dos variables es una cantidad que expresa la calidad de un ajuste por mínimos cuadrados de las variables. Para una ajuste de mínimos cuadrados de n puntos (x_i, y_i), el coeficiente b en.

$$y = ax + bx$$

está dado por

$$b = \frac{n\sum xy - \sum x \sum y}{n\sum x^2 - (\sum x)^2} \tag{2.3}$$

y el coeficiente b' en

$$x = a' + b'y$$

$$b = \frac{n\sum xy - \sum x \sum y}{n\sum y^2 - (\sum y)^2} \tag{2.4}$$

El cuadrado del coeficiente de correlación (r^2) esta definido como

$$r^2 = bb' \tag{2.5}$$

$$= \frac{[n\sum xy - \sum x \sum y]^2}{[n\sum x^2 - (\sum x)^2]\,[n\sum y^2 - (\sum y)^2]} \tag{2.6}$$

Por otra parte, la media de $\overline{x}$ e $\overline{y}$ de x e y respectivamente como

$$\overline{x} = \frac{1}{n}\sum x_i \tag{2.7}$$

$$\overline{y} = \frac{1}{n}\sum y_i \tag{2.8}$$

además, b y b' se pueden escribir:

$$b = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sum(x_i - \overline{x})^2} \tag{2.9}$$

$$b' = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sum(y_i - \overline{y})^2} \tag{2.10}$$

entonces, el coeficiente de correlación se define por

$$r = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum(x_i - \overline{x})^2}\sqrt{\sum(y_i - \overline{y})^2}} \tag{2.11}$$

2.7.1 Interpretación del coeficiente de correlación

El coeficiente de correlación tiene como resultado un valor en el intervalo [-1,1], donde si $r = 1$ entonces las variables están perfectamente correlacionadas, es decir, todos los datos se encuentran en una linea en la cual x incrementa cuando y incrementa. Si $r = 0$ no existe correlación y si $r = -1$ las variables se encuentran perfectamente inversamente correlacionadas, en otras palabras, si x incrementa y disminuye. La Figura 2.5 muestra ejemplos de variables con diferentes valores de r

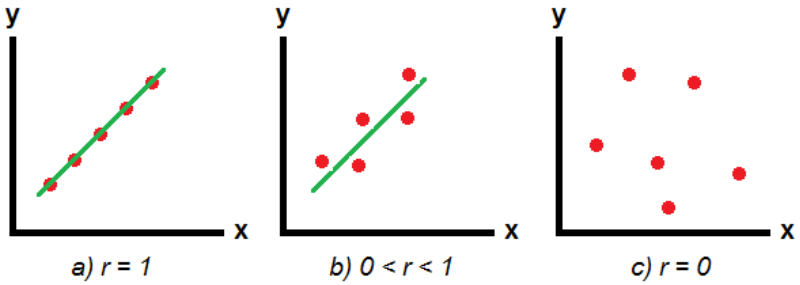

Figura 2.5: Ejemplos de diagramas de dispersión con diferentes valores de coeficiente de correlación r.

Puede observarse que cuando existe una relación lineal entre los datos, el valor de r sera cercano a 1, mientras que cuando los datos están dispersos, r será 0.

2.7.2 Coeficiente de correlación entre dos imágenes

Dadas dos imágenes $A(x, y)$ y $B(x, y)$ de tamaño $m \times n$, cuyos valores medidos de niveles de gris son dados por $\overline{A}$ y $\overline{B}$ respectivamente, el coeficiente de correlación r entre A y B está dado por:

$$r = \frac{\sum_{m,n}(A_{mn} - \overline{A})(B_{mn} - \overline{B})}{\sqrt{\left[\sum_{m,n}(A_{mn} - \overline{A})^2\right]\left[\sum_{m,n}(B_{mn} - \overline{B})^2\right]}} \tag{2.12}$$

A continuación se presentan ejemplos de comparación entre dos imágenes con diferentes grados de similitud, así cono las gráficas de los valores de escala de grises de las imágenes.

La Figura 2.6 presenta la comparación entre dos imágenes idénticas. En la gráfica de la figura se puede apreciar claramente la relación lineal que existe. Para esta comparación se obtuvo un valor de $r = 1$, esperado para dos imágenes iguales.

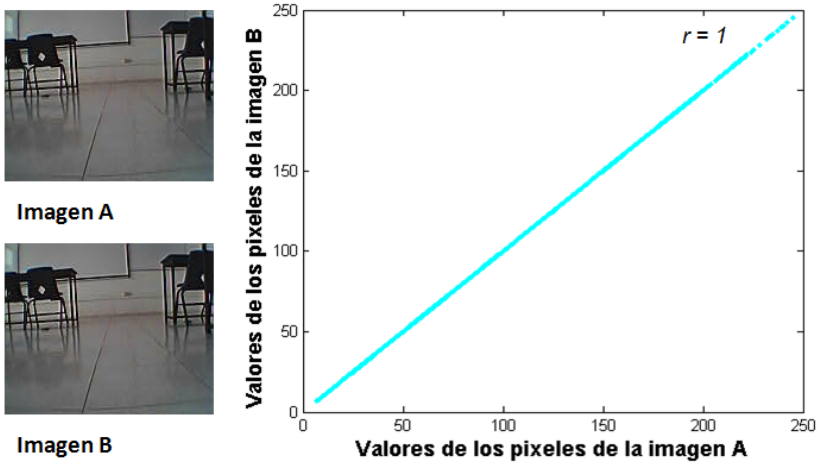

Figura 2.6: Coeficiente de correlación entre imágenes idénticas

La Figura 2.7 muestra el valor de r cuando las imágenes son muy parecidas,también se puede observar que las imágenes tienen una ligera variación en la iluminación, si embargo, conservan una relación lineal como lo india el valor de $r = 0.9204$. La Figura 2.8 presenta el resultado de la comparación entre dos imáge-

nes distintas, la gráfica de los valores de escala de grises muestra valores dispersos y no existe una relación lineal, comprobado por le resultado de $r = 1891$

