Instituto Politécnico Nacional

Centro de Investigación en Computación

«Localización de un robot móvil empleando memorias asociativas ALFA-BETA»

TESIS

Que para obtener el grado de Maestro
en Ciencias de la Computación

PRESENTA:

Viridiana G. Hernández Herrera

Directores de Tesis

Dr. Amadeo José Argüelles Cruz
Dr. David Jaramillo Vigueras

MÉXICO, D. F. DICIEMBRE DE 2012

SIP-14 bis

INSTITUTO POLITÉCNICO NACIONAL
SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

ACTA DE REVISIÓN DE TESIS

En la Ciudad de México, D.F. siendo las 12:00 horas del día 12 del mes de noviembre de 2012 se reunieron los miembros de la Comisión Revisora de la Tesis, designada por el Colegio de Profesores de Estudios de Posgrado e Investigación del:

Centro de Investigación en Computación

para examinar la tesis titulada:

"Localización de robots móviles empleando memorias asociativas Alfa-Beta"

Presentada por el alumno:

HERNÁNDEZ	HERRERA	VIRIDIANA GUDELIA
Apellido paterno	Apellido materno	Nombre(s)

Con registro: B 1 0 1 6 5 3

aspirante de: **MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN**

Después de intercambiar opiniones los miembros de la Comisión manifestaron ***APROBAR LA TESIS***, en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias vigentes.

LA COMISIÓN REVISORA

Directores de Tesis

Dr. Amadeo José Argüelles Cruz

Dr. David Jaramillo Vigueras

Dr. Cornelio Yáñez Márquez

Dr. Oscar Camacho Nieto

Dr. Luis Octavio López Leyva

M. en C. Sandra Dinora Orantes Jiménez

PRESIDENTE DEL COLEGIO DE PROFESORES

Dr. Luis Alfonso Villa Vargas

INSTITUTO POLITECNICO NACIONAL
CENTRO DE INVESTIGACION
EN COMPUTACION
DIRECCION

INSTITUTO POLITÉCNICO NACIONAL

SECRETARIA DE INVESTIGACIÓN Y POSGRADO

CARTA DE CESIÓN DE DERECHOS

En la ciudad de México el día 27 del mes de Noviembre del año 2012, la que suscribe **Viridiana G. Hernández Herrera** alumna del Programa de Maestría en Ciencias de la Computación con un registro B101653, intelectual del presente trabajo de Tesis bajo la dirección de Dr. Amadeo José Argüelles Cruz y de Dr. David Jaramillo Vigueras, cede los derechos del trabajo intitulado ***"Localización de un robot móvil empleando memorias asociativas Alfa-Beta"***, al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del autor y/o directores del trabajo. Este puede ser obtenido escribiendo a la siguiente dirección robotbailarin@hotmail.com o vhernandezhe@ipn.mx . Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.

Viridiana G. Hernández Herrera

(nombre y firma)

Resumen

En este trabajo de tesis se diseña e implementa un algoritmo que ofrece una solución competitiva en la tarea de la localización de un robot móvil. El algoritmo se basa en los fundamentos matemáticos de las memorias asociativas de tipo Alfa-Beta, las cuales forman parte del conjunto de las técnicas que comprenden la línea de investigación de la Inteligencia Artificial (IA).

Con el propósito de evaluar el desempeño de las memorias asociativas Alfa-Beta en el problema de localización de robots móviles se realizan dos pruebas experimentales, en la primera prueba se hace uso de uno de los bancos de datos de UCI (Machine Learning Repository), mientras que en la segunda prueba se emplea un simulador de robots. En ambas pruebas se obtienen resultados competitivos, que incluso superan el desempeño mostrado por otros clasificadores de patrones presentes en la literatura científica contemporánea.

De modo que el presente trabajo de investigación propone el empleo de memorias asociativas Alfa-Beta en el problema de localización de robots móviles, brindando así una solución innovadora, eficaz y competitiva.

Abstract

The aim of this work is to implement an algorithm that provides a competitive solution to the task of localization a mobile robot. The algorithm is based on mathematical foundations of the associative memories of type Alfa-Beta, which forms part of a set of techniques that are include in the line research of artificial intelligence (AI).

To evaluate the performance of the Alpha-Beta associative memories in the problem of localization mobile robots, there were perform two experimental tests. The first test is done using one database of UCI (Machine Learning Repository), while in the second it was used a robots simulator. In both tests were obtained competitive results that even exceed the performance shown by others classifiers patterns presented at the contemporary scientific literature.

So this study shows that the use of Alpha-Beta associative memories in the problem of localization of mobile robots, offers a innovative, effective and competitive solution.

Agradecimientos

Este trabajo de investigación no sería posible sin el apoyo de muchas personas e instituciones que en mucho colaboraron con la culminación de este trabajo y es por ello que deseo agradecer a:

El Instituto Politécnico Nacional, que me ha abierto las puertas hacia increíbles experiencias profesionales y personales, por darme la oportunidad de crecer y aprender en sus aulas, con maestros de alta calidad por eso y más soy orgullosamente politécnica.

Al Centro de Investigación en Computación CIC-IPN por guiarme en el aprendizaje de una nueva área de conocimiento, siempre brindando calidad de enseñanza y servicio en los procesos administrativos, por eso doy gracias a mis profesores y a todas las secretarias en donde verdaderamente se preocupan por ti.

Al grupo Alfa-Beta por ser abierto a propuestas y por compartir el conocimiento que se tiene de las Memorias Asociativas Alfa-Beta piedra angular del presente trabajo de investigación.

A el Dr. Amadeo Argüelles por ser un director de tesis integral, brindándome siempre su apoyo y confianza, permitiéndome innovar impulsando siempre mi creatividad. Gracias por ser un director de tesis excepcional.

A el Dr. David Jaramillo por ser una persona que admiro profundamente y a la que debo lo que tengo y lo que soy, su humildad y bondad infinita me alienta todos los días a seguir adelante e inspirarme en sus conocimientos y enseñanzas para no rendirme y pensar siempre que la vida nos regala una nueva misión para superarnos y ayudar a que el mundo sea un poco mejor cada día. Gracias Dr. David por las largas horas invertidas es mi formación profesional, personal y espiritual. No defraudaré todo cuanto me ha enseñado.

Agradecimientos especiales

A Dios, por darme la oportunidad de vivir y por estar conmigo en cada paso que doy, por fortalecer mi corazón e iluminar mi mente y por haber puesto en mi camino a aquellas personas que han sido mi soporte y compañía durante todo el periodo de estudio. Por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos, además de su infinita bondad cuidado y amor.

A mi mamita Lety, por ser mi ejemplo a seguir, por ser mujer guerrera y demostrarme día con día que para superar la adversidad se necesita valentía, paciencia y amor. Gracias mamita por ser la madre perfecta para mi, por tus regaños, consejos y apapachos. Te amo y te agradezco infinitamente todo el amor y tiempo que me has dado, por que todo lo bueno que hay en mi es gracias a ti.

A mi papa Raúl por las noches de desvelos haciendo tareas, por creer siempre en mí, por las caminas hacia la escuela, por ser una alma noble y humilde, papi siempre seré tu GRADIENTE. Te amo papá.

A mis hermanas y hermano, por enseñarme lo que significa una familia, por regalarme recuerdos increíbles con cada uno de ellos. Cynthia, Thalia y Brandon saben que los quiero mucho para mi siempre serán las niñas y mi bebe.

A Lili, Jimmy y Carlos por darme ánimos cuando más lo necesitaba, apoyarme y consolarme, ustedes saben que los quiero mucho y los guardo en mi corazón, arriba los C++.

A Abril por las horas de desvelo haciendo tarea en la casa, por las risas e incoherencias que surgían de las innumerables horas de estudio.

A la Sra. Lourdes que con su infinita bondad es cuna de los desprotegidos, por enseñarme el sentido de la entrega y la entereza hacia la familia, por ser mi motivación en los tiempos más adversos. Gracias y con el corazón haré todo lo que este a mi alcance para ser fiel a mi convicción de amar, cuidar y respetar a su familia ayer, hoy y siempre.

Al amor de mi vida y compañero de batallas, doy gracias a Dios por haberte puesto en mi camino y ser parte de mi vida llenando mi existir de felicidad con esos ojitos hermosos. Nene te amo 1800.

Índice general

Índice de algoritmos

Índice de figuras

Glosario

Accesibilidad	Son todos aquellos aspectos o características propias del entorno de navegación del robot, que pueden afectar las mediciones tomadas por el sensor en cuestión.
Asertividad	Evaluación para medir la eficiencia de un algoritmo de reconocimiento de patrones en el que asignan el valor de 1 a un acierto (Verdadero) y un 0 a un error (Falso).
Autonomía	El término autonomía proviene del griego "auto" que significa propias y "nomos" que significa ley.
Clase	Conjunto de patrones que comparten características en común.
Clasificación supervisada	También es conocida como clasificación con aprendizaje, en este tipo de problemas ya se encuentran definidas las clases, y éstas cuentan con algunos objetos previamente clasificados.
Dimensión	Son las unidades de medición del sensor y su representación en el espacio, de modo que la representación corresponde a: un plano (medición escalar), dos planos (vectorial), tres o más planos (mediciones n-dimensional).
Error	Se define como la diferencia entre el valor medido y el valor verdadero. Los errores pueden ser de dos tipos: Los determinísticos o sistemáticos que pueden preverse, calcularse o incluso eliminarse mediante calibraciones y compensaciones y los errores aleatorios que no pueden ser previstos, pues dependen de causas desconocidas o estocásticas.
Exactitud	Es el grado de inexactitud del sensor, es decir el rango de proximidad de la medición tomada por el sensor con respecto al valor real de la medición. En ocasiones la exactitud puede ser expresada como una variación de error máximo y mínimo (+/-), aunque también puede verse representado como un porcentaje de la salida a rango total.
Formatos de los datos	Es la estructura de los datos entregados por el sensor. Es de gran importancia conocer la forma de los datos ya que de ello depende su adecuada interpretación.

Odometría	Es una técnica de posicionamiento que emplea información de sensores para obtener una aproximación de la posición real a la que se encuentra un sistema móvil, en un determinado instante, respecto a un sistema de referencia inicial.
Patrón	Representación numérica del objeto que se desea clasificar.
Rango de operación	Es el conjunto de valores comprendidos entre los limites (Superior e Inferior) que es capaz de medir el sensor.
Rapidez de respuesta	Se refiere al tiempo mínimo entre muestra y muestra, por lo que esta estrechamente relacionado con la tasa de muestreo del sensor.
Rasgo	Propiedad, factor, característica, etc. que se toma en cuenta para estudiar los objetos.
Reconocimiento de patrones.	Rama del conocimiento, de carácter multidisciplinario, cuyo objeto de estudio son los procesos de identificación, caracterización, clasificación y reconstrucción sobre conjuntos de objetos o fenómenos, así como el desarrollo de teorías, tecnologías y metodologías relacionadas con dichos procesos.
Reconocimiento	Proceso de clasificación de un objeto en una o más clases.
Robot móvil	Se integra de un conjunto de dispositivos físicos y sistemas computacionales, los cuales que le permiten desplazarse en uno o más entornos espaciales (tierra, agua, aire, espacio estelar, entre otros) con la finalidad de realizar determinadas tareas.
Robótica móvil	Disciplina interdisciplinaria que encargada del estudio de los robótica móvil.
Sensor	Son dispositivos cuya función principal es obtener información. Gracias a los sensores los robots son capaces de recibir información de las condiciones internas y externas a él.

Nomenclatura

AGV	Vehículos Guiados Automáticamente
AGV	Vehículos Guiados Automáticamente
BRA	Asociación Británica de Robótica
CCR	Concurrency and Coordination Runtime
DSS	Decentralized Software Services
GNSS	Sistemas Globales de Posicionamiento por Satélite
GPS	Sistema de Posición Global
IA	Inteligencia Artificial
ICC	Centro Instantáneo de Curvatura
ICR	Centro Instantáneo de Rotación
JIRA	Asociación Japonesa de Robots Industriales
MLP	Multilayer Perceptrón
MRDS	Microsoft Robotics Developer Studio
PPR	Pulsos Por Revolución
RF	Radio Frecuencia
RFID	Identificación por Radiofrecuencia
RIA	Robot Institute of America
TA	Timing Advance
TDOA	Time Difference Of Arrival
UCI	Machine Learning Repository
VPL	Visual Programming Language

Capítulo 1

Introducción

1.1. Importancia del problema de la localización

La llamada "*Revolución Tecnológica*" trajo consigo un impulso constante en el mejoramiento de hardware y software, lo que ha constituido la base del crecimiento y perfeccionamiento de los actuales procesos industriales, además de representar un apoyo medular en el avance y generación de nuevos descubrimientos científicos de áreas diversas.

Una de las tantas áreas que se han visto favorecidas con el auge tecnológico es la robótica, trayendo consigo la disminución en orden potencial de la brecha que existe entre los robots futuristas de las películas de ciencia ficción y la realidad de los robots de servicio que sean capaces de realizar deberes en casa que el ser humano no desea o no puede realizar.

Los robots de servicio pretenden hacer la vida del hombre más cómoda y segura, constituyendo sin lugar a dudas un nicho de mercado potencialmente explotable, lo cual ha despertado el interés de la comunidad científica y por consecuencia existen diversos esfuerzos colaborativos orientados en la búsqueda de soluciones a los problemas que actualmente impiden que un robot de servicio sea apto para realizar tareas de forma eficiente y efectiva.

En consecuencia las investigaciones encaminadas en el perfeccionamiento de la robótica de servicio son numerosas, lo que se asocia con el hecho de que dichas investigaciones son derivadas de campos disciplinarios diversos, se tiene por lo tanto que la robótica es el producto del esfuerzo colaborativo de muchas disciplinas tales como la mecánica, electrónica, eléctrica, control, programación, telecomunicaciones, entre otras; es por ello que la robótica en general es considerada un estudio interdisciplinario.

Ahora bien, *¿Porque la robótica industrial presenta un desarrollo más activo que la robótica de servicio?,* se debe al hecho de que los costos de un robot son en este momento poco accesibles y uno de los mercados que son capaces de invertir esa cantidad de capital es precisamente la industria y es por esa razón, que la mayor parte de las investigaciones se vierten en robots que sean capaces de realizar tareas de manufactura. Gracias a este acontecimiento se puede decir que, hoy por hoy, la robótica industrial es una historia de éxito contundente, confirmándose al comparar los resultados antes y después del empleo de robots en tareas monótonas, repetitivas y/o peligrosas dentro de los procesos industriales.

Considerando que el capital de inversión, el avance tecnológico y el interés científico es directamente proporcional al progreso de los robots de servicio, es de importancia tener un contexto general de cómo es que la robótica de servicio representa actualmente un potencial campo de estudio para la comunidad científica, por lo que en los siguientes párrafos se brinda un breve recorrido de los acontecimientos que

dieron lugar al surgimiento de las investigaciones encaminadas en el diseño y construcción de robots de servicio.

El incremento acelerado de la demanda de robots por parte de industrias de diferentes sectores aunado al avance tecnológico de dispositivos electrónicos, electromecánicos y desarrollo de software, incentivaron el interés de los investigadores en la posibilidad de diseñar sistemas complejos, que permitan que un robot sea más rápido, preciso y fácil de programar. La consecuencia directa de este avance dio como resultado el siguiente paso en dirección a la automatización industrial, flexibilizando la producción con el nacimiento de células de manufactura bajo mando robótico.

La aplicabilidad de estos robots en su mayoría manipuladores es muy exitosa, ya que son los encargados de realizar tareas monótonas, repetitivas y/o peligrosas, ofreciendo así, grandes beneficios para la industria, no obstante la mayoría de los robots utilizados en las industria se encuentran en estaciones de trabajo fijas, y los movimientos están restringidos por los grados de libertad del robot, esto significa una gran desventaja ya que presentan poca flexibilidad para desplazarse y adaptarse a las diferentes células de trabajo.

Dentro de las primeras soluciones al problema de desplazamiento de los robots industriales se encuentran los Vehículos Guiados Automáticamente (AGV por sus siglas en inglés); estos se transportan de un punto a otro en cuestión de segundos. Los AGV se desplazan por medio de rieles que proporcionan al robot una guía para seguir trayectorias fijas bajo tiempos y distancias precisas. Desafortunadamente el empleo de este tipo de robots es poco útil en un entorno que no este previamente acondicionado con mecanismos que sirvan de guía para el desplazamiento del robot, por esta razón su alcance pocas veces va mas allá del ámbito industrial.

En este punto *¿Cuál es el desafío que se debe enfrentar para que los robots de servicio sean una realidad?,* lo primero es superar las restricciones de navegación de los robots industriales, dando lugar a un nuevo tipo de robots llamados “robots móviles”. En el capítulo II se aborda más a detalle cómo es que a lo largo del tiempo los esquemas tradicionales han tenido que adaptarse o cambiarse por completo, para lograr que los robots sean capaces de realizar tareas en lugares complejos y cambiantes tal como los hogares, en donde el entorno de navegación esta lleno de obstáculos, de modo que es necesaria una programación que se ajuste constantemente dependiendo del tipo de tareas a realizar, además de tener la habilidad de generar soluciones óptimas frente a los inconvenientes que puedan presentarse durante la ejecución de la tarea.

Una solución factible en el problema de la navegación de los robots, es la posibilidad de implementar sensores en lugar de rieles, lo que otorga al robot la posibilidad de obtener información de su entorno. Ahora bien, el sistema de sensores debe ir acompañado de una robusta programación, que permita equipar al robot no solo de herramientas útiles para la extracción de información del ambiente, sino que al mismo tiempo debe estar preparado para la correcta interpretación de lo que hay a su alrededor.

Entonces se puede decir que los robots móviles son el pasado inmediato de los robots de servicio, por lo que su estudio e innovaciones tecnológicas están íntimamente relacionados con el desarrollo de los robots de servicio del futuro. Es la razón por la que los siguientes párrafos y prácticamente toda la tesis están enfocados en el estudio de los robots móviles.

Como ya se mencionó los sensores son el puente entre los AGV y los robots móviles. Haciendo una analogía, los sensores en los robots móviles tienen una función parecida a la que tienen los cinco sentidos (oído, vista, gusto, tacto y olfato) en el ser humano ya que tanto los sensores como los sentidos les permiten tener un cuadro estructural de su entorno.

Ahora bien, las tareas de un robot móvil útil no se limitan a detectar y evadir obstáculos, ya que un robot móvil debe estar dotado de cierto grado de ***inteligencia***, considerando que sus labores no son solo una tarea repetitiva como en los robots industriales, incluso al realizar la misma tarea puede ser que el robot deba tomar decisiones que lleven a soluciones totalmente diferentes, esto se debe a que el robot móvil esta sujeto a las condiciones del ambiente de navegación y rara vez éstas son estáticas.

Es por ello que surgen una serie de alternativas que pretenden en menor o mayor grado proporcionar autonomía a los robots móviles y es ahí, cuando el software juega un papel importante, razón por la cual en la actualidad se pueden encontrar numerosas investigaciones dedicadas al diseño de programas encauzados en el mejoramiento del control de movimientos del robot, dichos algoritmos centran su interés en problemas como la precisión, la coordinación y el equilibrio del robot.

En segundo lugar y de manera paralela, los programas también deben resolver problemas derivados de la interacción del robot con su entorno, ya que se debe considerar que el robot rara vez es un ente único dentro del ambiente de trabajo. De forma que para cumplir con una tarea compleja al robot no le es suficiente con gobernar de manera óptima sus movimientos, ya que también le son necesarios algoritmos capaces de extraer datos de un conjunto de sensores dedicados a recaudar datos del entorno y un requisito más para lograr su autonomía es el buen uso de esos datos, por lo que la ***"Inteligencia Artificial"*** entra en acción proporcionando algoritmos que pueden ser capaces de aprender de esos datos convirtiéndolos en información útil.

Luego entonces, se puede decir que para lograr autonomía en la navegación de un robot móvil son necesarios los siguientes pasos:

Primero. Contar con algoritmos que permitan el control de todos los dispositivos dedicados al movimiento del robot, el cual debe cumplir de manera eficiente con los parámetros que son requeridos en la realización de su(s) tarea(s).

Segundo. Que dichos algoritmos también posibiliten al robot para recibir, procesar, interpretar y aprender de los datos provenientes de sensores, con el objetivo de lidiar mejor con los inconvenientes del entorno.

Pero estos dos únicos pasos no son sufrientes para que un robot móvil sea un robot de servicio, ya que aun falta un paso más: ***Dotar al robot de la capacidad de tomar "decisiones" por sí mismo.***

Sin embargo, en este momento aún se esta lidiando con encontrar algoritmos óptimos que permitan resolver los desafíos derivados del primer y segundo paso, y aunque ya existen esfuerzos importantes resolviendo el tercer paso, estos algoritmos se ven limitados por la falta de eficacia y eficiencia del primer y segundo paso.

Y es precisamente el interés de este trabajo de tesis contribuir en la solución del paso número dos, abordando el tema de la "auto-localización de robots móviles" que es un sub-problema de la navegación de robots móviles. La localización da respuesta a la pregunta *¿Dónde estoy?*, lo cual constituye un conjunto de información auxiliar que es de gran ayuda en la determinación de las trayectorias que el robot puede seguir al desplazarse de un punto a otro, o bien permite almacenar un historial de los lugares que ya han sido visitados por el robot y aprender de ello, tal como lo hacemos los seres humanos, lo que quiere decir que un robot puede ser capaz de aprender de los lugares en donde ha estado, generando información relevante como su posición con respecto a otros lugares u objetos, además generar trayectorias para poder regresar a alguno de estos sitios. Es la razón por la que el presente trabajo orienta sus esfuerzos a fin de proporcionar una solución viable para la fase de auto-localización.

1.2. Motivación

El estudio, diseño y construcción de un robot móvil con usos prácticos es sin duda todo un reto, ya que la robótica móvil es una disciplina que se auxilia de los fundamentos teórico-prácticos de diversas áreas de estudio, por lo que el conjunto de problemas que surgen al diseñar y construir un sistema robótico deben ser resueltas de manera colaborativa, trabajando paralelamente con los desafíos concernientes a las partes físicas, de control e inteligencia del robot y es justamente esta simbiosis la clave que permitirá evolucionar de un robot móvil a un robot de servicio con la capacidad de realizar múltiples tareas de forma sistemática y precisa, pero que al mismo tiempo *tenga la posibilidad de decidir de forma certera las acciones pertinentes cuando se le presenten acontecimientos inesperados que representen un impedimento para cumplir con su objetivo.*

Los robots de servicio es uno de los nichos de mercado más atractivos, no obstante la robótica móvil aún presenta múltiples problemas que deben resolverse antes para contar con un robot de servicio con características ideales y uno de esos problemas es la *localización*. El problema de la localización constituye un desafío interesante, ya que el hecho de que el robot móvil conozca su posición y orientación en el ambiente de navegación le brinda la posibilidad de dar un paso más, incorporando información útil que le permita tomar decisiones propias y así poder lograr su ***autonomía***.

El problema de la localización está íntimamente relacionado con los datos suministrados por los diferentes sensores que pueden integrarse en un robot móvil y el proceso de conversión de los datos en información útil que le permita determinar su ubicación en un ambiente de navegación. El propósito de atacar el problema de localización es colaborar en el problema de la navegación autónoma de un robot dentro de escenarios no controlados y en donde existe un amplio espacio de oportunidades para generar soluciones que conjuguen el hardware y software, dando lugar a una apasionante línea de investigación.

1.3. Justificación

El uso de robots móviles está justificado primordialmente en las múltiples aplicaciones que puede tener un robot móvil realizando tareas molestas o arriesgadas para el trabajador humano, algunas de las más importantes y rentables son:

Medicina. Debido a las características de este tipo de aplicaciones, los robots utilizados en medicina desarrollan su trabajo a bajas velocidades y fuerza, disponen de una cinemática compleja y de una interfaz de usuario de alto nivel de inteligencia.

Limpieza. Actualmente, en la tecnología de las máquinas de limpieza ya existen algunos avances significativos, en donde los robots móviles, funcionan sin operador y dotados de sensores de posición que les permite recorrer espacios y sortear los obstáculos con cierto grado de destreza y eficiencia.

Construcción. Las tareas de construcción que se podrían realizar son: colocación de ladrillos y de piezas grandes (vigas), unión de piezas (soldadura, remaches), rellenado, excavación y movimiento de tierras, control de calidad, entre muchas otras.

Defensa y seguridad. En esta línea, los avances de la robótica están dando frutos; los robots están siendo dotados de más inteligencia y cuentan con mayor autonomía. Sobre todo en la inspección de objetos sospechosos y desactivación de explosivos suelen usarse robots tele-operados.

Pero para lograr que los robots móviles sean capaces de realizar estas y más tareas de manera eficiente es necesario romper con el esquema de que los robots están hechos para realizar tareas únicas y monótonas, así el *aprendizaje* es una interesante alternativa en la solución del problema de la autonomía, el cual le permitiría a los robots solucionar un problema, e incluso basarse en su experiencia para resolver problemas que previamente se le han enseñado a resolver, y en caso de que el problema no esté dentro del conjunto de entrenamiento, que este pueda encontrar la mejor manera de resolverlo, y si la resolución fue exitosa entonces que guarde la información como un ejemplo de entrenamiento.

Dotar al robot con la habilidad de aprender significa que éste puede ser capaz de dar un paso muy importante, pasando de resolver tareas repetitivas y monótonas (automatización) a resolver tareas complejas de forma autónoma.

1.4. Objetivos

1.4.1. Objetivo General

Implementar un algoritmo haciendo uso de los operadores y propiedades de las memorias asociativas Alfa-Beta, el cual brinde una solución competitiva en el problema de localización en robots móviles.

1.4.2. Objetivos Particulares

1. Proponer dos experimentos que permitan evaluar la asertividad de las memorias asociativas Alfa-Beta en el problema de localización de robots móviles.
2. Programar los algoritmos de aprendizaje y recuperación tomando como base los fundamentos matemáticos de las memorias asociativas de tipo Alfa-Beta, los cuales serán implementados durante la fase de experimentación.
3. Probar la asertividad la memoria asociativa Alfa-Beta previamente programada en las dos pruebas de experimentación propuestas y comparar los resultados obtenidos con los de otros métodos de clasificación de patrones.

1.5. Organización del documento

La organización de la presente tesis esta sustentada en el desarrollo de los conceptos teóricos fundamentales correspondientes al problema de localización de un robot móvil, así como la validación experimental efectuada en cada una de las pruebas propuestas, que tienen la finalidad de determinar la competitividad de las memorias asociativas Alfa-Beta en dicho problema. El documento se encuentra dividido en 5 capítulos los cuales presentan el siguiente contenido.

El **capítulo 1** esta integrado por una breve introducción que enfatiza "la importancia del problema de la localización en los robots móviles", el cual es la motivación de la línea de investigación de la presente tesis. También se integran la justificación y objetivos que se persiguen durante el desarrollo de este trabajo.

En el **capítulo 2** se realiza un trabajo de investigación del estudio del arte, proporcionando una panorámica global de factores que intervienen de forma directa en el problema de localización de robots móviles. Seguidamente se abordan las diferentes configuraciones cinemáticas así como los cálculos correspondientes. Finalmente se presentan algunas soluciones que han sido propuestas para resolver el problema de localización citando a los autores que en la actualidad trabajan bajo dichos contextos.

El **capítulo 3** se concentra en la descripción de los modelos matemáticos y metodologías empleados para el diseño y la implementación de una memoria asociativa Alfa-Beta. Así mismo se describen los bancos de datos necesarios para cada experimento, el equipo utilizado y las herramientas empleadas para el desarrollo de las pruebas experimentales propuestas.

En el **capítulo 4** se documentan los resultados obtenidos de cada uno de los experimentos realizados bajo el modelo propuesto, del mismo modo estos resultados son comparados con los obtenidos en otros métodos probados bajo las mismas condiciones de experimentación. El objetivo es obtener una comparativa de asertividad midiendo los resultados de la memoria asociativa Alfa-Beta con respecto a otros métodos que también pertenecen a la línea de investigación de la Inteligencia Artificial y que son comúnmente empleados en la comunidad científica para la solución del problema de localización

Para terminar, en el **capítulo 5** se presentan las conclusiones, así como una recapitulación del trabajo realizado y un análisis de la técnica propuesta planteando las posibles mejoras en los trabajos futuros que se proponen derivados de este trabajo de investigación.

Capítulo 2

Contexto general del problema de localización

Cualquier *problema*, es sin duda una oportunidad latente para poner en práctica las habilidades y conocimientos que el ser humano ha adquirido, con la única intención de brindar una solución apropiada; para ello es de vital importancia saber cuál es el contexto general que rodea a dicho problema, lo cual representa una ventaja competitiva al contar con las herramientas necesarias para proponer una solución óptima.

Esta idea es la razón por la cual el presente capítulo pretende mostrar los conceptos generales de la robótica móvil, así como las principales circunstancias que hacen de la robótica móvil una línea de investigación en constante crecimiento y por supuesto una área con interesantes retos que pueden ser vistas como grandes oportunidades...

2.1. Introducción a la Robótica

La robótica es una área de investigación interdisciplinaria, la cual se dedica al estudio, diseño, desarrollo y construcción de robots.

No obstante quizás el concepto que hace falta para comprender de mejor forma qué es la **robótica** es definir a su objeto de estudio ***"El Robot"***.

¿Qué es un Robot?

La mayoría de las definiciones que existen en la literatura concerniente a los robots son insuficientes para describir el real alcance de los diferentes tipos de robots actúales, esto es porque en sus inicios la robótica tenía su mayor campo de aplicación en la industria, en donde los robots estaban destinados a la realización de tareas repetitivas y pre-programadas.

Sin embargo, con el tiempo la robótica ha tenido un crecimiento potencial, lo que le ha permitido ampliar su alcance inicial. A continuación se presentan las definiciones de "robot" más citadas, aun debe considerarse que las actuales instituciones encargadas de la estandarización y homologación del significado de las nuevas tecnologías muy probablemente tendrán la labor de modificar estas definiciones.

"Es un dispositivo reprogramable y multifuncional diseñado para mover materiales, piezas, herramientas o dispositivos especializados a través de movimientos programados".