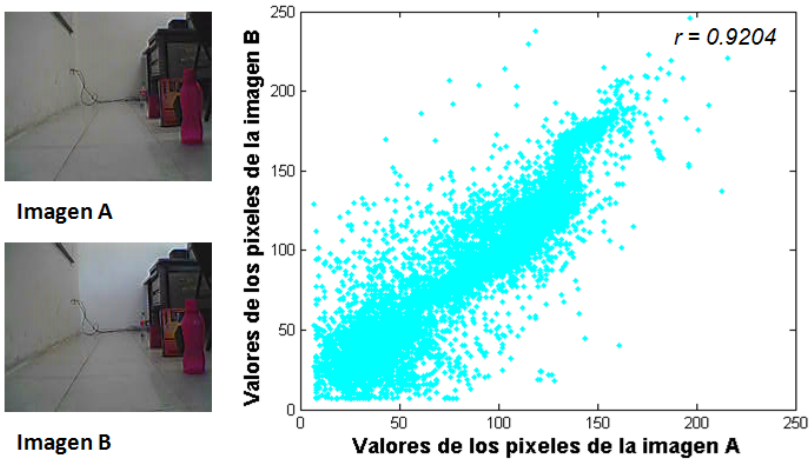

Figura 2.7: Coeficiente de correlación entre imágenes similares

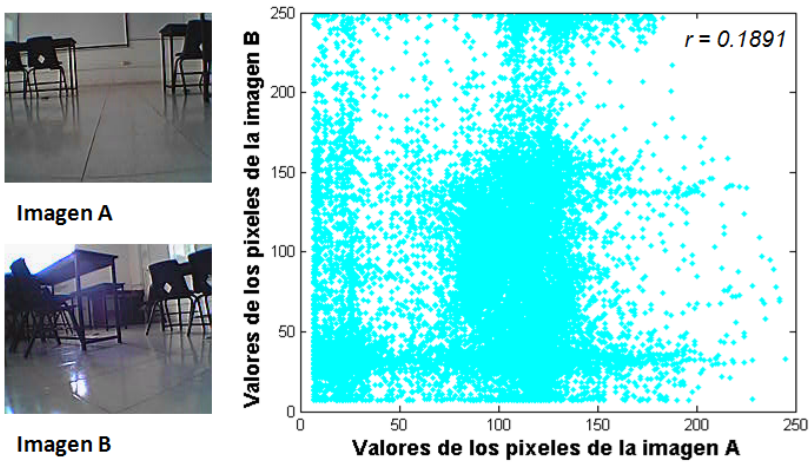

Figura 2.8: Coeficiente de correlación entre imágenes distintas

2.7.3 Algoritmo SURF

SURF es un detector y un descriptor de alto rendimiento de los puntos de interés de una imagen, donde se transforma la imagen en coordenadas, utilizando una técnica

llamada multi-resolución. Consiste en hacer una réplica de la imagen original de forma Piramidal Gaussiana o Piramidal Laplaciano, y obtener imágenes del mismo tamaño pero con el ancho de banda reducido. De esta manera se consigue un efecto de borrosidad sobre la imagen original, llamado Scale-Space [35]. Esta técnica asegura que los puntos de interés son invariantes en el escalado. El algoritmo SURF esta basado en el predecesor SIFT.

2.7.3.1 **Detección**

El algoritmo de SURF está basado en los mismos principios y pasos que el SIFT, pero utiliza un esquema diferente y esto debería proveer mejores resultados: más rapidez. Con el fin de detectar puntos característicos en una escala de manera invariable SIFT utiliza filtros de aproximación en cascada. Donde la Diferencia de Gaussianos, DOG, se calcula sobre imágenes re-escaladas progresivamente .

2.7.3.2 **Imagen Integral**

Semejante al SDoG. En vez de utilizar gaussianas para promediar la imagen, se utilizan cuadrados (aproximaciones). Hacer la convolución de la imagen con un cuadrado es mucho más rápido si se utiliza la imagen integral.

2.7.4 **Algoritmo RANSAC**

Descripción del algoritmo RANSAC [36].

```
Dados:
   datos - conjunto de puntos observados
   modelo - modelo que puede ser ajustado al conjunto de datos
   n - numero mínimo de valores requeridos para ser ajustados al modelo
   k - numero máximo de iteraciones permitidas del algoritmo
   t - valor de umbral para determinar que un conjunto de datos se ajusta a un modelo
   d - numero de datos cercanos requeridos para aceptar el ajuste de un modelo
Regresa:
   mejor_ajuste - parámetros del modelo que mejor se ajusta a los datos (es nulo si ningún modelo se
ajusta)
Iteraciones = 0
Mejor ajuste = nulo
Mejor error = algo muy grande
Mientras iteraciones < k {
   Posibles candidatos = n aleatoriamente seleccionados de los datos
   Posible modelo = parámetros de modelo ajustados a Posibles candidatos
   También candidatos = conjunto vacío
   Para cada punto en el conjunto de datos que no están en Posibles candidatos {
      Si los puntos de ajustan a Posible modelo con un error menor a t
         agrega un punto a Posibles candidatos
   }
   Si el número de elementos en También candidatos es > d {
         mejor_modelo = parámetros del modelo ajustados a todos los puntos en Posibles candidatos y a
         También candidatos
         Este error = una medida de que tan bien ajusta el modelo a los puntos
      Si Este error < Mejor error {
         Mejor ajuste = Mejor modelo
         Mejor error = Este error
      }
   }
   incrementa iteraciones
}
regresa Mejor ajuste
```

Figura 2.9: Algoritmo RANSAC

CAPÍTULO

3

Diseño e Implementación del Algoritmo

3.1 Características Físicas del Sistema

El sistema consta de un robot móvil de 2 ruedas que transmite la información de los sensores hacia una laptop, esta ejecuta el algoritmo y envía de regreso instrucciones de movimientos al robot. El robot que se usó fue el "Dr. Robot X80SV WiFi Mobile Development Platform"(Figura 3.1), el cual es una plataforma robótica de uso especial para prueba de algoritmos.

El robot tiene las siguientes características principales:

* Cámara con pan-tilt-zoom a color con resolución de 640x480 pixeles.
* Dos motores de 12 Volts con mas de 40 kg.cm de torque cada uno.
* Completa comunicación inalámbrica.
* Sensores de rango ultrasónicos e infrarrojos
* Dimensiones 38cm(largo) x 35cm. (ancho)x 28cm. (alto).

Figura 3.1: Robot utilizado para la ejecución del algoritmo.

A la computadora se le instalo Matlab R2014a para usar la paquetería de procesamiento de imágenes mas actual y también se le instalo SharpDevelop 4.3 para programar la interfaz que controla al robot.

3.2 Programación de la Interfaz

El programa que ejecuta el algoritmo de navegación consta de una capa base programada en VB.Net cuya tarea es interpretar comandos de movimiento generados por Matlab y trasmitir la información sensada por el robot hacia las funciones de Matlab.

Para lograr el comportamiento deseado se programó una interfaz con algunas de las funciones que la plataforma trae en su manual de referencia de la API (Application Programming Interface)[37, 38, 39]. Para hacer la vinculación con matlab fue necesario profundizar en la programación de objetos y hacer la evaluación de una función en tiempo de ejecución dentro de VB.Net. [40, 41].

La decision de limitar las funciones realizadas por VB.Net fue para optimizar el tiempo de desarrollo, ya que de lo contrario en cada nuevo comportamiento del

robot se hubieran tenido que hacer modificaciones tanto en Matlab como en VB.net lo que podría acarrear muchos problemas por el hecho de ser dos lenguajes de programación distintos.

En los dos lenguajes, la programación fue modular y con el menor numero de variables globales, para así lograr un mejor control de cada una de las etapas del procesamiento.

La interfaz esta hecha para usarla en modo entrenamiento o en modo navegación autónoma.Para el modo entrenamiento se habilita el botón “Mover” y las cajas de texto para indicar el giro y la distancia que se desee. Para el modo navegación autónoma se habilita el botón “Seguir Ruta” como se ve en la Figura 3.2.