Robot Institute of America (RIA), 1979.

"Un mecanismo reprogramable con un mínimo de cuatro grados de libertad diseñado para manipular y transportar partes, herramientas o implementar manufactura especializada a través de movimientos programados para la ejecución de la una tarea específica de manufactura"

Asociación Británica de Robótica (BRA).

"Robot inteligente: robot que puede determinar su propio comportamiento/conducta a través de sus funciones de senseo y reconocimiento"

Asociación Japonesa de Robots Industriales (JIRA).

La primera vez que se hace uso del término "robot" es en 1920 por Karel Capek [3], el cual tiene sus orígenes en la palabra checa "robota" que significa "trabajo" [4].

Pero el término original de Karel Capek es retomado y reforzado algún tiempo después por Isaac Asimov [5] definiendo el término de "robótica" que es la ciencia que estudia a los robots, y da origen a la concepción de las "***Tres leyes de la Robótica***".

> ***Primera.*** "Un robot no debe dañar a un ser humano o, por inacción, dejar que un ser humano sufra daño".
>
> ***Segunda.*** "Un robot debe obedecer las órdenes que le son dadas por un ser humano, excepto si estas órdenes entran en conflicto con la Primera Ley".
>
> ***Tercera.*** "Un robot debe proteger su propia existencia, en la medida en que esta protección no entre en conflicto con la Primera Ley o la Segunda Ley".

La robótica es una línea de investigación que hoy en día es punta de lanza tecnológica, sin embargo por siglos, el ser humano ha intentado construir máquinas que imiten las partes y comportamiento del cuerpo humano. Por lo cual la robótica ha jugado un papel importante en el desarrollo de la humanidad y su evolución. La cual ha ido siempre de la mano con los avances tecnológicos dependiendo de la época. Por ejemplo en antiguas civilizaciones, como la griega, se hablaba de seres mecánicos con vida que eran movidos por mecanismos construidos con poleas y bombas hidráulicas.

La historia de la robótica, esta estrechamente ligada con las herramientas que el ser humano ha tenido a su alcance para diseñar diferentes mecanismo, las cuales poco a poco permiten la concepción de robots realmente útiles.

Unas de las primeras ciencias que sirvieron de apoyo en el desarrollo de la robótica fueron la mecánica, la hidráulica y la neumática, las cuales permitieron realizar máquinas que aunque con movimientos poco precisos fueron los principios de esta ciencia tecnológica, posteriormente el desarrollo de la eléctrica propicio un impulso importante, ya que las máquinas podían incorporar dispositivos eléctricos que permitían mejorar los mecanismos para lograr movimientos con mayor precisión, así como el control y duración de los mismos.

2.2. Clasificación general de los robots

Las clasificaciones se basan en perspectivas y en características de similitud, lo que hace que por cada perspectiva que se tome exista una clasificación nueva, es por ello que la robótica se ha clasificado de muchas formas, algunas perspectivas que suelen considerarse en la clasificación de la robótica son: generaciones, morfología, aplicación, nivel de inteligencia, entre otras.

La gran mayoría de las clasificaciones de robots citadas en la literatura se encuentran enfocadas en los robots industriales, no obstante existen algunas clasificaciones que contemplan las implementaciones de los robots en otras áreas de aplicación. De forma que en esta sección se presenta una breve descripción de las clasificaciones más importantes que han formado parte de la historia de la robótica.

La clasificación propuesta por T. M. Knasel [6, 7], se basa principalmente en 5 etapas generacionales, por lo que el tiempo juega un papel importante en cada categoría, la idea del autor es ilustrar la evolución de los robots a lo largo de su historia, proporcionando la relación entre las características de los robots y el espectro de la evolución dependiendo de la época. El control es una de las características que se consideran dentro de esta clasificación, la cual se relaciona con el grado de utilidad y flexibilidad del robot y depende de las limitantes del diseño mecánico, así como de la capacidad de los sensores, todo ello representa un nivel de impacto en la aplicabilidad del robot dependiendo de la generación.

La clasificación puede resumirse en la siguiente tabla.

Tabla 2.1: Clasificación de las generaciones de la robótica según T. M. Knasel [6, 7]

Generación	Nombre	Tipo de Control	Grado de movilidad	Usos más frecuentes
Primera Generación (1982)	Pick & place	Fines de carrera, aprendizaje	Ninguno	Manipulación, servicio de máquinas
Segunda Generación (1984)	Servo	Servocontrol, Trayectoria continua, progr. condicional	Desplazamiento por vía	Soldadura, pintura
Tercera Generación (1989)	Ensamblado	Servos de precisión, visión, tacto	Guiado por vía	Ensamblado, Desbardado
Cuarta Generación (2000)	Móvil	Sensores inteligentes	Patas, Ruedas	Construcción, Mantenimiento
Quinta Generación (2020)	Especiales	Sensores inteligentes	Andante, Saltarín	Militar, Espacial

La Asociación de Robots Japonesa (JIRA) presenta una perspectiva que se basa en el nivel de inteligencia que los robots demuestran en la realización de diversas tareas. JIRA ha clasificado a los robots dentro de seis clases dependiendo su nivel de inteligencia:

1. **Dispositivos de manejo manual.** Operados y manipulados por un operario humano.
2. **Robots de secuencia arreglada.** Son aquellos que están programados para seguir una trayectoria fija.
3. **Robots de secuencia variable.** En donde un operador puede modificar la secuencia fácilmente.

4. **Robots regeneradores.** El operador humano conduce el robot a través de la tarea.
5. **Robots de control numérico.** El operador alimenta la programación del movimiento, hasta que se enseñe manualmente la tarea.
6. **Robots inteligentes.** Los cuales pueden entender e interactuar con cambios en el medio ambiente.

2.3. Robótica Móvil

En las clasificaciones presentadas en la sección anterior de forma directa o indirecta se contempla a la robótica móvil, como una de las líneas de investigación de la robótica y es justamente esta línea de investigación en la que se enfocará el trabajo de investigación de la presente tesis.

La robótica móvil surge de la creciente necesidad de robots con un amplio grado de movilidad. Gran parte de los robots hoy en día se encuentran concentrados en la industria, principalmente en células de manufactura por lo que la mayoría de los robots industriales son fijos y realizan tareas pre-programadas, sin embargo existen muchas otras posibles aplicaciones para los robots, no obstante es necesario que los robots puedan moverse libremente, es por esto que la robótica móvil es una vertiente de la robótica que surge de la necesidad del desplazamiento de los robots.

La robótica móvil representa un reto diferente a los robots industriales, ya que es necesario considerar problemas como potencia, precisión, consumo de energía, dimensión, localización, navegación, planeación de rutas, modo de locomoción, morfología, etc. Todas estas necesidades intrínsecas de la robótica móvil, que hacen que este campo de estudio se haya desarrollado, acoplado o incluso fusionado un conjunto de conocimientos de diferentes áreas de estudio, con la finalidad de avanzar poco a poco e ir solucionando aspectos necesarios para que un robot móvil fuese funcional y que por supuesto sea aplicado de forma exitosa en la realización de diversas tareas.

Para realizar el diseño y fabricación de un robot móvil es necesario tener noción de un conjunto de conocimientos, ya que la robótica móvil es una disciplina tecnológica interdisciplinaria, a continuación se muestran algunos de los conocimientos necesarios para realizar un robot móvil con aplicaciones útiles para el ser humano.

Tabla 2.2: Áreas de apoyo a la investigación de la robótica móvil

Conocimiento	**¿Por qué?**
Matemáticas y Física	Es necesario realizar modelos dinámicos y cinemáticos. Es difícil mencionar un nivel aproximado en los conocimientos de estas áreas, sin embargo se debe tener en cuenta que es proporcional a la complejidad de la tarea del robot móvil.
Electrónica, Eléctrica y Mecánica	Actualmente existen carreras de ingeniería dedicadas al estudio de cada una de estas áreas de conocimiento, no obstante son muy necesarias para poder realizar un buen diseño, que asegure la precisión, potencia y peso necesarios, y al mismo tiempo que el consumo de energía sea óptimo.
Informática y Computación	Estos conocimientos han dado a los robots móviles la posibilidad de controlar y optimizar la utilización de los recursos disponibles. Además existen algunos programas de simulación que nos permiten probar los algoritmos antes de ser implementados en un robot real.

La tabla 2.2 muestra solo un compensado de las ciencias básicas en las que se apoya la robótica, no obstante esta es una área de investigación multidisciplinaria por lo que en la actualidad dependiendo de la aplicación de un robot móvil intervienen otras áreas de conocimiento como: la biología, anatomía, medicina, astronomía, entre otras.

Hoy en día existen muchas ventajas tecnológicas para el desarrollo de la robótica móvil, algunas de las razones que hacen de la robótica móvil un área de investigación dinámica son el acelerado incremento de tecnología de hardware y software así como los costos accesibles de los mismos. Algunos factores tecnológicos que propician el desarrollo de la robótica móvil son:

- **Sensores.** Mayor precisión y menor consumo, la ventaja es significativa si se considera el único recurso que el robot utiliza para conocer las condiciones internas y externas a él.
- **Procesadores.** Alto rendimiento, bajo consumo y múltiples procesadores en un solo integrado, lo que permite migrar de programación seriada a paralela y hacer múltiples tareas al mismo tiempo.
- **Simuladores.** Esta clase de programas permiten realizar un diseño preliminar del robot así como probar su comportamiento bajo diferentes algoritmos, sensores, mecanismos, entre otros. Todo ello con tan solo cambiar los dispositivos y algoritmos de forma virtual.
- **Las tecnologías de comunicación.** Un gran avance es la estandarización de protocolos de comunicación, además de ser tecnologías que pueden ser ocupadas en espacios cerrados y abiertos, lo que nos permite realizar la transferencia de información entre robots móviles, con la idea de no solo tener un solo robot realizando una tarea, ya que las comunicaciones proporcionan en la actualidad la posibilidad de robots trabajando en equipo por un mismo objetivo.
- **Robots programables.** Esta clase de robots permiten simplificar los diseños mecánicos, eléctricos y electrónicos, la finalidad de esta clase de robots es facilitar la aplicación de algoritmos en un modelo real.
- **Kits de ensamblado fácil.** Tienen el objetivo de acercar e incentivar a niños y jóvenes en el área de robótica móvil, de forma que representan una manera sencilla de integrar la mecánica, eléctrica, y electrónica; además de que existen algunos de ellos que también presentan software muy amigable para su programación.

2.3.1. ¿Qué es un robot móvil?

"Un robot móvil es un conjunto de dispositivos físicos y sistemas computacionales, los cuales que les permiten desplazarse en uno o más entornos espaciales (tierra, agua, aire, espacio estelar, entre otros) para la realización de determinadas tareas".

Los robots móviles pueden ser valorados por su capacidad para:

- **Moverse (locomoción).** Esto depende del medio de desplazamiento, así como de su estructura electro-mecánica y de control; de manera general se evalúan aspectos como rapidez, precisión, consumo de energía, etc.
- **Percepción del medio ambiente que lo rodea.** Está relacionado con la capacidad de los dispositivos sensoriales implementados en el robot y el procesamiento de los datos extraídos de los sensores, con el propósito de obtener el más apropiado concepto del entorno de navegación del robot móvil.
- **Comunicación**. Está basada en la comunicación que existe entre el robot y los otros dispositivos o bien con otros robots, con el propósito de la transferencia de información. Un ejemplo en donde

es prioridad un buen nivel de comunicación es en sistemas multi-robot, ya que los robots trabajan en tareas conjuntas para lograr un mismo fin y es necesaria una coordinación lo mas precisa posible.

- **Razonamiento**. Es decir su capacidad de decisión; el robot móvil en un ambiente real debe enfrentarse a una serie de problemas que tendrá que solucionar por medio de la toma de decisiones. Este concepto es el que da pie a los ***Robots Móviles Autónomos.***

2.3.2. Tipos de robots móviles

La literatura presenta pocas clasificaciones concretas dirigidas a los robots móviles, de ese modo esta sección intenta definir una clasificación conveniente que sirva como marco conceptual para el presente trabajo de tesis.

La presente clasificación está basada en una perspectiva morfológica, locomoción y medio ambiente en el que se desenvuelve un robot móvil, sin embargo existen algunas clasificaciones en la literatura con perspectivas diferentes, por ejemplo la ***aplicación*** (Servicio manufacturero, Servicio de mantenimiento, Servicio directo de humanos), y el ***modo de operación*** (Remotos, Tele-operados, semi-autónomo, autónomos).

La clasificación que se ha realizado para el presente trabajo de tesis se muestra en la figura 2.1, la cual pretende mostrar una categorización de los robots móviles basada en la perspectiva del entorno espacial en el que se desplazan, a la que también se puede llamar ambiente de navegación, siendo el espacio físico en el que el robot debe ser capaz de desplazarse con el propósito de realizar su tarea objetivo. Se ha propuesto este tipo de clasificación ya que permite ver la estrecha relación que existe entre la morfología de un robot móvil y medio ambiente al que es expuesto para cumplir con su tarea. De esta forma los robots que navegan en un ambiente aéreo deben cumplir con características intrínsecas de su ambiente para poder volar (ser ligero, tener un diseño aerodinámico, tipos de combustible, etc.), mientras que por otro lado un robot terrestre no tiene que cumplir con esas restricciones aéreas, pero tendrá que resolver los problemas que conlleva un desplazamiento en un terreno firme, con lo cual se puede imaginar que el robot terrestre podría ser más pesado y su forma al no tener que romper el viento puede ser más irregular o incluso amorfa, sin embargo el robot tendrá que considerar un diseño que le permita quizás, solucionar el problema del desplazamiento en terrenos de formas irregulares, en espacios estrechos, con diferentes niveles de texturas, entre otras consideraciones.

Ahora bien, cabe mencionar que se ha considerado al espacio estelar como uno de los posibles entornos de navegación, ya que las condiciones del espacio pueden ser muy diversas. Un ejemplo puede ser un robot de exploración espacial el cual tendrá que desplazarse en la superficie lunar, entonces se puede pensar que esta clase de robots son similares a los terrestres, sin embargo no es así, la diferencia radica en el hecho de que las condiciones de la luna no son las mismas que las que hay el planeta, por lo que un robot espacial puede requerir especificaciones especiales tales como, soportar temperaturas extremas, considerar la utilización de una fuente de energía recargable con medios a su alcance, o bien un diseño que le permita permanecer en una lugar o desplazarse con menos fuerza gravitacional que la tierra, entre otras muchas consideraciones. Todo lo anterior permite pensar que las condiciones en el espacio influyen en la forma y locomoción de un robot, por lo que se ha concluido que este tipo de clasificación es viable y permite dar un contexto general de los robots de servicio y su morfología y locomoción dependiendo de su entorno de navegación.

Debido a que el presente trabajo de tesis encamina su estudio a los robots móviles en un medio de navegación terrestre, la clasificación (véase la figura 2.1) se ha ampliado permitiendo mostrar los tipos de morfología de robots móviles más comunes en este medio.