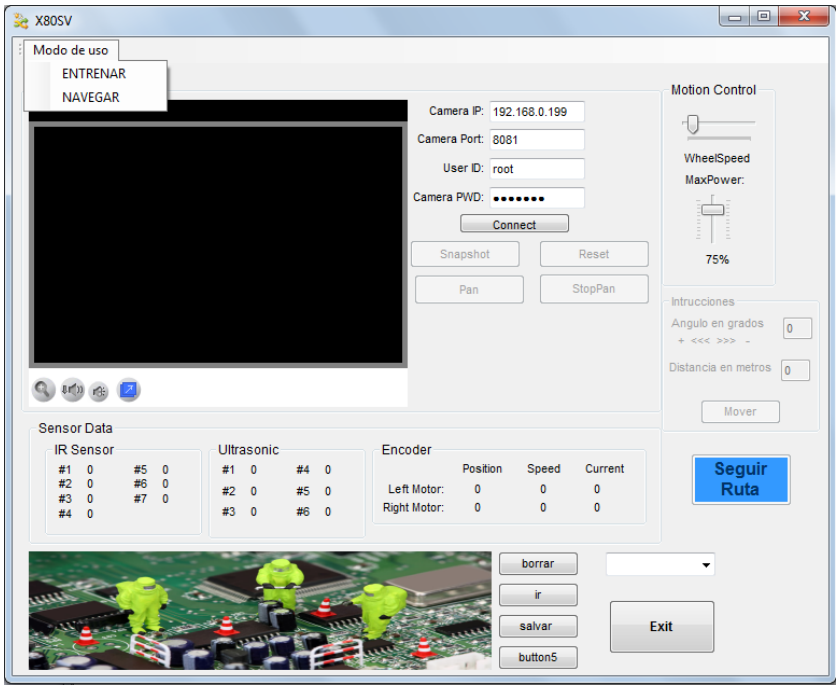

Figura 3.2: Interfaz usada para controlar el vehículo X80SV

3.3 Diseño del Algoritmo

El algoritmo final es resultado de la optimización de parámetros y numerosa pruebas comparativas entre dos enfoques de navegación usando vision artificial basada en apariencia.

El algoritmo de navegación que se programo para ambos casos consto de dos etapas, una de entrenamiento y otra de navegación autónoma.

En el entrenamiento, se crea una base de datos de la ruta en donde se almacenan tres imágenes de la vista frontal de cada una de las posiciones como se ve en la Figura 3.3.

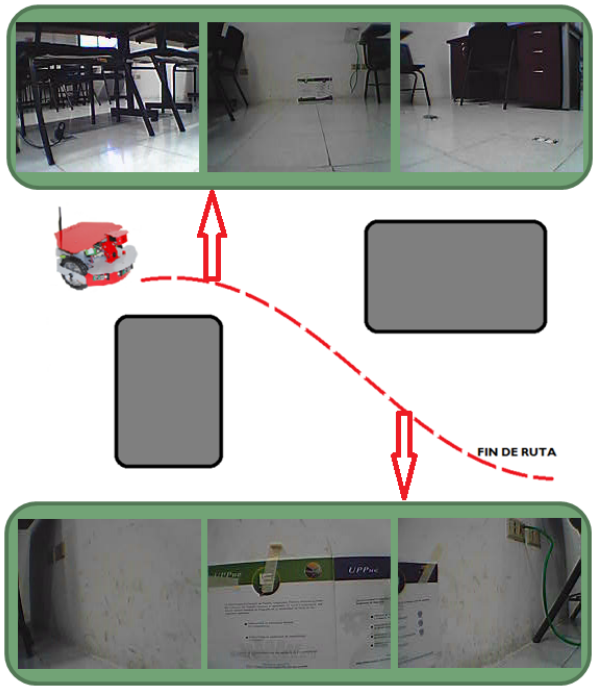

Figura 3.3: Forma en la que se genera el entrenamiento de la ruta

La imagen almacenada esta compuesta por una fotografía central y dos laterales, logrando así un ángulo de visión aproximado de 228 grados. Una vez entrenada la ruta, se coloca al robot en el punto de inicio para realizar el recorrido autónomo, este comienza haciendo uso únicamente de la información visual que entrega la cámara. Para comenzar a recorrer la ruta, lo primero que se necesita es conocer la posición y la orientación del robot. En las condiciones actuales, la posición se conoce dado que es una condición de inicio, no así la orientación precisa.

3.3.1 Comparación de Plantillas usando el Coeficiente de Correlación

En este caso para conocer dicha orientación, se cuenta con la imagen de la posición actual almacenada en memoria, sobre la que se comparará la vista que el robot acaba de capturar. Esto se hace comparando la imagen de la vista actual del robot y segmentos del mismo tamaño de la imagen almacenada en memoria. La comparación se hace calculando el coeficiente de correlación con cada uno de los segmentos.

Dependiendo del punto en el que la correlación entre la imagen actual y la imagen del banco sea mayor, se genera una posible respuesta de movimiento que puede ser giro a la izquierda, giro a la derecha ó avance hacia el frente; la amplitud en los giros depende de la distancia en pixeles donde se encontró la mayor correlación respecto del centro de la imagen como se ejemplifica en la Figura3.4.

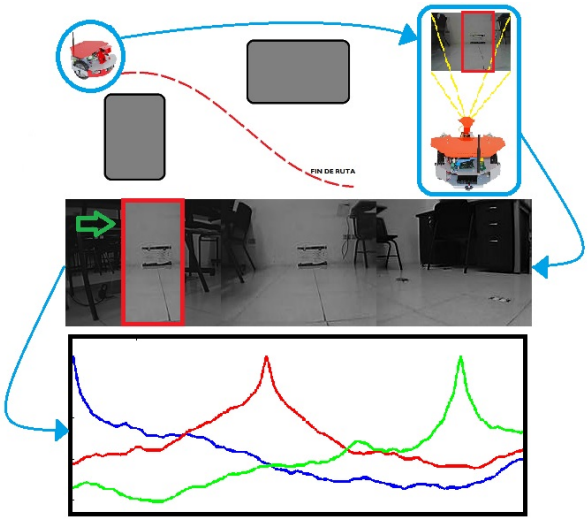

Figura 3.4: Ilustración del modo para determinar la amplitud del giro.

En el caso particular del ensamble del robot, cada grado de la cámara, corresponde al desplazamiento de 4.2 pixeles, por lo tanto como la imagen almacenada mide 960 pixeles de largo, el máximo ángulo de corrección de orientación aproximado es de 76 grados. Aunque en algunas circunstancias del entorno, el ángulo de corrección seria un poco mayor, provocado por la similitud de los segmentos laterales de la imagen de la base de datos y de la imagen actual.

3.3.2 Comparación de descriptores SURF usando RANSAC

De un modo similar, sa calcula el ángulo que el robot tiene que girar para quedar orientado correctamente en la ruta, pero en este caso esa distancia en pixeles es definida por la separación entre puntos SURF.

Esos puntos son asignados después de haber comparado sus descriptores y después de haber aplicado el algoritmo RANSAC a todos los puntos encontrados, lo que asegura los puntos en las dos imágenes corresponden con la misma posición de la ruta como se ejemplifica en la Figura3.5.