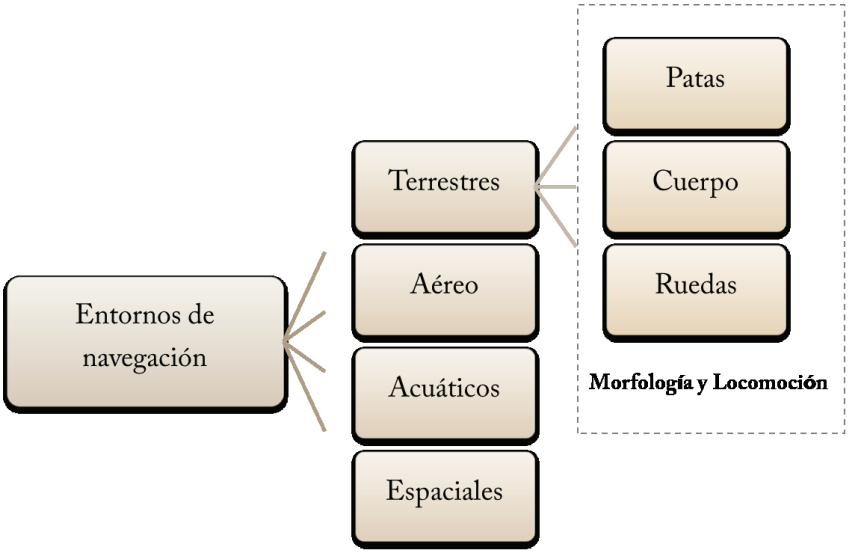


Figura 2.1: Clasificación de Robots Móviles.

- **Patas.** En este rubro pueden encontrarse a los robots que tienen articulaciones en forma de patas y su función principal es lograr el desplazamiento del robot. Algunos ejemplos son los robots: bípedos, cuadrúpedos, hexápodos, entre otros.
- **Cuerpo**. Son comúnmente llamados ápodos, el término ápodo significa "carente de patas". Estos robots utilizan su propio cuerpo para desplazarse, la forma de desplazamiento es a menudo parecida a los gusanos o las serpientes. Esta clase de robots posen una estructura capaz de doblarse y adoptar la forma del terreno por el que se desplazan.
- **Ruedas**. Como su nombre lo indica la forma en la que esta clase de robots se desplaza es por medio del giro de las ruedas. Son los robots más comunes ya que poseen una dinámica y cinemática menos compleja que los presentados anteriormente.

2.3.3. Robot Móvil Autónomo

Hasta este punto se tiene definido el concepto de «robot móvil», de modo que solo faltaría agregar a esta definición el concepto de ***«autonomía»***.

El término autonomía proviene del griego "auto" que significa propias y "nomos" que significa ley. En un contexto orientado a la robótica, la autonomía esta relacionada con la capacidad de realizar tareas sin la necesidad de la ayuda humana, esto quizás podría parecer algo sencillo sin embargo, en la realidad la autonomía representa todo un reto ya que se integra de problemas particulares complejos, por lo que brindar cierto grado de autonomía a un robot es actualmente una de las líneas de investigación en desarrollo. La idea de la autonomía recae en dos premisas principalmente, la primera es la complejidad de la tarea misma y la segunda es la capacidad de decisión del robot.

Ya que hoy en día la autonomía es uno de los principales problemas que presenta la robótica móvil, también representa una de las principales motivaciones del presente trabajo de tesis ya que en la actualidad gracias a la aportación de la Inteligencia Artificial (IA) se cuenta con algoritmos que son capaces de realizar tareas de aprendizaje de patrones, lo cual abre un mundo de posibilidades para diseñar robots que sean capaces de realizar tareas desempeñadas originalmente por el hombre.

En la tabla 2.3 puede verse una comparativa de funciones entre un ser humano, un robot móvil autónomo y un robot móvil.

Tabla 2.3: Comparativo de funciones entre el ser humano, robot móvil autónomo y robot móvil.

	Ser Humano	**Robot Móvil Autónomo**	**Robot Móvil**
¿Cómo se mueve?	Está formado por esqueleto que proporciona la forma y sistemas orgánicos que le permiten moverse.	Cuenta con una estructura mecánica y sistemas eléctricos y electrónicos, así como un sistema de control que le permiten moverse.	Cuenta con una estructura mecánica y sistemas eléctricos y electrónicos, así como un sistema de control que le permiten moverse.
¿Cómo obtiene información del entorno?	Tiene 5 Sentidos: gusto, tacto, olfato, vista y oído.	Por medio de sensores.	Por medio de sensores.
¿Cómo resuelve los problemas?	Tiene un cerebro que funciona neuronalmente.	Utiliza algoritmos de IA los cuales le permiten tomar decisiones por si mismo.	Él no toma decisiones por sí mismo, ya que solo puede realizar las acciones que previamente le han sido definidas.

Para explicar la autonomía, primero se debe decir que el ser humano es capaz de realizar muchas tareas de forma autónoma. Si se observa la tabla comparativa 2.3 se puede ver que el único cambio que hay entre un ***robot móvil*** y un ***robot móvil autónomo*** se encuentra en la respuesta a la pregunta de ¿Cómo resuelve los problemas?, ya que mientras el robot móvil solo sigue las indicaciones que le fueron establecidas previamente en su programación, el robot móvil autónomo cuenta con algoritmos que le permiten evaluar la información y tomar una decisión ante determinado problema.

Algunos autores llaman a esto inteligencia, sin embargo sí se analizan las características necesarias que se requieren para que el robot cumpla con una tarea objetivo sin ayuda del ser humano, se puede intuir que el problema de la inteligencia en robots puede verse reducida a la toma certera de decisiones.

Lo que lleva a la siguiente pregunta ¿Qué se necesita para tomar una decisión?, para tomar una decisión es preciso tener la información necesaria sobre el problema, el segundo paso podría ser que a partir de la información que se tiene con respecto al problema, el robot debe ser capaz de evaluar las opciones viables que se tienen para resolver el problema, finalmente que el robot tenga la capacidad de seleccionar la opción que mas le convenga para la realización de la tarea objetivo.

2.4. Morfología y locomoción de los robots móviles terrestres con ruedas

Retomando la figura 2.1 en donde se puede observar un panorama general del área de conocimiento de los robots móviles, este análisis permite realizar una primera acotación del trabajo de estudio de la presente tesis, en donde se ha seleccionado como tema de estudio “Robots Móviles Terrestres con Ruedas”, ya que las ruedas son la solución más simple y eficiente para desplazar a un robot, además de que ofrecen buena estabilidad en terrenos de superficies planas y en superficies irregulares, entre otras muchas ventajas.

Dicho esto, en esta sección se realiza un análisis de las distribuciones de ruedas más utilizadas en la literatura referida al área de investigación, así como un estudio general de los modelos cinemáticos que pertenecen a cada configuración de ruedas.

Uno de los pasos más importantes en el diseño de un robot de este tipo, es la selección de las ruedas así como, la distribución de las mismas, ya que cada configuración influye directamente en los modelos cinemáticos empleados para el desplazamiento del robot, ahora bien la dificultad de cada distribución de ruedas también depende del número y de la posición de las rudas que sea seleccionada, lo cual se ve fuertemente reflejado en los algoritmos utilizados para que el robot se mueva en determinados espacios. De forma que la selección de la configuración de las ruedas representa una serie de problemas intrínsecos de cada tipo de configuración y que influyen directamente en el la complejidad del algoritmo empleado para el desplazamiento, así como en la eficiencia energética y en muchas ocasiones la forma y dimensiones del robot.

2.4.1. Tipos de ruedas

La diferencia principal entre los tipos de ruedas radica en los ejes de dirección y materiales para antiderrape. Uno de los componentes del vector de dirección es la orientación por lo que es común que algunos autores se refieran a la dirección como las orientaciones que posee una rueda.

Ahora bien existen muchos tipos de ruedas, sin embargo durante el presente trabajo de tesis se dará especial atención a los tipos de ruedas que típicamente son utilizados en las estructuras de robots móviles terrestres y que cumplen con las siguientes premisas:

1. El robot móvil posee una estructura rígida es decir que no tiene segmentos flexibles o con movimiento independiente.
2. El sentido de giro de las ruedas debe estar directamente en contacto con una superficie plana o con inclinaciones poco prolongadas.

Los tipos de ruedas que se consideran bajo estas premisas son:

- **Rueda fija.** Como su nombre lo indica es una estructura fija que no presenta articulación de dirección. Como se muestra en la figura 2.2.a.
- **Rueda con Orientación Centrada.** Este tipo de rueda cuenta con articulación de dirección, por lo que también se le suele denominar como rueda orientable, ya que se orienta con respecto a la estructura del robot y el eje de rotación se encuentra alineado con el centro de la rueda. Como se muestra en la figura 2.2.b.
- **Rueda con Orientación Descentrada (Castor).**También llamada rueda loca, tiene articulación de dirección y también es orientable con respecto a la estructura del robot, la diferencia con respecto a la anterior es que su eje de rotación NO se encuentra alineado con respecto a el centro de la rueda. Como se muestra en la figura 2.2.c.
- **Rueda fija con Rodillos.** Existen muchas variantes de nombres comunes de este tipo de rueda (suecas, universal, mecanum, entre otros), este tipo de rueda son fijas con respecto a la estructura del robot y tiene rodillos entre la rueda y la superficie, de modo que poseen cierta orientación con respecto a la rueda. Tal como se muestra en la figura 2.2.d.

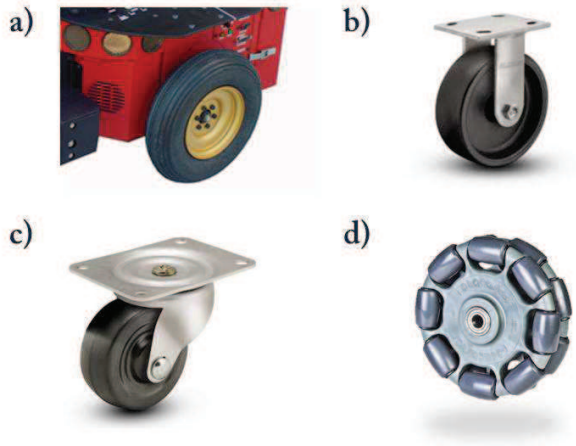


Figura 2.2: Tipos de ruedas. a) Rueda fija b) Rueda con Orientación Centrada c) Rueda con Orientación Descentrada d) Rueda Sueca.

2.5. Configuraciones típicas de los robots móviles terrestres con ruedas

Existen múltiples modelos de configuración de ruedas y estudios al respecto, en esta sección se presentan las clases de configuraciones más comúnmente utilizadas en el diseño y construcción de robots móviles con ruedas.

Para abordar el presente tema se definirá al "Centro Instantáneo de Rotación" el cual será utilizado a lo largo del presente tema.

Centro Instantáneo de Rotación (ICR). También llamado Centro Instantáneo de Curvatura (ICC), es definido como el punto en que se cruzan los ejes de todas las ruedas, dicho en otras palabras es el punto en el que gira el robot en un instante dado. En la figura 2.3 se pueden apreciar algunos ejemplos.

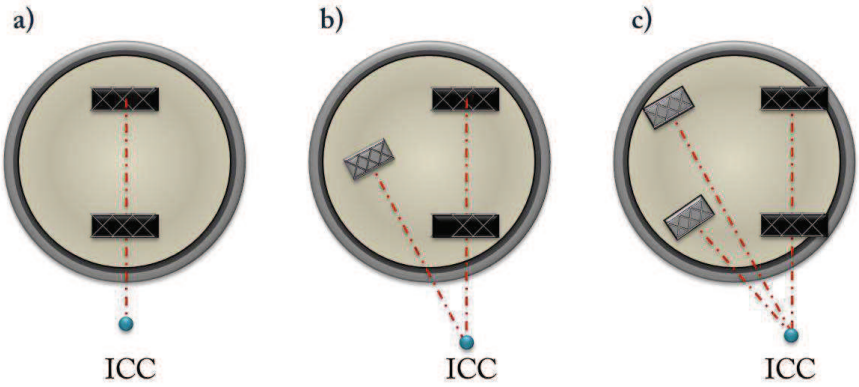


Figura 2.3: ICC en a) Configuración diferencial, b) Configuración Ackerman, c) Configuración triciclo.

2.5.1. Diferencial

La posición de las ruedas en modo diferencial es una de las configuraciones más sencillas en cuanto a locomoción se refiere. Para comenzar con la descripción de esta configuración se debe aclarar que existen algunas variantes de este tipo, así que primeramente se describirán los aspectos generales de la configuración diferencial y posteriormente se hará mención de las variantes más comunes de esta configuración.

Este tipo de configuración está formado por dos ruedas cada una de ellas con un motor, las cuales brindan tracción al robot y sus centros se encuentran alineados como se puede observar en la figura 2.4.

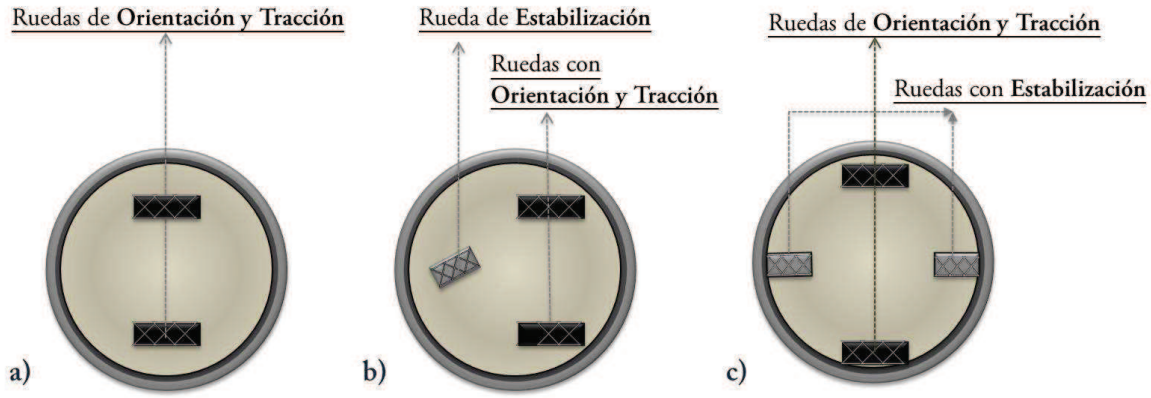


Figura 2.4: Tipo de configuración diferencial. a) Configuración Diferencial Básica. b) Configuración Diferencial Triángulo. c) Configuración Diferencial Diamante.

Ahora bien, ya que el modelo básico solo cuenta con dos ruedas, la estabilidad del robot suele ser un problema en esta configuración, por lo cual existen algunas variantes para minimizar dicho problema. La solución es agregar ruedas al diseño, así que las dos variantes más comunes son:

Triángulo. Se le añade una rueda, de forma que las ruedas de tracción deben estar alineadas y la rueda añadida se acomoda de forma que las tres ruedas formen un triángulo, esto con el propósito de brindar estabilidad al robot como se puede observar en la figura 2.4.b.

Es común confundir la "configuración diferencial triángulo" con la "configuración de triciclo" estudiada en la sección 2.5.2, ya que como se puede observar la formación de las ruedas del triciclo y diferencial triángulo es la misma, sin embargo la diferencia entre ellas es que la rueda delantera del triciclo marca la dirección y tracción del robot y en el tipo de diferencial triángulo las dos ruedas traseras son las encargadas de la tracción y dirección, mientras que la tercera rueda solo brinda estabilidad al robot.