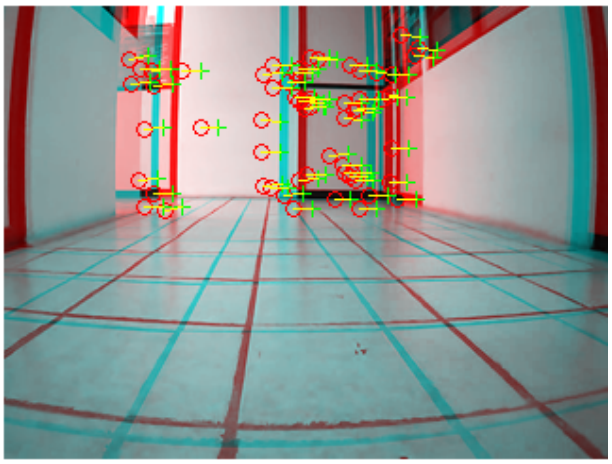

Figura 3.5: Los puntos marcados con una cruz corresponden a la imagen actual y los puntos marcados con un circulo corresponden a la imagen de la memoria, La linea que los une es la distancia que indica los grados que se deben girar.

3.4 Evasión de Obstáculos

El algoritmo de evasión de obstáculos comienza durante la etapa de navegación autónoma. La detección de un obstáculo por los sensores frontales del vehículo es lo que provoca que inicie la evasión. La Figura3.6 Muestra el rango de visión que tiene la plataforma.

Una vez detectado el obstáculo por los sensores de rango frontales, se adquiere una imagen de la vista frontal para compárala con la imagen de la base de datos en esa posición. Estas dos imágenes son restadas para detectar el obstáculo. Si las dos

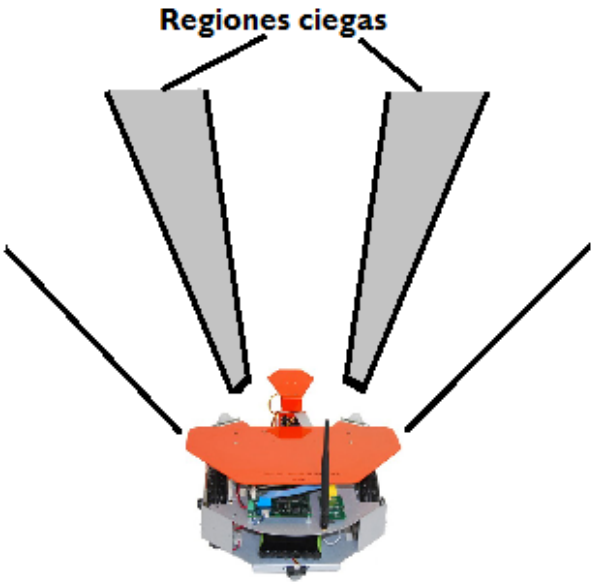

Figura 3.6: Rango de visión de los sensores integrados en la plataforma

imágenes fueran idénticas a excepción del obstáculo, el resultado de esta operación sería una imagen del obstáculo con fondo negro. Sin embargo en la práctica, en la imagen resultante aparecen otras marcas debido a la desviación existente entre la posición del robot durante el entrenamiento y la posición durante el recorrido autónomo.

Para poder determinar de un modo más aproximado el ancho del obstáculo y así calcular el mejor ángulo de giro y distancia de avance para el primer paso de la evasión, se tienen que eliminar los rastros de imágenes adicionales como las mostradas en la Figura 3.7. Para este propósito se “erosiona” la imagen, obteniendo así una reducción de otros restos. Sin embargo, este procedimiento reduce también el ancho del objeto, por lo que es necesario recuperar el ancho mostrado de la imagen original y para esa tarea se “dilata” quedando una imagen final como en la Figura3.8.

Una vez detectado y aislado el obstáculo, se cuentan los pixeles que abarca y se calcula el ángulo de giro y la distancia de desplazamiento, de tal modo que el vehículo no toque el obstáculo como se ilustra en la Figura 3.11.

Una vez asegurado el primer paso de evasión, el vehículo tiene que continuar

Figura 3.7: Imagen con rastros de entorno

Figura 3.8: Imagen de la que se obtiene el ancho

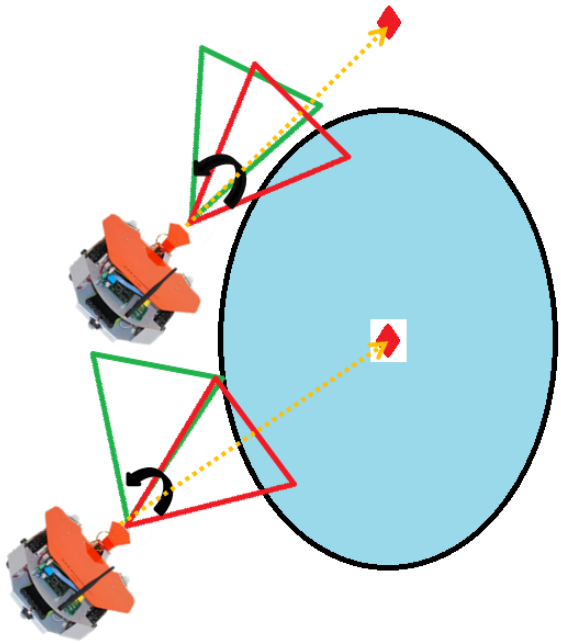

Figura 3.9: Imagen con rastros de entorno

librando el obstáculo y regresar a la ruta, sin embargo la identificación de la posición cuando el robot esta fuera de la trayectoria entrenada no se puede determinar del mismo modo que se hace cuando se sigue la ruta sin obstáculos debido a que no se posee más información visual del mundo circundante. Por lo tanto los siguientes pasos de la evasión se tienen que hacer usando información con la que sí se cuente.

Estos pasos, comienzan haciendo uso de la información generada cuando se creo la ´´Base de Conocimiento.ªl inicio de la navegación autónoma

Dado que cuando se creó la base de conocimiento se conoce la distancia entre cada punto de ruta ya que por consenso siempre es de treinta centímetros y también se pudo estimar el ángulo que existe entre cada posición de la ruta. Es posible hacer una aproximación de la posición hacia la que el robot debe ir y la localización del robot en cada movimiento de evasión. La estimación de la posición en cada movimiento de evasión y el cálculo de la trayectoria para regresar a la ruta se hacen en primer lugar, creando un plano cartesiano en el que el origen es el punto en donde se inicio la evasión, la trayectoria de retorno a la ruta es el vector resultante de la resta entre el vector resultante del avance y el vector resultante de la evasión. La Figura 3.10 y el pseudocódigo siguiente ejemplifican este comportamiento.

Pseudocódigo del calculo de trayectoria de regreso a ruta (VB.Net):

$Pi =$ Posición de inicio de evasión.

$Pr =$ Punto de la ruta al que se regresó después de la evasión

$Cx =$ Componente sobre el eje x del avance entre un punto de ruta y el anterior.

$Cy =$ Componente sobre el eje y del avance entre un punto de ruta y el anterior.