Diamante. Se le añade dos ruedas, este esquema sigue presentando las dos ruedas de tracción y dirección alineadas mientras que las otras son distribuidas de forma que se forma un rombo o un cuadrado como se muestra en la figura 2.4.c.

Ventajas y desventajas de la configuración Diferencial

Ventajas. Buena maniobrabilidad ya que es capaz de ir en línea recta, girar sobre sí mismo y trazar curvas.

Desventajas. La velocidad de los motores debe ser ajustada mecánicamente, ya que los motores pueden desajustarse. En su modelo clásico se debe tratar el problema de equilibrio del robot. Este tipo de configuración no presenta buenos resultados en terreno irregulares.

Los cálculos cinemáticos para este tipo de configuración de ruedas se presentan en el apéndice A.

2.5.2. Triciclo Clásico

Este tipo de configuración solo tiene tres ruedas las cuales se ordenan en forma de triángulo. La rueda delantera es de tipo orientable y tiene dos funciones la primera función es brindar dirección y la segunda es dar tracción al robot, mientras las dos ruedas traseras son fijas y no están acopladas a ningún motor y suelen ofrecer estabilidad y soporte.

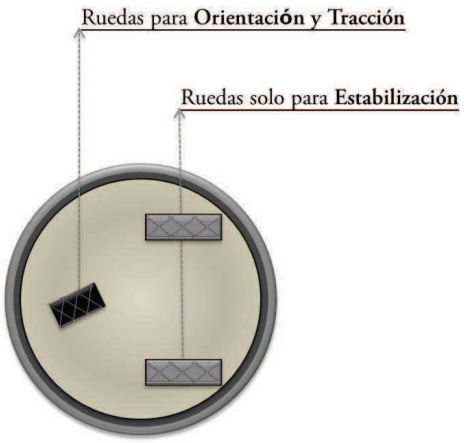


Figura 2.5: Orientación y mecanismos de tracción de configuración de triciclo clásico.

Ventajas y desventajas de la configuración Triciclo Clásico

Ventajas. Su principal ventaja es que no hay deslizamiento de las ruedas

Desventajas. No se pueden hacer giros de $\pm 90°$, su cinemática es complicada, además el centro de gravedad del robot tiende a moverse lejos de la rueda delantera en terrenos con inclinación, lo que ocasiona perdida de tracción, cuando el robot se encuentra de subida.

Los cálculos cinemáticos para este tipo de configuración de ruedas se presentan en el apéndice A.

2.5.3. Ackerman

Quizás este tipo de configuración sea más conocida ya que es la utilizada en los automóviles. Esta configuración es una variante de la configuración de triciclo. La configuración tipo Ackerman está constituida por 4 ruedas, 2 de ellas son colocadas en la parte trasera con tracción y las 2 restantes en el frente para controlar la dirección (véase la figura 2.6).

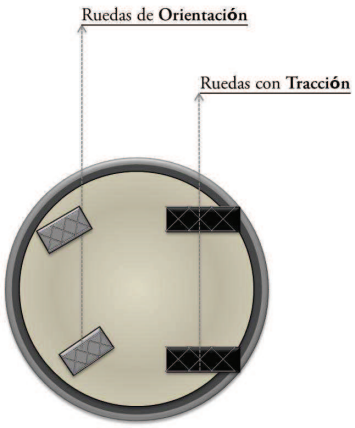


Figura 2.6: Orientación y mecanismos de tracción de configuración ackerman.

Ventajas y desventajas de la configuración Ackerman

Ventajas. Proporciona buena estabilidad, presenta un buen sistema de tracción y su facilidad para seguir trayectoria en línea recta.

Desventajas. La complejidad de los cálculos Cinemáticos, no puede realizar giros angostos.

Los cálculos cinemáticos para este tipo de configuración de ruedas se presentan en el apéndice A.

2.5.4. Síncrona

Como su nombre lo indica las ruedas se mueven de forma síncrona, es decir que todas las ruedas son capaces de tener tracción al mismo tiempo. La configuración síncrona es una derivación de la configuración diferencial, la cual está constituida por tres o más ruedas y quizás la diferencia más significativa con algún otro tipo de configuración es que todas las ruedas tienen tracción. Existen ciertas variaciones de este tipo de configuración dependiendo del tipo de ruedas utilizadas. Se debe considerar que este tipo de configuración requiere sincronización para realizar el desplazamiento del robot.

Figura 2.7: Algunos tipos de orientación y mecanismos de tracción de configuración síncrona.

Ventajas y desventajas de la configuración Síncrona

Ventajas. Control relativamente sencillo, puede ir en línea recta pero depende de la alineación mecánica de las ruedas, permite movimientos complejos, se puede cambiar la dirección de robot con tan solo cambiar la orientación de las ruedas.

Desventajas. La implementación mecánica es complicada así como su diseño.

Los cálculos cinemáticos para este tipo de configuración de ruedas se presentan en el apéndice A.

2.6. Localización de Robots Móviles

El tema de localización y navegación de los robots móviles tiene su origen en la ventaja competitiva que representaría para un robot poder contestarse a si mismo las siguientes preguntas:

¿Dónde estoy? ⇢ Localización
¿Hacia dónde voy? ⇢ Localización y navegación
¿Cómo debo llegar ahí? ⇢ Navegación

Esta idea fue presentada por Leonard y Durrant-Whyte [8], quienes también plantean un análisis de la relación que existe entre las preguntas y los problemas con los que se relaciona la respuesta; la presente tesis orienta su atención en brindar una solución viable al problema de localización de robots móviles terrestres, con lo cual se pretende dar respuesta a la pregunta ¿Dónde estoy?.

Cuando se habla del problema de localización de robots móviles es complicado establecer una clasificación general de las técnicas típicamente empleadas para la solución de este problema, algunas de las razones fundamentales que dificultan la realización de una clasificación que incluya a todas los técnicas son:

- Existe una gran variedad de perspectivas que pueden considerarse para la clasificación.
- Muchos técnicas están en constante perfeccionamiento.
- Algunas de estas técnicas están siendo fusionadas con otras para mejorar su eficiencia (técnicas híbridas).
- Siguen surgiendo nuevos algoritmos.

Borenstein [9] presenta una elegante clasificación de las técnicas de Seguimiento de Posición (Tracking) empleadas en la solución del problema de localización.

- **Medición con respecto a la Posición Relativa (Dead-Reckoning)**
  - Odometría.
  - Navegación Inercial.
- **Medición con respecto a la Posición Absoluta**
  - Balizas Activas.
    - Sistema de Posición Global (GPS).
    - Basados en tecnología RF y RFID.
  - Navegación con Landmarks.
  - Correspondencia entre Modelos (Model Matching).
  - Brújula Magnética.

La clasificación mostrada en la figura 2.8 fue acoplada en el presente trabajo de tesis con la finalidad de presentar un enfoque general de las técnicas que típicamente son referidas en la literatura sobre el tema, aunque cabe mencionar que muy probablemente surjan nuevas que no puedan ser acopladas a la presente clasificación.

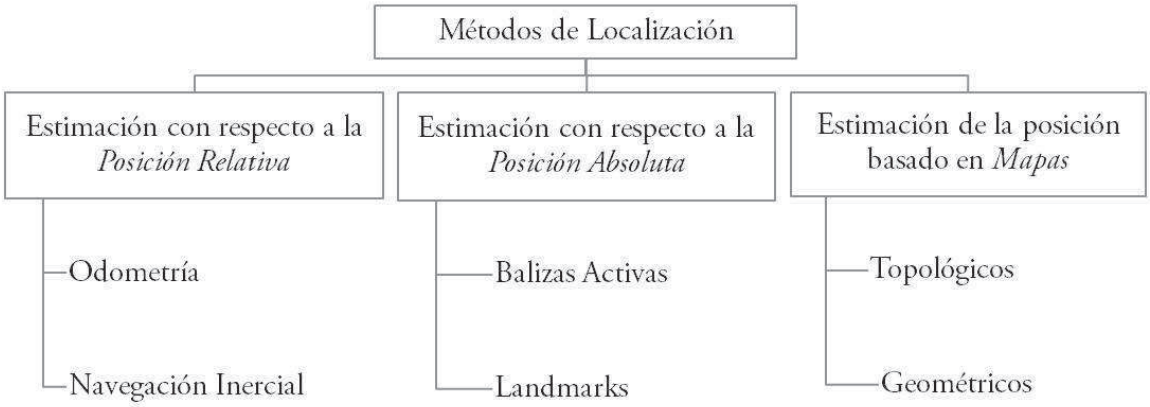


Figura 2.8: Clasificación de las técnicas comúnmente empleadas en la resolución del problema de localización de robots móviles.

Ahora bien, antes de describir las técnicas que son típicamente empleadas en la solución del problema de localización, es necesario mencionar que este problema está íntimamente ligado con los sistemas sensoriales que integran al robot, por lo que se ha considerado un apartado en la sección 2.7, el cual esta dedicado a la descripción de los sensores que comúnmente son integrados en los robots móviles, la importancia de los sensores recae en el hecho de que los sensores buscan extraer datos que permitan al robot conocer el medio que lo rodea y también sus condiciones internas, algunos de estos datos pueden ser empleados para alimentar cierto método de localización que permita al robot estimar su posición en un momento dado.

2.6.1. Medición de Posición Relativa

También llamados *Dead Reckoning*, esta técnica tiene el objetivo de estimar la posición de un robot móvil a partir de una posición inicial conocida. Estas técnicas presentan por lo general buenos resultados, con precisión aceptable en ambientes controlados.

Las técnicas que emplean la posición relativa, pueden ser divididas en *Sistemas Odométricos* y *Sistemas de Navegación Inercial*, tal como se puede ver en la figura 2.8.

2.6.1.1. Los Sistemas Odométricos

Son aquellos cuyo propósito es estimar la posición y orientación de un robot móvil a partir de la medición del recorrido del robot con respecto a un punto inicial determinado.

Estas técnicas hacen uso de cálculos odométricos, los cuales son simples procedimientos matemáticos que tienen el objetivo de estimar la posición de un robot en un instante dado a partir de un punto inicial conocido, en el caso particular de los robots móviles con ruedas, es común el uso de encoders para medir el número de pulsos que se obtienen cuando giran las ruedas. Los cálculos obtenidos por medio de odometría dependen del tipo de configuración de ruedas que sea seleccionado (diferencial, triciclo, ackerman, síncrona, entre otra), tal como se muestra en el apéndice A.

Algunas ventajas de este método son la simplicidad y el bajo costo, sin embargo las desventajas también deben ser consideradas, la primera es la acumulación de errores lo cual suele afectar gravemente la precisión del método, la calibración es otro aspecto que debe ser tomado en cuenta ya que el desajuste de ejes y de ruedas podría generar errores en donde la precisión se vería seriamente comprometida.

La odometría presenta desventajas importantes sin embargo, existen algoritmos que permiten minimizar el efecto de los errores, de manera que esta técnica puede ser capaz de presentar buenos resultados, por lo que es ampliamente empleada sobre todo en combinación con otras técnicas, ya que los cálculos odométricos son una opción auxiliar en caso de que los sensores presenten fallos, esta técnica hace uso de la información que proporciona el desplazamiento de las ruedas y no de los sensores.

Errores Frecuentes en las Mediciones Odométricos.
Los errores que deben considerarse cuando se usa la odometría como medio de estimación de la posición de un robot móvil con ruedas pueden clasificarse de forma general en: *errores sistemáticos* y *no sistemáticos*.
Los errores sistemáticos son particularmente muy graves, ya que almacenan errores acumulativos de forma constante, algunos errores sistemáticos frecuentes son:

- Que los diámetros de la ruedas sean diferentes entre sí.
- Que las especificaciones de fabricación de los diámetros de las ruedas sean diferentes a las reales, ya que los cálculos pueden estar considerando las especificaciones y no las mediciones reales de las ruedas.
- Desajuste del eje de las ruedas o bien que estén desalineadas.
- Resolución discreta (no continua) del encoder.
- La variación de la tasa de muestreo del encoder.

Los errores no sistemáticos se ocasionan de forma inesperada en el entorno del robot. En suelos desnivelados se puede dar el caso de que no todas las ruedas tengan contacto con la superficie, o bien que el robot haya sido levantado del suelo y puesto en otra posición a propósito, efecto llamado Kidnapping [10].

Otro error no sistemático muy común, es el patinaje de las ruedas debido a:

- Suelos resbaladizos.
- Sobre-aceleración.
- Derrapes (debidos a una rotación excesivamente rápida).
- Fuerzas externas (interacción con cuerpos externos).

2.6.1.2. Sistemas de Navegación Inercial

Es un sistema que permite estimar la posición de un robot móvil a partir de la medición de las aceleraciones y ángulos de orientación. Para ello suele hacerse uso de sensores como acelerómetros y giroscopios. Los sistemas de navegación inercial pueden ser utilizados en terrenos con irregularidades. El principal inconveniente de esta clase de sistemas es que los sensores suelen ser costosos.

Ya que este tipo de sistemas se basa en la extracción de los datos a partir de sensores inerciales, se puede obtener otra clase de datos útiles como el desplazamiento, la velocidad y la aceleración del robot, esto es gracias a el empleo del calculo diferencial e integral como se puede ver en la tabla 2.4.

Tabla 2.4: El desplazamiento, la velocidad y la aceleración, relacionadas con los cálculos diferencial e integral

	Forma Derivada	Forma Integral
Desplazamiento	$r(t)$	$r(t) = r_0 + \int_0^t v\, dt\,(t)$
Velocidad	$v(t) = \frac{dr}{dt}$	$r(t) = r_0 + \int_0^t v\, dt\,(t)$
Aceleración	$v(t) = \frac{dv}{dt} = \frac{d^2r}{dt^2}$	$r(t)$

Los acelerómetros comúnmente suelen hacer uso de sistemas pendulares y proporcionan una medición a partir de la aceleración del robot móvil. Para que el cálculo de la posición por medio de la integración sea preciso, es necesario considerar los errores de medición intrínsecos del sensor, ya que estos errores resultan críticos, incluso los pequeños errores cometidos por el sensor repercuten notablemente en la posición estimada. Para medir los ángulos de orientación del robot se utilizan sensores llamados giroscopios, hoy en día existen muchos tipos, si se desea una mayor descripción ver la sección 2.7.

Los giroscopios más comunes son los ópticos (de anillo ó de fibra óptica) y los mecánicos (masa giratoria)[11], aunque también se suelen utilizar las brújulas magnéticas para detectar cambios en la orientación del robot.

Los sistemas de navegación inerciales a diferencia de los sistemas odométricos, no se ven afectados por las irregularidades del terreno. Esto hace que en la práctica, sean mucho más fiables y precisos que los sistemas basados en odometría.

Existen algunas técnicas que suelen combinar la odometría y la navegación inercial llamada girodometría, la cual combina el uso de los encoders y giroscopios de modo que permite extraer lo mejor de cada una de las técnicas, ya que cuando el robot se desplaza en una superficie irregular, el giroscopio es capaz de detectarlo y se puede implementar un algoritmo que permita corregir los errores que se generen durante su desplazamiento, pero si el terreno es plano se puede hacer uso de la odometría sin problemas.

Las siguientes referencias presentan estudios muy completos acerca de la estimación inercial [12, 13, 14, 15, 16].

2.6.2. Medición de Posición Absoluta

La estimación de la posición de un robot móvil utilizando técnicas de posición absoluta no requiere conocer la posición inicial del robot. Estas técnicas hacen uso de información que proviene del exterior

del robot, una de las formas más habituales es utilizando dispositivos transmisores distribuidos estratégicamente en el área de navegación, mientras que en el robot se montan los receptores, razón por la cual esta clase de medición se encuentra estrechamente relacionada con las tecnologías de comunicación. Algunas técnicas enfocadas en estimar la posición del robot móvil sin conocer su posición inicial son:

- ≻ Balizas Activas.
- ≻ Navegación con Landmarks.
- ≻ Brújula Magnética (Magnetic Compass).

2.6.2.1. Balizas Activas

La técnica de balizas activas o en ingles *"Active Beacon"* [9] consiste básicamente en la estimación de la distancia que existe entre un receptor montado en el robot móvil y uno o más transmisores colocados anticipadamente en el entorno de navegación del robot, de los cuales se conoce su posición exacta. Para conocer la posición del robot se suelen utilizar técnicas como la Trilateración, Triangulación, Telemetría, entre otras.

Las balizas activas proporcionan una forma fiable y precisa de conocer la ubicación del robot móvil, su mayor inconveniente son los altos costos de instalación y de operación sobre todo en el caso del empleo de tecnología GPS (Sistema de Posicionamiento Global). Otro inconveniente es la necesidad de conocer previamente el entorno de navegación con la finalidad de colocar los transmisores en posiciones estratégicas, de forma que las señales de transmisión no tengan interrupciones sobre todo en tecnologías donde es necesario la visibilidad directa entre transmisor y receptor. Esta clase de sistemas son útiles en espacios relativamente pequeños y en espacios amplios siempre y cuando estos estén bien estructurados y no exista ruido que intervenga en la transmisión de señales.

Algunos trabajos relacionados con el uso de balizas activas para la localización de robots móviles son [17, 18, 19, 20, 21, 8].

Ahora bien se puede distinguir dos tipos de cálculos para la estimación de posición con balizas activas que son comúnmente referenciadas en la literatura concerniente al tema: *la Trilateración* y *la Triangulación.*

Debe tenerse en cuenta que estas no son las únicas formas para calcular la posición de un robot móvil por medio de balizas ya que también se encuentran la *Multilateración*, *TA (Timing Advance), TDOA (Time Difference Of Arrival)*, entre otras, las cuales no son descritas en el presente trabajo de tesis.

Trilateración (distancias) Es un método matemático que determina la posición relativa del robot móvil usado geometría de triángulos, para determinar las distancias entre los transmisores y receptores. Para realizar un cálculo preciso por medio de trilateración es necesario contar con por lo menos 3 puntos de referencia.

El procedimiento básico de la trilateración se basa en calcular la distancia entre el receptor montado en el robot móvil y por lo menos tres transmisores (balizas) instalados previamente en el entorno de navegación del robot móvil, de forma que si se traza una circunferencia con centro en cada transmisor y un radio igual a la distancia entre el transmisor y el robot móvil, tal como se muestra en la figura 2.9. Ahora bien el punto en el que se intersectan las tres circunferencias es la posición del robot móvil.

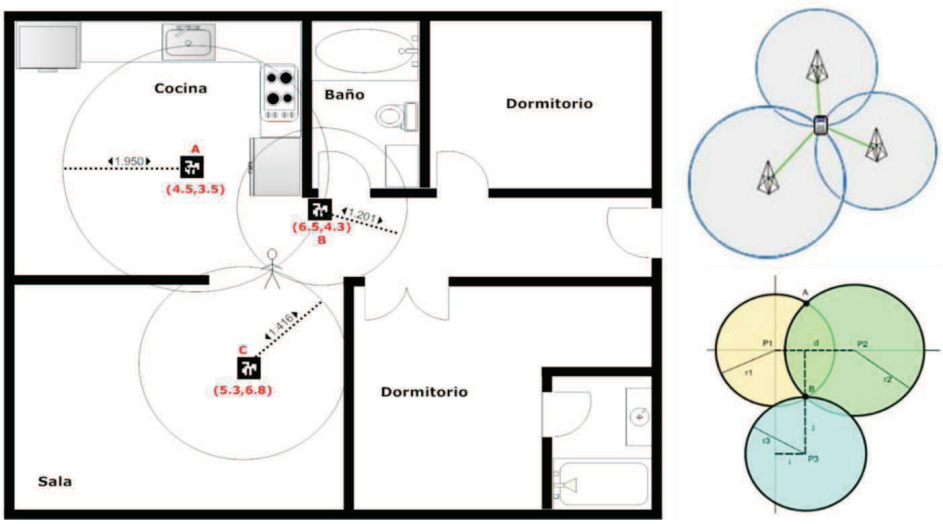


Figura 2.9: Estimación de la posición de un robot móvil por medio de Trilateración.

La Triangulación (ángulos + una o más distancias) La triangulación estima la posición del robot móvil por medio del uso de los ángulos formados por los transmisores instalados en el entorno y el receptor montado en el robot móvil, también hace uso de las distancias entre estos.

Al igual que en la Trilateración para obtener resultados confiables es necesario tener tres puntos de referencia conocidos.

- ≻ Sensores.
  - Infrarrojos.
  - Láser.
  - Ultrasónicos.
  - Entre otros.
- ≻ Tecnología de radiofrecuencia (RF).
  - Terrestres.
    - Radio de banda ultra ancha (UWB-Ultrawideband).
    - Marcadores de radiofrecuencia (RFID).
    - Entre otros.
  - Orbitales.
    - Sistemas Globales de Posicionamiento por Satélite (GNSS).

La figura 2.10 es extraída del artículo “Location-Aware Computing Comes of Age” elaborado por Hazas et al. [2], el cual muestra una clasificación de las técnicas típicamente utilizadas en la localización contra la precisión de cada tecnología con fines de localización, la extensión horizontal de cada rectángulo muestra el rango de precisión que cubre la tecnología en cuestión; La extensión vertical representa los rasgos significativos de la aplicación de la tecnología: el primero es el límite inferior el cual indica el desarrollo actual, mientras que el límite superior se muestra el despliegue previsto, en los próximos años.

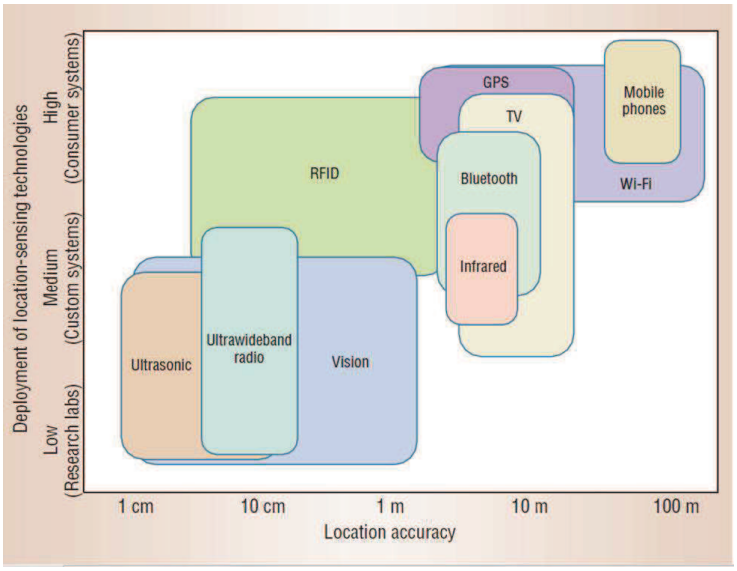


Figura 2.10: Cuadro comparativo de las tecnologías mayormente utilizadas en la localización[2]. Se compara el área de aplicación vs precisión de cada tecnología con fines de localización.

En el apéndice C se proporciona una descripción general de las estaciones de trabajo, mientras que los sensores son detallados en el apéndiceB.

2.6.2.2. Landmarks

Las landmarks son marcas o puntos de referencia que se destacan del entorno ya que poseen características distintivas, la finalidad de estos puntos es que sean reconocidos por el robot cuando este se encuentra navegando en el entorno. Las landmarks tienen posición fija y se conoce su localización dentro del entorno, este tipo de marcas deben tener características peculiares que sean distintivas del entorno, estas características son de utilidad cuando el robot debe reconocer una landmark de otro elemento del ambiente que no es una landmark. Es por ello que las landmarks deben ser cuidadosamente seleccionadas, de forma que sean de fácil identificación para el robot, además se deben considerar factores como alturas y ángulos de visibilidad, contraste de luz, entre otros.

El método puede ser visto en cuatro fases generales como se muestra en la figura 2.11:

Observación. El robot hace uso de los dispositivos para obtener información de su entorno.

Reconocimiento. Identifica cuales son los puntos de referencia y discrimina la información entre los datos de relevación de los que no lo son.

Correspondencia. Realiza tareas de correspondencia con información que el robot posee previamente, si la información no corresponde hay algoritmos que permiten determinar una nueva clase para esa información.

Localización. En esta fase el robot ha podido identificar la correspondencia entre los datos obtenidos del sensor(es) y su banco de datos, de forma que ya puede presentar un resultado de su ubicación.

Figura 2.11: Fases generales de localización de robots basada en Landmarks.

La principal tarea del robot cuando se hace uso de landmarks reside en la identificación de estos puntos de referencia, por lo que el sistema sensorial debe ser los más preciso posible, de forma que es muy común encontrar este tipo de método relacionado con la visión artificial, ya que esta representa una opción de sensado robusta. Otro aspecto de relevancia es la información que se encuentra en el banco de datos previamente guardado en la memoria del robot, ya que una vez identificados los puntos de referencia es necesario hacer una correspondencia con la información que se le ha almacenado al robot, para determinar a que posición corresponde la landmark. Por lo que ambos datos de referencia deben ser lo más certeros posibles.

Las landmarks son consideradas dentro del rubro de medición de posición absoluta como se muestra en la clasificación previamente ilustrada en la figura 2.8, ya que el robot puede estimar su localización sin tomar como referencia su posición inicial. Ya que él solo necesita identificar la landmark más cercana y cotejar la información con el banco de datos previamente almacenado y entonces obtendrá su posición.

Existen básicamente dos tipos diferentes de landmarks: *Landmark Naturales y Landmark Artificiales*, las cuales serán descritas de forma general a continuación.

Landmarks Naturales

Son las más difíciles de identificar ya que son marcas que de forma intrínseca y natural forman parte del entorno. Por ejemplo para *ambientes interiores* pueden considerarse las esquinas, focos, apagadores, ventanas y puertas, mientras que para *ambientes exteriores* se tienen los semáforos, árboles y postes de luz. Dicho en otras palabras las landmark naturales son marcas distintivas que pertenecen de forma natural al entorno de navegación del robot móvil y de las que de antemano se sabe que no cambian su posición, esto último es muy importante ya que sí el árbol es talado, entonces el robot será incapaz de conocer su posición.

Ahora bien, como ya se mencionó las landmark naturales presentan una alta dificultad para ser identificadas dentro del entorno, por lo cual los algoritmos muchas veces requieren una alta tasa de muestreo para poder determinar inequívocamente su posición.

Landmarks Artificiales
Este tipo de marcas se colocan de forma predeterminada en el entorno de navegación del robot, las landmark artificiales poseen características significativamente distintivas, ya que su propósito es que el robot pueda identificarlas de forma inequívoca del entorno y no puedan ser confundidas con algún otro elemento del ambiente. Este tipo de Landmarks suele clasificarse en *activas* y *pasivas*.

Activas. Son aquellas que emiten algún tipo de señal y de esta manera informan sobre su localización dentro del ambiente de navegación.

Pasivas. Son aquellas que no emiten activamente ningún tipo de señal, por lo que el robot debe buscarlas activamente mediante sus sensores.

Algunos trabajos relacionados con el uso del método de landmarks o hitos para la localización de robots móviles son [22, 23, 24, 25, 26, 27].

2.6.3. Empleando Mapas

Este método de localización tiene como objetivo la creación de mapas del entorno de navegación del robot con la finalidad de conocerlo y ser capaz de estimar su posición a partir de éste.

Un modelo o mapa del entorno es una abstracción creada a partir de aquellas características que definen el ambiente de navegación, ahora bien las características que definen este ambiente pueden ser muchas y formar una lista muy grande, por lo que debe considerarse la selección de aquellas que puedan ser útiles en la navegación y/o la localización del robot, dicha selección ayudara a considerar cuáles son los sensores requeridos para proporcionar al robot los datos necesarios y que sea capaz de construir un mapa.

De forma general los algoritmos empleados para este tipo de localización suelen basarse en la «correspondencia de modelos», la cual básicamente consiste en obtener las lecturas provenientes de los sensores con el fin de que el robot sea capaz de crear un mapa y establecer una correspondencia con un mapa global previamente almacenado. Borenstein [9], menciona que dicho método consta de cuatro etapas:

1. Obtener información del mundo real lo más fidedigna posible a través de los sensores.
2. Construir un mapa local a partir de los datos previamente obtenidos.
3. Es la fase de correspondencia ya que se compara el mapa local (generado en el paso 2) con el mapa global que con anticipación se ha almacenado en la memoria del robot.
4. En la última fase se calcula la posición actual del robot a partir del resultado obtenido en el paso 3.

Para obtener buenos resultados con esta técnica es vital obtener información lo más certera posible del entorno de navegación, por lo que la clave es la calidad en la extracción de los datos durante la fase de percepción y también la capacidad del sistema para construir una buena representación del entorno, otro criterio fundamental es el tipo de algoritmo seleccionado para la correspondencia entre los mapas.

En torno a los tipos de mapas han surgido dos paradigmas principales con respecto al modelado de los entornos de oficina: modelos métricos y modelos topológicos [28].

2.6.3.1. Localización con mapas métricos

Este enfoque se basa en las representaciones métricas identificables de elementos que se encuentran en el entorno, **un factor fundamental** en este enfoque es la *precisión* con la que deben ser percibidos y después modelados los elementos y espacios dentro del entorno, en otras palabras el resultado de la localización de mapas métricos depende directamente de la información proporcionada por los sensores que se encargan de determinar la posición de cada elemento, así como del algoritmo que permite su correcta interpretación logrando así el modelado del mapa.

Los elementos y espacios suelen ser representados por medio de figuras geométricas (puntos, círculos, líneas, polígonos, entre otras) los cuales simbolizan un área acotada dentro del entorno de navegación del robot.

Algunos trabajos relacionados con el uso de los mapas métricos para la localización de robots móviles son [29, 30, 31, 32].

Ahora bien, para la descripción de esta clase de modelos en el presente trabajo de tesis se han dividido los mapas métricos en dos tipos de mapas: *basados en rejillas* y *geométricos.*

Mapas basados en rejillas
Propuestos por Moravec y Elfes [33], la idea básica es una matriz MxN de celdas que representa el área de navegación del robot móvil, en donde todas las celdas son de igual dimensión. Cada celda almacena la probabilidad de ser ocupada o vacía, de modo que es la posibilidad de que en dicha celda exista o no un obstáculo. El rango de probabilidades es de 0 a 1, en donde 1 es la probabilidad máxima de que la celda este ocupada y cero señala la máxima probabilidad de que este vacía.