$Cex =$ Componente sobre el eje x del avance entre un punto de evasión y el anterior.

$Cey =$ Componente sobre el eje y del avance entre un punto de evasión y el anterior.

$\theta =$ Ángulo de giro entre un punto de ruta y el anterior.

$D =$ Distancia entre un punto de ruta y el anterior.

$CxT =$ Suma de todas las componentes en x de la ruta.

$CyT =$ Suma de todas las componentes en y de la ruta.

$Ne =$ Numero de movimientos evasivos.

$Dtex =$ Suma de todas las componentes en x de la evasión.

$Dtey =$ Suma de todas las componentes en y de la evasión.

$Drr =$ Distancia para regresar a un punto de ruta.

$Arr =$ Angulo para regresar a un punto de ruta.

Para $i \leftarrow Pi$ hasta Pr hacer

Si $\theta \neq (0\ and\ 90\ and\ 180\ and\ 270)$

$Cx(i) \leftarrow Cos(\frac{\theta(i)}{D(i)})$

$Cy(i) \leftarrow Sen(\frac{\theta(i)}{D(i)})$

Si no entonces

$Cx(i) \leftarrow D(i)\ \forall\ \theta\ \epsilon\ \{0, 180\}$

$Cy(i) \leftarrow D(i)\ \forall\ \theta\ \epsilon\ \{90, 270\}$

Fin Si

$CxT \leftarrow CxT + Cx(i)$

$CyT \leftarrow CyT + Cy(i)$

Fin Para

Para $i \leftarrow 1$ hasta Ne hacer

$Dtex \leftarrow Dtex + Cex(i)$

$Dtey \leftarrow Dtey + Cey(i)$

Fin Para

$Drr \leftarrow \sqrt{(CxT - Dtex)^2 + (CyT - Dtey)^2}$

$$Arr \leftarrow sin^{-1}(\frac{sen90 * (CyT - Dtey)}{Drr})$$

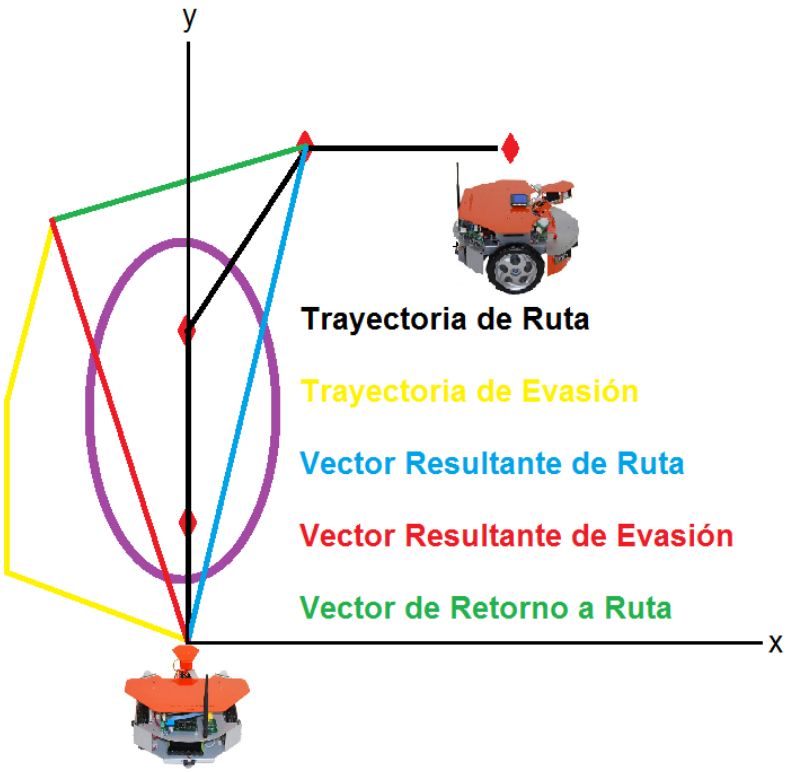

Figura 3.10: Ilustración del modo en que se determina la trayectoria para regresar a la ruta.

El comportamiento descrito anterior mente se repite cada vez que el vehículo realiza un paso para evadir los obstáculos, si el obstáculo fuera pequeño; ejecutando la detección del obstáculo, la primara evasión y el cálculo de la trayectoria de regreso, seria suficiente para que el robot regresara a la ruta. Pero como la mayor parte del tiempo, los obstáculos no son tan pequeños, fue necesario programar una rutina para poder rodear obstáculos que tuvieran formas irregulares y grandes en comparación con el robot y sus avances.

Para hacer que el robot pudiera evadir obstáculos más grandes e irregulares, lo que se necesitó fue hacer una buena determinación de hacia dónde se tiene que orientar el robot cuando intenta regresar a la ruta, y el modo en que se tiene que

generar la trayectoria de evasión. En primer lugar se determina el punto hacia donde el robot debe orientarse para intentar regresar a la ruta, Se considera un punto objetivo al punto de ruta con la distancia minima a la posición actual del robot. En segundo lugar para continuar generando la ruta de evasión, se consideran los sensores de rango frontales para que en cada intento de regresar a la ruta, los sensores indiquen si hay espacio libre para regresar o si se tiene que girar para intentar regresar después de una avance como se ejemplifica en la Figura 3.11.

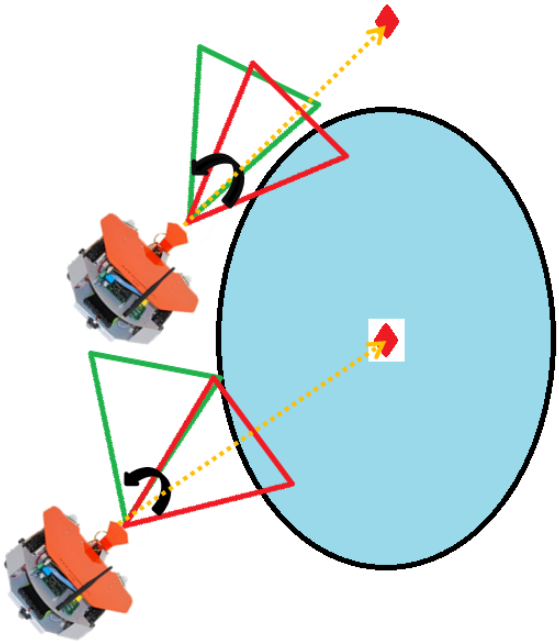

Figura 3.11: Ilustración del modo en que se usan los sensores para definir el avance

Para usar los sensores de rango en la evasión de los obstáculos, se definió un rango seguro de la lectura de cada uno de los sensores frontales y se programó una función para que el robot girara a una velocidad constante mientras la lectura de sus sensores estuvieran fuera del el rango de seguridad para posteriormente avanzar. La rutina que se programó toma en cuenta las características físicas de la plataforma móvil para calcular el ángulo girado, a continuación se muestra su pseudocódigo.