Es posible utilizar rejillas de ocupación definidas por el usuario, pero lo usual es que sea el propio robot móvil quien recorra de forma iterativa todo el entorno de navegación y que realice la construcción de la rejilla de forma autónoma, mediante algún algoritmo de exploración [34, 35, 31].

Modelos geométricos
Los mapas geométricos son básicamente una representación de elementos relevantes, así como de las propiedades métricas representativas dentro del entorno de navegación, Las representaciones métricas suelen ser el área, distancia, tamaño, posición, orientación, entre otras. Estos datos son esquematizados con primitivas geométricas tales como puntos, líneas, arcos y círculos, los cuales pueden ser dimensionados en función de su longitud, diámetro, color, entre otros. [8].

Este tipo de mapas pueden ser útiles en la representación de entornos de navegación relativamente grandes, esto debido a que su interpretación suele ser sencilla. Una de las dificultades del modelado de este tipo de mapas es el rango de error de los sensores (ultrasónico, infrarrojo, laser, entre otros), ya que como se mencionó en general los mapas métricos dependen de la precisión, por lo que son sensibles a la inestabilidad de las lecturas de los sensores.

Otra dificultad es la integración de las lecturas locales, para lograr el modelado de un mapa global que represente fidedignamente el entorno de navegación, por ejemplo supóngase un ambiente de navegación

determinado el cual esta constituido por cuatro habitaciones, entonces el robot deberá realizar lecturas locales de cada habitación, sin embargo para modelar un mapa general de las cuatro habitaciones se deben reunir las cuatro lecturas locales, lo cual suele convertirse en un problema en el momento de emparejar o hacer coincidir las mediciones respectivas.

2.6.3.2. Localización con mapas topológicos

El concepto de mapa topológico fue descrito por Benjamin Kuipers [36]. La idea central de los mapas topológicos es realizar una descripción a partir de la percepción del entorno de navegación del robot, la representación se realiza por medio de un conjunto de nodos no dirigidos, en donde cada nodo simboliza una colección de lugares del entorno, los cuales tienen relevancia para la localización del robot.

En los experimentos realizados por Kuipers, se menciona que la función de distinción calcula el número de objetos cercanos que se encuentran a igual distancia del robot. Mientras que los arcos entre los nodos representan caminos que se han seguido para llegar de un nodo a otro utilizando una determinada estrategia de control local.

Los mapas topológicos presentan ventajas potenciales, e.g. no requieren demasiada precisión en el modelado, lo cual se ve reflejado en un bajo costo computacional, además de que hacen uso frecuente de las teorías probabilísticas para describir las inexactitudes de los sensores.

Algunos trabajos relacionados con el uso de los mapas topológicos para la localización de robots móviles son [37, 38, 39].

2.7. Percepción

Sensor. Son dispositivos cuya función principal es obtener información. Gracias a los sensores los robots son capaces de recibir información de las condiciones internas y externas a él. Los sensores son similares en muchas formas a los cinco sentidos de un ser humano, los cuales permiten recibir información sobre el medio ambiente.

Los sensores son esenciales en la autonomía de un robot, ya que estos brindan la posibilidad de que el robot explore y conozca su entorno. Si se considera todo lo que el ser humano ha logrado a partir de explorar y comprender lo que le rodea, entonces se puede deducir cuán importante es para el robot móvil sentir y percibir el exterior, por supuesto los sensores son solo una parte de la autonomía, que aunque imprescindible no sirven de nada si el robot no sabe qué hacer con todos esos datos proporcionados por los sensores.

En el siguiente apartado se presenta una descripción general de los sensores usualmente aplicados en la localización de los robots móviles. Como debe comprenderse la explicación a detalle de cada uno de estos sensores seria extensa, por lo que en el presente apartado solo se describen características básicas así como los artículos de investigaciones que actualmente presentan resultados con la utilización de cada sensor en robots móviles. Si se desea conocer más acerca de los sensores empleados en la robótica móvil se recomienda consultar la siguiente bibliografía [9, 40].

En la actualidad existe un gran número de sensores que son vendidos a nivel comercial, esto se debe a que la tecnología en áreas como nano-materiales y telecomunicaciones avanza a pasos agigantados, lo

que permite adquirir sensores de menor tamaño, mejor precisión, a precios accesibles (muchos de ellos) y con menor consumo energético. Lo que significa que se puede dotar a un robot de múltiples sistemas sensoriales, que permitan obtener mayor y mejor información del entorno.

Algunas características técnicas básicas que deben considerarse cuando se selecciona un sensor son:

Accesibilidad. Son todos aquellos aspectos o características propias del entorno de navegación del robot, que pueden afectar las mediciones tomadas por el sensor en cuestión.

Dimensión. Son las unidades de medición del sensor y su representación en el espacio, de modo que la representación corresponde a: un plano (medición escalar), dos planos (vectorial), tres o más planos (mediciones n-dimensional).

Rango de operación. Es el conjunto de valores comprendidos entre los limites (Superior e Inferior) que es capaz de medir el sensor.

Formatos de los datos. Es la estructura de los datos entregados por el sensor. Es de gran importancia conocer la forma de los datos ya que de ello depende su adecuada interpretación.

Exactitud. Paradójicamente la exactitud es el grado de inexactitud del sensor, es decir el rango de proximidad de la medición tomada por el sensor con respecto al valor real de la medición. En ocasiones la exactitud puede ser expresada como una variación de error máximo y mínimo (+/-), aunque también puede verse representado como un porcentaje de la salida a rango total.

Error. Se define como la diferencia entre el valor medido y el valor verdadero. Los errores pueden ser de dos tipos: Los determinísticos o sistemáticos que pueden preverse, calcularse o incluso eliminarse mediante calibraciones y compensaciones y los errores aleatorios que no pueden ser previstos, pues dependen de causas desconocidas o estocásticas.

Rapidez de respuesta. Se refiere al tiempo mínimo entre muestra y muestra, por lo que esta estrechamente relacionado con la tasa de muestreo del sensor.

2.7.1. Clasificación de los sensores empleados en la localización

Hoy en día los sensores son clasificados en la literatura de diferentes formas, sin embargo hay dos de ellas que prestan puntual atención en la robótica móvil. La primera se muestra en la figura 2.12, en donde el enfoque de clasificación realiza una sinergia entre las técnicas de localización y los sensores típicamente empleados en cada una.

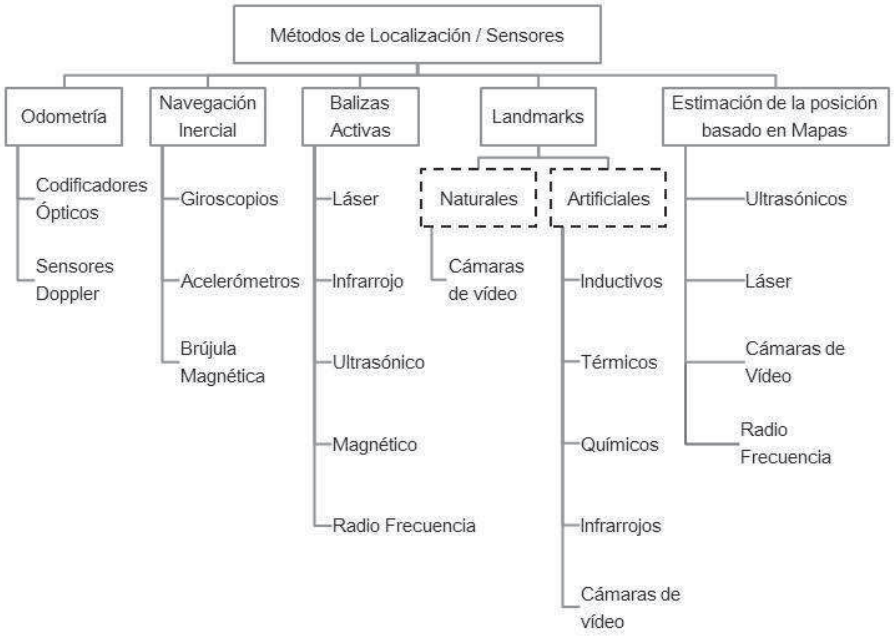


Figura 2.12: Clasificación de las técnicas de localización en relación con los sensores y tecnologías comúnmente empleadas.

Otra clasificación usualmente referida en la robótica móvil es la que considera la procedencia de la información obtenida por los sensores, es decir si los datos proporcionan información de las condiciones internas del robot o bien si pertenecen al entorno de navegación, estos dos tipos de sensores son denominados propioceptivos y exteroceptivos respectivamente [41].

Propioceptivos. Obtienen información referente al estado interno del robot. e.g. nivel de carga de la batería, orientación, velocidad, aceleración, temperatura, entre otros.

Exteroceptivos. Son sensores que adquieren información del exterior del robot, es decir extrae datos del entorno. Por ejemplo distancias, intensidades de luz, amplitudes de sonido, presión y temperatura ambiental, entre otros.

La descripción detallada de los sensores mayormente empleados en la localización de robots móviles se encuentra definida en el apéndice B.

Capítulo 3

Métodos y Herramientas

El presente capítulo se encuentra destinado a la descripción de herramientas y métodos empleadas a lo largo de la fase experimental, de tal manera el Capítulo III se ha dividido para su mejor organización en tres secciones principales. La primera sección tiene el propósito de describir a detalle el funcionamiento y modelo matemático de las memorias asociativas basadas en modelos Alfa-Beta, las cuales son empleadas en la localización del robot móvil, ya que el uso de esta clase de memorias en la robótica móvil son la motivación principal del presente trabajo de investigación.

Por otro lado en una segunda sección del capítulo se proporciona una descripción general de Microsoft Robotics Development Studio (MRDS), herramienta utilizada en la programación de diversos robots, y la cual es de gran importancia en el presente trabajo ya que será empleada en la simulación de un robot móvil al cual se le ha incorporado las memorias asociativas Alfa-Beta, con la finalidad de determinar su efectividad al ser empleadas en el problema de localización. MRDS fue seleccionada debido a sus características y versatilidad de programación, además de las bondades que ofrece para la migración entre la simulación de un entorno virtual y la implementación en un robot móvil real.

Finalmente la tercera sección tiene el propósito de describir de forma muy general y breve el robot móvil y los sensores que han sido seleccionados para la implementación del algoritmo de las memorias Alfa-Beta; para lo cual se consideraron una serie de características tales como la configuración de las ruedas (diferencial, triciclo, ackerman, síncrono), precisión de los sensores y compatibilidad con respecto al robot seleccionado, y finalmente se debe considerar que el robot móvil seleccionado se encuentre dentro de la gama de robots que son soportados por la herramienta de simulación de la plataforma de MRDS.

3.1. Memorias Asociativas Alfa-Beta

Esta primera sección del capítulo III se concentra en las memorias asociativas Alfa-Beta, estas son de particular interés en el presente trabajo de tesis debido a sus alto grado de eficiencia en las tareas de reconocimiento de patrones, sin embargo no es el único método que ésta dedicado en la realización de dicha tarea, es por ello que se hace necesario ubicar el papel que tienen las memorias asociativas con respecto a otros métodos, en el contexto de la investigación referenciada a la Inteligencia Artificial (IA), por lo que a su vez la presente sección está dividida en 3 partes; en una primera parte se denota el rol de las memorias asociativas en el contexto general de la literatura de la Inteligencia Artificial (IA), la cual da pie a la segunda parte que pretende describir los conceptos básicos y propiedades de los

operadores α y β, que son la base medular en el diseño e implementación de las memorias asociativas, también durante el desarrollo de esta sección se presentan las operaciones matriciales que se derivan de estas operaciones originales, para finalmente en una tercera parte sea descrita de forma puntual el modelo matemático de las Memorias Asociativas Alfa-Beta, incluyendo las fases de aprendizaje y recuperación de las memorias Alfa-Beta, empleando los operadores $\bigvee$ (max) y $\bigvee$ (min).

3.1.1. El enfoque de las Memorias Asociativas y la Inteligencia Artificial (IA)

Existen un conjunto de enfoques que son empleados en la solución de los problemas relacionados con la clasificación de patrones, a continuación se describen los enfoques más comunes:

Enfoque estadístico—probabilístico. Es de los más empleados ya que son pioneros en el área, los principios fundamentales de esta clase de métodos se encuentran en las teorías de probabilidad y estadística, y de forma puntual en el teorema de Bayes, Markov, entre otros. [42, 43, 44, 45].

Clasificadores basados en métricas. Este enfoque hace uso de las mediciones espaciales que existen entre los patrones, por lo que hace uso de principios métricos y espaciales, este tipo de métodos se caracterizan por su sencillez y por su alto porcentaje de asertividad, quizás uno de los métodos más conocidos de este enfoque es el vecino más cercano (K-Nearest Neighbor KNN en inglés) [46].

Enfoque sintáctico—estructural. Este tipo de enfoque tiene sus bases en la teoría de autómatas así como en lenguajes formales, fundamentalmente establecen si una determinada estructura pertenece a cierto conjunto de reglas que son previamente establecidas llamadas gramáticas [47, 48, 49, 50].

Enfoque neuronal. Son métodos inspirados en los modelos biológicos del comportamiento de una red neuronal, los modelos matemáticos empleados hacen posible que este tipo de métodos sean capaces de clasificar patrones con gran asertividad [51].

Enfoque asociativo. Este enfoque fue creado en México en el año 2002 dentro del Instituto Politécnico Nacional IPN, en uno de los centros de investigación llamado "Centro de Investigación en Computación CIC". En esencia hace uso de los modelos de memorias asociativas para dar nacimiento a reconocedores robustos en la clasificación de patrones con ruido, dichos modelos son capaces de recuperar los patrones aprendidos de una forma óptima y además de clasificar patrones nuevos con alto grado de asertividad [52, 53, 54, 55, 56, 57, 58].

Ahora bien, conociendo de forma general los enfoques que mayormente predominan en la literatura, el presente trabajo de investigación concentrará su estudio en el último enfoque presentado en la sección 3.1.1 , el ***enfoque asociativo***.

3.2. Operaciones binarias α y β: definiciones y propiedades

En esencia una memoria asociativa tiene el objetivo de recuperar de forma correcta patrones completos, a partir del previo aprendizaje con patrones de entrada, los cuales pueden estar alterados por un ruido de tipo aditivo, sustractivo o combinado, lo cual brinda un mundo de posibilidades para la utilización de las memorias asociativas en la solución de problemas que presenten dichas características. Por lo que un considerable grupo de investigadores se han dado a la tarea de generar modelos de memorias

asociativas enfocándose en mejorar la asertividad en los resultados, así como, también en la eficiencia de respuesta o recuperación de patrones, rapidez y el grado de inmunidad al ruido.

Las memorias asociativas pueden ser divididas en dos fases:

Fase de aprendizaje. Esta fase esta enfocada en la construcción de la memoria asociativa, por lo que el modelo aprende a partir de los patrones de entrada. La idea fundamental se encuentra enfocada en encontrar el o los operadores necesarios que sean capaces de codificar la relación que existe entre los patrones de entrada y de salida, y con dicha codificación se genere la matriz de aprendizaje M.