Pseudocódigo dela función programada para girar hasta que los sensores del robot indiquen lo contrario (VB.Net):
$Srf =$ Lectura en sensores de rango frontales

$Rs =$ Rango seguro para desplazamiento
$Ag =$ Ángulo girado
$Lca =$ Lectura de los codificadores antes del giro
$Lcd =$ Lectura de los codificadores antes del giro
$Rll =$ Radio de las llantas del robot
$Sell =$ Separación entre las llantas del robot
$Ppv =$ Pulsos por vuelta
Mientras $Srf < Rs$ hacer
    Girar a una velocidad constante
Fin Mientras

$$Ag \leftarrow= \frac{2 * (Lcd - Lca) * Rll * 360}{Sell * Ppv}$$

Una vez comenzada la evasión, es necesario que el vehículo se ajuste a las condiciones del obstáculo y sea capaz de regresar a la ruta.

Para este fin se sigue la estrategia de la Figura 3.12.

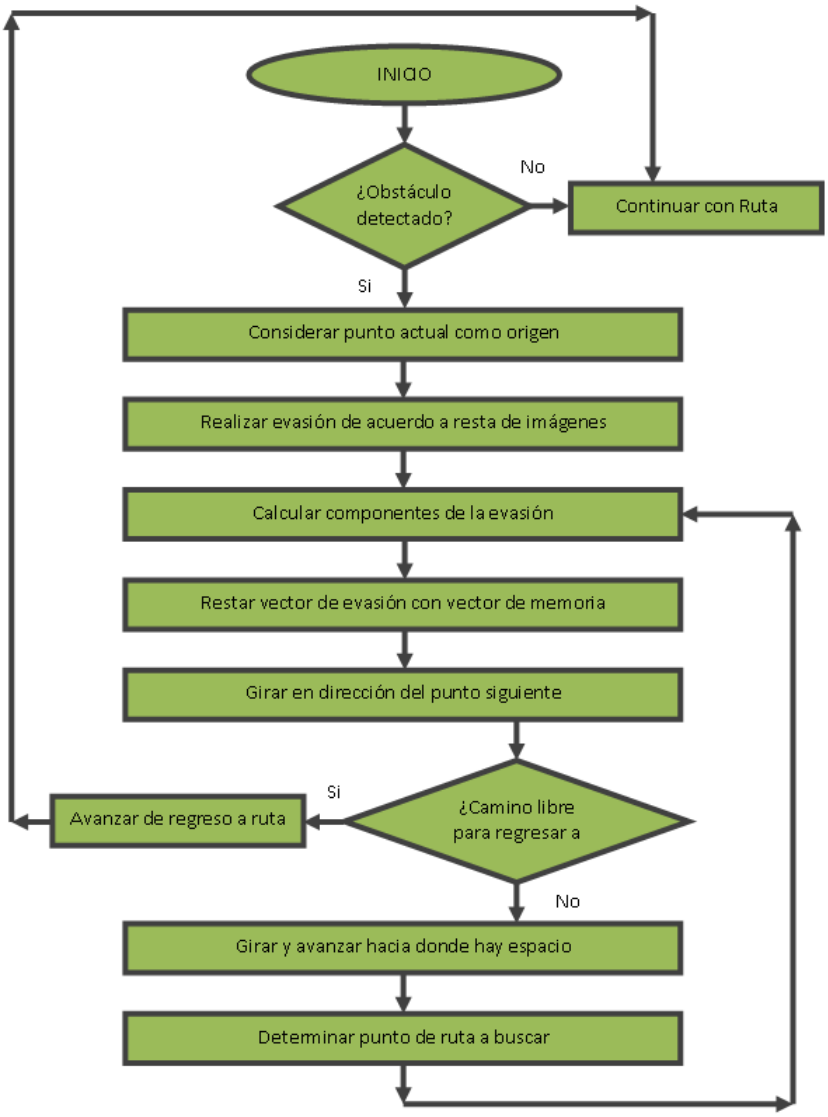

Figura 3.12: Diagrama de flujo de pasos para evasión completa

CAPÍTULO

4

Resultados

Durante las pruebas de navegación se observó que uno de los principales motivos por los que la plataforma no era capaz de regresar a su ruta fue el error acumulado en profundidad y orientación después de seguirse una trayectoria larga o después de terminar una evasión, asi como diferencias de iluminacion y cambios en el entorno.

4.1 Pruebas de Navegación con Modificación de Parámetros

La primera prueba que se realizó fue seguimiento de ruta sin modificaciones de entorno entre el entrenamiento y la navegación con lo que se pudo obtener un error de posición promedio por cada método como se ve en la Figura4.1 y en la Figura4.2.

La segunda prueba que se realizó fue el seguimiento de ruta larga para verificar si los dos metodos logran terminar exitosamente Figura4.3.

La tercera prueba que se realizó fue el seguimiento de ruta con una desviación en la posición inicial de veinte centímetros a la izquierda y de veinte centímetros a la derecha como se ve en la Figura4.4.

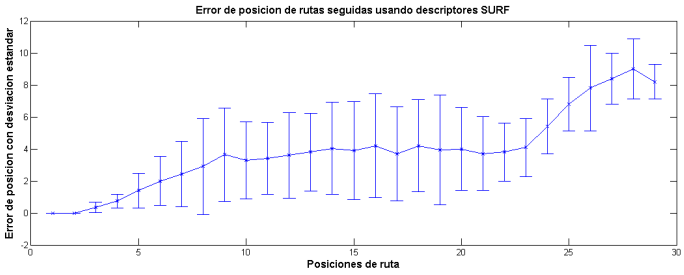

Figura 4.1: Gráfica de errores de posición en cinco corridas

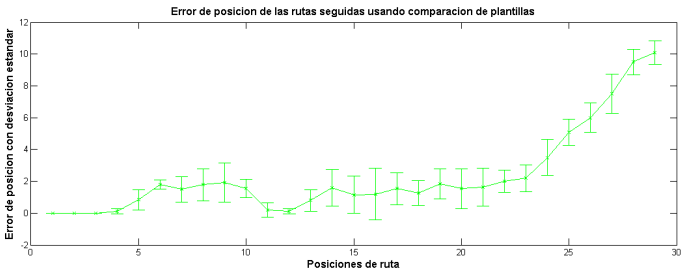

Figura 4.2: Gráfica de errores de posición en cinco corridas

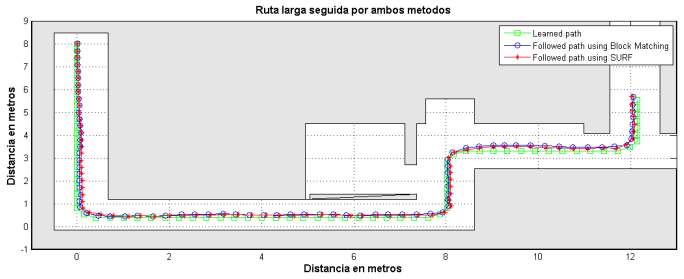

Figura 4.3: Gráfica de errores de posición en cinco corridas

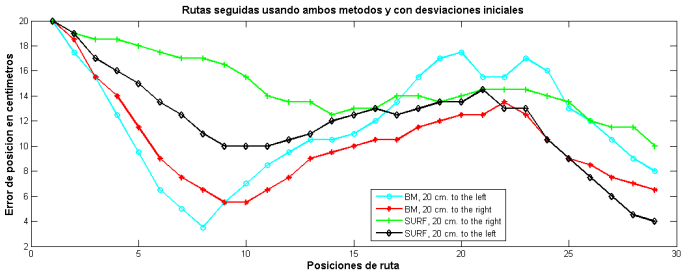

Figura 4.4: Gráfica de errores de posición durante una trayectoria que inicio con un error de posición de 20 cm para ambos métodos

En la cuarta prueba se modificó el entorno de la navegación con algún objeto que no obstruyera la ruta, En la Figura4.6 se muestra el impacto del objeto sobre el el vector de correlación y en la Figura4.5 se muestra en impacto que tiene el objeto en la determinación de la ruta usando descriptores SURF

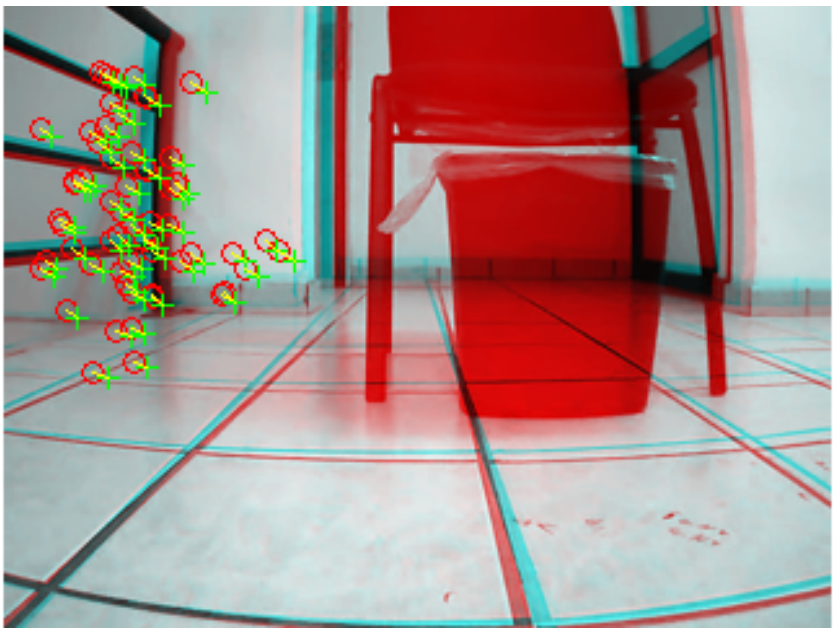

Figura 4.5: Efecto en el uso de descriptores SURF de un objeto que modifica el entorno

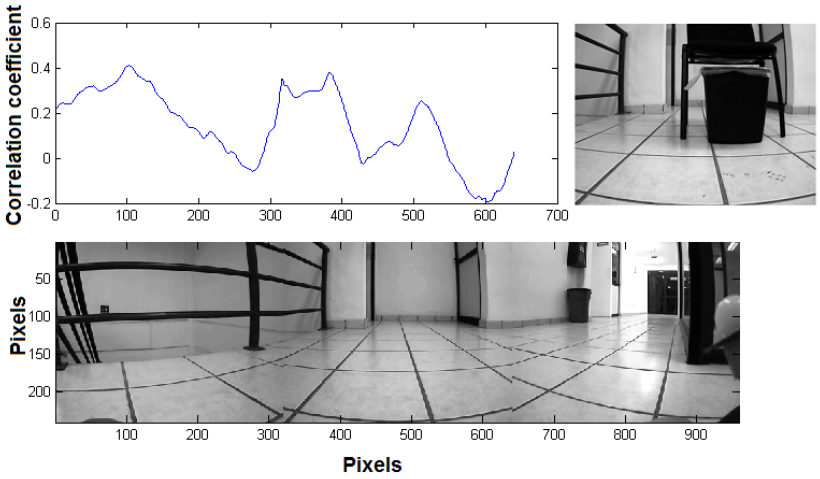

Figura 4.6: Efecto en el uso de comparación de plantillas de un objeto que modifica el entorno

CAPÍTULO

5

Conclusiones y trabajo futuro

5.1 Conclusiones de las pruebas realizadas

Como se puedo observar en las pruebas del capitulo anterior, cuando las condiciones del entorno son constantes, el uso de descriptores SURF y el uso de comparación de plantillas entrega resultados muy similares. Sin embargo cuando se modifica el entorno, el resultado es muy favorable para el método de los descriptores.

Por lo tanto el algoritmo final usa el método de los descriptores SURF para el seguimiento de ruta y para el retorno después de la evasion, ya que es mas robusto a cambios de iluminación, oclusiones parciales del ambiente, y es menos susceptible al error de posición que se acumula normalmente.

5.2 Publicaciones y ponencias resultado del trabajo de investigación

A partir del trabajo realizado, se generaron una ponencia:

- Pedro Alejandro De Jesús González,*Algoritmo de visión basado en apariencia para navegación autónoma con capacidad de evadir obstáculos*, Primer Encuentro Nacional de Matemáticas Aplicadas a la Ciencias Computación y la Electrónica (ENMATCE), Facultad de Ciencias de la Computación (FCC), Benemérita Universidad Autónoma de Puebla (BUAP), Junio 2014, Puebla, Pue.-México, Ponente.

5.3 Trabajo futuro

Como posibles líneas de investigación para lograr que el robot navegue únicamente con vision artificial propongo:

* Usar un algoritmo de segmentación de suelo.

* Usar una cámara omnidireccional para tener mas información del entorno y menos perturbaciones en cuanto al cambio de foco y de iluminación en la base de datos.

* Investigar el problema de la estimación de pose con cinco puntos de referencia.

* Investigar la extracción de pistas del entorno, como son la identificación de paredes, identificación de horizontes, calculo de flujo óptico, estimación del espacio,etc.

De acuerdo a los objetivos establecidos al inicio de este trabajo de tesis y con los resultados obtenidos durante su proceso, se concluye que se han cumplido los objetivos planteados.

Bibliografía

[1] Guoli Song, Jianda Han, Yiwen Zhao, Chen Chen, and Zhongxin Yao. Dynamic tracking of minimally invasive spine surgery robot. In *Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference on*, pages 1462–1467. IEEE, 2013. 1

[2] Qun Zhang, Pey Yuen Tao, Aswin Thomas Abraham, Brice Rebsamen, Chenguang Yang, and Shuzhi Sam Ge. Mission-oriented design: A fully autonomous mobile urban robot. In *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2010 IEEE Conference on*, pages 261–266. IEEE, 2010. 1

[3] Seiichi Arima, S Shibusawa, N Kondo, and Jun Yamashita. Traceability based on multi-operation robot; information from spraying, harvesting and grading operation robot. In *Advanced Intelligent Mechatronics, 2003. AIM 2003. Proceedings. 2003 IEEE/ASME International Conference on*, volume 2, pages 1204–1209. IEEE, 2003. 2

[4] Sungsik Huh, Unghui Lee, Hyunchul Shim, Jong-Beom Park, and Jong-Ho Noh. Development of an unmanned coal mining robot and a tele-operation system. In *Control, Automation and Systems (ICCAS), 2011 11th International Conference on*, pages 31–35. IEEE, 2011. 2

[5] Christian Mandel, Kai Huebner, and Tilman Vierhuff. Towards an autonomous wheelchair: Cognitive aspects in service robotics. In *Proceedings of Towards Autonomous Robotic Systems (TAROS 2005)*, pages 165–172. Citeseer, 2005. 2

[6] Jong-Hwan Kim, Brian Keller, and Brian Y Lattimer. Sensor fusion based seek-and-find fire algorithm for intelligent firefighting robot. In *Advanced Intelligent Mechatronics (AIM), 2013 IEEE/ASME International Conference on*, pages 1482–1486. IEEE, 2013. 2

[7] Nan Wu, Yunfeng Lou, Nan Sun, Wei Wang, and Harutoshi Ogai. Automatic driving system by small electric vehicle for elderly person. In *SICE Annual Conference (SICE), 2012 Proceedings of*, pages 232–235. IEEE, 2012. 2

[8] Jong-Hann Jean and Jheng-Liang Wang. Development of an indoor patrol robot based on ultrasonic and vision data fusion. In *Mechatronics and Automation (ICMA), 2013 IEEE International Conference on*, pages 1234–1238. IEEE, 2013. 3, 5

[9] JM Maja, T Takahashi, ZD Wang, and E Nakano. Real-time obstacle avoidance algorithm for visual navigation. In *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, volume 2, pages 925–930. IEEE, 2000. 5

[10] Guilherme N DeSouza and Avinash C Kak. Vision for mobile robot navigation: A survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(2):237–267, 2002. 10

[11] Johann Borenstein, Liqiang Feng, and HR Everett. *Navigating mobile robots: systems and techniques*. AK Peters, Ltd., 1996. 10

[12] Taha Z. and Mat-Jizat J.A. Landmark navigation in low illumination using omnidirectional camera. *9th International Conference on Digital Object Identifier*, pages 262–265, 2012. 11

[13] Zhang Yingchun, Cao Juan, Lou Lu, and Su Bohong. Appearance-based mobile robot navigation using omnidirectional camera. *Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on Digital Object Identifier*, pages 2357 – 2361, 2012. 11

[14] Ke Wang, Lijun Zhao, and Ruifeng Li. Mobile robot map building in eigenspace a pea-based approach. In *Control and Decision Conference (CCDC), 2013 25th Chinese*, pages 3199–3203. IEEE, 2013. 11

[15] K Chitra, S Dinesh, D Mishra, V Brinda, VR Lalithambika, and BM Kumar. 3d information retrieval for visual odometry system of planetary exploration rovers - a stereo vision approach. *Advances in Computing, Communications and Informatics (ICACCI)*, pages 354 – 360, 2013. 12

[16] Konstantin Schauwecker and Andreas Zell. On-board dual-stereo-vision for autonomous quadrotor navigation. 12

[17] C Hsia. A stereo vision based self localization system. 2013. 12

[18] Cheng Pei Tsai, Chin Tun Chuang, Ming Chih Lu, Wei Yen Wang, Shun Feng Su, and Shyang Lih Chang. Machine vision based obstacle avoidance system for robot system. In *System Science and Engineering (ICSSE), 2013 International Conference on*, pages 273–277. IEEE, 2013. 12

[19] M Yamada, Chi-Hsien Lin, and Ming-Yang Cheng. Vision based obstacle avoidance and target tracking for autonomous mobile robots. In *Advanced Motion Control, 2010 11th IEEE International Workshop on*, pages 153–158. IEEE, 2010. 12

[20] Vladimir J Lumelsky and Tim Skewis. Incorporating range sensing in the robot navigation function. *Systems, Man and Cybernetics, IEEE Transactions on*, 20(5):1058–1069, 1990. 13

[21] Vladimir J Lumelsky and Alexander A Stepanov. Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2(1-4):403–430, 1987. 13

[22] Ishay Kamon, Ehud Rivlin, and Elon Rimon. A new range-sensor based globally convergent navigation algorithm for mobile robots. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 1, pages 429–435. IEEE, 1996. 13

[23] Iwan Ulrich and Johann Borenstein. Vfh+: Reliable obstacle avoidance for fast mobile robots. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 2, pages 1572–1577. IEEE, 1998. 13

[24] Iwan Ulrich and Johann Borenstein. Vfh[*]: local obstacle avoidance with look-ahead verification. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 3, pages 2505–2511. IEEE, 2000. 13

[25] JAVIER Minguez and LUIS Montano. Nearness diagram navigation (nd): A new real time collision avoidance approach. In *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, volume 3, pages 2094–2100. IEEE, 2000. 14

[26] Roland Philippsen and Roland Siegwart. Smooth and efficient obstacle avoidance for a tour guide robot. In *ICRA*, pages 446–451. Citeseer, 2003. 14

[27] Christian Schlegel. Fast local obstacle avoidance under kinematic and dynamic constraints for a mobile robot. In *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, volume 1, pages 594–599. IEEE, 1998. 14

[28] Oliver Brock and Oussama Khatib. High-speed navigation using the global dynamic window approach. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 1, pages 341–346. IEEE, 1999. 14

[29] Nak Yong Ko and Reid G Simmons. The lane-curvature method for local obstacle avoidance. In *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, volume 3, pages 1615–1621. IEEE, 1998. 14

[30] Maher Khatib, H Jaouni, Raja Chatila, and Jean-Paul Laumond. Dynamic path modification for car-like nonholonomic mobile robots. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 4, pages 2920–2925. IEEE, 1997. 14

[31] Ayanna Howard, Homayoun Seraji, and Barry Werger. Fuzzy terrain-based path planning for planetary rovers. In *Fuzzy Systems, 2002. FUZZ-IEEE'02. Proceedings of the 2002 IEEE International Conference on*, volume 1, pages 316–320. IEEE, 2002. 14

[32] Emidio Di Giampaolo and Francesco Martinelli. Mobile robot localization using the phase of passive uhf-rfid signals. 2013. 15

[33] Han Wang, Wei Mou, Minh Hiep Ly, MWS Lau, Gerald Seet, and Danwei Wang. Mobile robot ego motion estimation using ransac-based ceiling vision. In *Control and Decision Conference (CCDC), 2012 24th Chinese*, pages 1939–1943. IEEE, 2012. 15

[34] Jan Elseberg, Dorit Borrmann, and Andreas Nuchter. 6dof semi-rigid slam for mobile scanning. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1865–1870. IEEE, 2012. 15

[35] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, pages 404–417. Springer, 2006. 21

[36] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 21

[37] Inc Dr Robot. User manual, 2013. 24

[38] Inc Dr Robot. Data sheet, 2013. 24

[39] Inc Dr Robot. Sdk api reference manual, 2013. 24

[40] R Mariano, Jiménez María I., V Lara, I Alberto, V Juan J., and C Myriam. Matlab com integration for engineering applications, 2013. 24

[41] Inc Microsoft. Ayuda de visual basic. net, 2013. 24