Fase de recuperación. En la fase de recuperación la memoria pone a prueba lo que ha aprendido previamente. Entonces una vez que se ha conformado la matriz M (en la fase de aprendizaje), se le presenta un patrón de entrada x que previamente se aprendió, M se opera con el operador o los operadores necesarios bajo ciertas circunstancias con el patrón x, y se genera un patrón de salida y.

Fase de aprendizaje Fase de recuperación

Figura 3.1: El patrón de entrada representado por un vector columna x y el patrón de salida representado por y.

Ya que el interés del presente trabajo de tesis se encuentra enfocado en las memorias asociativas Alfa-Beta, es necesario describir los conceptos básicos y los modelos matemáticos de los operadores α y β, para ello se hará uso de la bibliografía correspondiente a las investigaciones realizadas por el grupo Alfa-Beta [58, 56, 57, 45].

Las memorias asociativas Alfa-Beta hacen uso de dos operadores binarios α y β, además hacen uso de máximos $\vee$ y mínimos $\wedge$; este tipo de memorias son capaces de trabajar bajo dos diferentes arquitecturas (autoasociativo y heretoasociativo).

El operador α es utilizado para la fase de aprendizaje y el operador β en la fase de recuperación. La utilización de ambos operadores se rigen bajo las directrices que pueden verse en las tablas 3.1 y 3.2 y su correspondiente demostración matemática se puede encontrar en [58].

Para la definición de los operadores binarios α y β se especifican los conjuntos denominados A y B los cuales contienen los elementos:

$$A = 0, 1 \;\; y \;\; B = 0, 1, 2$$

Entonces la operación binaria $\alpha = A\, x\, A \rightarrow B$ está establecida por la tabla 3.1:

Tabla 3.1: Operación binaria $\alpha = A\ x\ A \rightarrow B$

x	y	$\alpha(x, y)$
0	0	1
0	1	0
1	0	2
1	1	1

Mientras que la operación binaria $\beta = B\ x\ A \rightarrow A$ se define como se muestra en la tabla 3.2:

Tabla 3.2: Operación binaria $\beta = BxA \rightarrow A$

x	y	$\beta(x, y)$
0	0	0
0	1	0
1	0	0
1	1	1
2	0	1
2	1	1

3.2.1. Propiedades algebraicas de los operadores binarios Alfa-Beta

El operador binario Alfa (α) se rige bajo las propiedades algebraicas mostradas en la tabla 3.3, debe recordase que $\vee$ es el operador máximo y $\wedge$ es el operador de mínimo.

Tabla 3.3: Propiedades algebraicas de operador binario α

α1 isoargumentos en α	$\alpha(x, x) = 1$
α2.- intercambio de argumentos en α	$(x \leq y) \longleftrightarrow \alpha(x, y) \leq \alpha(y, x)$
α3.-es creciente por la izquierda	$(x \leq y) \longleftrightarrow [\alpha(x, z) \leq \alpha(y, z)]$
α4.-es decreciente por la derecha	$(x \leq y) \longleftrightarrow [\alpha(z, x) \leq \alpha(z, y)]$
α5.-es distributiva por la derecha respecto al $\vee$	$[\alpha(x \vee y), z] = \alpha(x, z) \vee \alpha(y, z)$
α6.-es distributiva por la derecha respecto al $\wedge$	$[\alpha(x \wedge y), z] = \alpha(x, z) \wedge \alpha(y, z)$

El operador binario α se rige bajo las propiedades algebraicas mostradas en la tabla 3.4, debe recordase que $\vee$ es el operador máximo y $\wedge$ es el operador de mínimo.

Tabla 3.4: Propiedades algebraicas de operador binario β

$\beta 1$ isoargumentos en β	$\beta(1,x) = x$
$\beta 2$.- intercambio de argumentos en β	$\beta(x,x) = x \; \forall x \epsilon A$
$\beta 3$.-es creciente por la izquierda	$(x \leq y) \longrightarrow [\beta(x,z) \leq \beta(y,z)]$
$\beta 4$.-es decreciente por la derecha	$(x \leq y) \longleftrightarrow [\beta(z,x) \leq \beta(z,y)]$
$\beta 5$.-es distributiva por la derecha respecto al $\vee$	$[\beta(x \vee y), z] = \beta(x,z) \vee \beta(y,z)$
$\beta 6$.-es distributiva por la derecha respecto al $\wedge$	$[\beta(x \wedge y), z] = \beta(x,z) \wedge \beta(y,z)$

3.2.2. Operaciones matriciales

Los conjuntos A y B, las operaciones binarias α y β junto con los operadores $\wedge$ (mínimo) y $\vee$ (máximo) conforman el sistema algebraico (A, B, α, β , $\wedge$,$\vee$) en el que están inmersas las memorias asociativas Alfa-Beta.

Para comenzar se requiere la definición de cuatro operaciones matriciales que se muestran en la tabla 3.5.

Tabla 3.5: Operaciones Matriciales

Operación $\alpha max : P_{m\,x\,r} \triangle_a Q_{r\,x\,n} = \mid f_{ij}^{\alpha} \mid$, *donde* $f_{ij}^{\alpha} = \bigvee_{k=1}^{r} \alpha\,(p_{ik}, q_{kj})$
Operación $\beta max : P_{m\,x\,r} \triangle_\beta Q_{r\,x\,n} = \mid f_{ij}^{\beta} \mid$, *donde* $f_{ij}^{\beta} = \bigvee_{k=1}^{r} \beta\,(p_{ik}, q_{kj})$
Operación $\alpha min : P_{m\,x\,r} \nabla_a Q_{r\,x\,n} = \mid h_{ij}^{\alpha} \mid$, *donde* $h_{ij}^{\alpha} = \bigwedge_{k=1}^{r} \alpha\,(p_{ik}, q_{kj})$
Operación $\beta min : P_{m\,x\,r} \nabla_\beta Q_{r\,x\,n} = \mid h_{ij}^{\beta} \mid$, *donde* $h_{ij}^{\beta} = \bigwedge_{k=1}^{r} \beta\,(p_{ik}, q_{kj})$

Los lemas presentados a continuación permiten comprender de mejor manera a las memorias asociativas, ya que presentan la forma de utilizar los operadores para la construcción de la matriz de aprendizaje a partir de patrones de entrada [58].

El siguiente lema muestra los resultados obtenidos al utilizar las operaciones correspondientes a la utilización del operador α, las aplicadas al vector columna y vector fila que corresponden a los dos primeros patrones del banco de aprendizaje.

Lema 1. Sean $x \epsilon A^n$ y $y \epsilon A^m$; entonces $y \nabla_\alpha x^t$ es una matriz de dimensiones $m\,x\,n$ y que además se cumple que: $y \nabla_\alpha x^t = y \triangle_\alpha x^t$.

Demostración

$$y\nabla_\alpha x^t = \begin{pmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{pmatrix} \nabla_\alpha(x_1,\ x_2,\ ...\ x_n) \tag{3.1}$$

$$y\nabla_\alpha x^t = \begin{pmatrix} \nabla^1_{k=1}\alpha(y_1,x_1) & \nabla^1_{k=1}\alpha(y_1,x_2) & . & . & . & \nabla^1_{k=1}\alpha(y_1,x_n) \\ \nabla^1_{k=1}\alpha(y_2,x_1) & \nabla^1_{k=1}\alpha(y_2,x_2) & . & . & . & \nabla^1_{k=1}\alpha(y_2,x_n) \\ . & . & & & & . \\ . & . & . & . & . & . \\ . & . & & & & . \\ \nabla^1_{k=1}\alpha(y_m,x_1) & \nabla^1_{k=1}\alpha(y_{m1},x_2) & . & . & . & \nabla^1_{k=1}\alpha(y_m,x_n) \end{pmatrix}_{mxn} \tag{3.2}$$

$$y\nabla_\alpha x^t = \begin{pmatrix} \alpha(y_1,x_1) & \alpha(y_1,x_2) & . & . & . & \alpha(y_1,x_n) \\ \alpha(y_2,x_1) & \alpha(y_2,x_2) & . & . & . & \alpha(y_2,x_n) \\ . & . & & & & . \\ . & . & . & . & . & . \\ . & . & & & & . \\ \alpha(y_m,x_1) & \alpha(y_{m1},x_2) & . & . & . & \alpha(y_m,x_n) \end{pmatrix}_{mxn} \tag{3.3}$$

$$y\nabla_\alpha x^t = \begin{pmatrix} \bigwedge^1_{k=1}\alpha(y_1,x_1) & \bigwedge^1_{k=1}\alpha(y_1,x_2) & . & . & . & \bigwedge^1_{k=1}\alpha(y_1,x_n) \\ \bigwedge^1_{k=1}\alpha(y_2,x_1) & \bigwedge^1_{k=1}\alpha(y_2,x_2) & . & . & . & \bigwedge^1_{k=1}\alpha(y_2,x_n) \\ . & . & & & & . \\ . & . & . & . & . & . \\ . & . & & & & . \\ \bigwedge^1_{k=1}\alpha(y_m,x_1) & \bigwedge^1_{k=1}\alpha(y_{m1},x_2) & . & . & . & \bigwedge^1_{k=1}\alpha(y_m,x_n) \end{pmatrix}_{mxn} \tag{3.4}$$

Entonces se puede decir que la matriz $y\ \nabla_\alpha x^t$ es una matriz de dimensiones mxn, y que $y\nabla_\alpha x^t = y\triangle_\alpha x^t$.

Para representar las operaciones ∇_α y $\triangle_\alpha$, se ha designado el siguiente símbolo $\otimes$, el cual se opera un vector columna de dimensión m con un vector fila de dimensión n:

$$y\nabla_\alpha x^t = y\ \otimes\ x^t = y\triangle_\alpha x^t \tag{3.5}$$

Donde ij-ésima componente de la matriz $y\ \otimes\ x^t$está dada por:

$$\left[y\ \otimes\ x^t\right]_{ij} = \alpha(y_i,\ x_j) \tag{3.6}$$

Dado un índice de asociación μ, la expresión anterior indica ij-ésima componente de la matriz $y^\mu \otimes (x^\mu)^t$, es posible expresarla de la siguiente manera:

$$\left[y^\mu \otimes (x^\mu)^t\right]_{ij} = \alpha(y^\mu_i,\ x^\mu_j) \tag{3.7}$$

Hasta este punto se ha analizado al operador α, ahora toca el turno al análisis del operador β, el cual opera con una matriz de dimensiones mxn con un vector columna de dimensión n usando las operaciones ∇_β y $\triangle_\beta$. A continuación se presentan los lemas 2 y 3, los cuales están relacionados con las i-ésimas componentes de los vectores columna resultantes de dimensión m, a partir de ambas operaciones ∇_β y $\triangle_\beta$.

Lema 2. Sean $x \epsilon A^n$ y P una matriz de dimensiones $m\,x\,n$. La operación $P_{mxn} \nabla_\beta x$ da como resultado un vector columna de dimensión m, cuya i-ésima componente tiene la siguiente forma y que además se cumple que: $\left(P_{mxn} \nabla_\beta x\right)_i = \bigvee_{j=1}^{n} \beta\left(p_{ij},\, x_j\right)$.

Demostración.

$$P_{mxn} \nabla_\beta x = \begin{pmatrix} p_{11} & p_{12} & . & . & . & p_{1n} \\ p_{21} & p_{22} & . & . & . & p_{2n} \\ . & . & & & & . \\ . & . & . & . & . & . \\ . & . & & & & . \\ p_{m1} & p_{m2} & . & . & . & p_{mn} \end{pmatrix} \nabla_\beta \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ \\ x_n \end{pmatrix} \tag{3.8}$$

$$P_{mxn} \nabla_\beta x = \begin{pmatrix} \beta(p_{11}, x_1) & \bigvee & \beta(p_{12}, x_2) & \bigvee & . & . & . & \beta(p_{1n}, x_n) \\ \beta(p_{21}, x_1) & \bigvee & \beta(p_{22}, x_2) & \bigvee & . & . & . & \beta(p_{2n}, x_n) \\ . & & . & & & & & . \\ . & & . & & . & . & . & . \\ . & & . & & & & & . \\ \beta(p_m, x_1) & \bigvee & \beta(p_{m1}, x_2) & \bigvee & . & . & . & \beta(p_{mn}, x_n) \end{pmatrix} = \begin{pmatrix} \bigvee_{j=1}^{n} \beta(p_{1j}, x_j) \\ \bigvee_{j=1}^{n} \beta(p_{2j}, x_j) \\ . \\ . \\ . \\ \\ \bigvee_{j=1}^{n} \beta(p_{mj}, x_j) \end{pmatrix} \tag{3.9}$$

Se obtiene un vector columna de dimensión m cuya i-ésima componente es

Lema 3. Sean $x \epsilon A^n$ y P una matriz de dimensiones mxn. La operación $P_{mxn} \triangle_\beta x$ da como resultado un vector columna de dimensión m, cuya i-ésima componente tiene la siguiente forma y que además se cumple que: $\left(P_{mxn} \triangle_\beta x\right)_i = \bigwedge_{j=1}^{n} \beta\left(p_{ij},\, x_j\right)$.

Demostración.

$$P_{mxn} \triangle_\beta x = \begin{pmatrix} p_{11} & p_{12} & . & . & . & p_{1n} \\ p_{21} & p_{22} & . & . & . & p_{2n} \\ . & . & & & & . \\ . & . & . & . & . & . \\ . & . & & & & . \\ p_{m1} & p_{m2} & . & . & . & p_{mn} \end{pmatrix} \triangle_\beta \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ \\ x_n \end{pmatrix} \tag{3.10}$$

$$P_{mxn} \nabla_\beta x = \begin{pmatrix} \beta(p_{11}, x_1) & \bigwedge & \beta(p_{12}, x_2) & \bigwedge & . & . & . & \beta(p_{1n}, x_n) \\ \beta(p_{21}, x_1) & \bigwedge & \beta(p_{22}, x_2) & \bigwedge & . & . & . & \beta(p_{2n}, x_n) \\ . & & . & & & & & . \\ . & & . & & . & . & . & . \\ . & & . & & & & & . \\ \beta(p_m, x_1) & \bigwedge & \beta(p_{m1}, x_2) & \bigwedge & . & . & . & \beta(p_{mn}, x_n) \end{pmatrix} = \begin{pmatrix} \bigwedge_{j=1}^{n} \beta(p_{1j}, x_j) \\ \bigwedge_{j=1}^{n} \beta(p_{2j}, x_j) \\ . \\ . \\ . \\ \\ \bigwedge_{j=1}^{n} \beta(p_{mj}, x_j) \end{pmatrix} \tag{3.11}$$

3.2.3. Memorias heteroasociativas Alfa-Beta

Se tienen dos tipos de memorias heteroasociativas Alfa-Beta ($\alpha\beta$) tipo Max ($\bigvee$) denotada por M y su ij-ésima componente como m_{ij} y las tipo Min, denotadas por W y su ij-ésima componente como w_{ij}. En la generación de ambos tipos de memorias se usará el operador $\otimes$, el cual como se explicó (véase 3.2.2) tiene la siguiente forma:

$$\left[y^{\mu} \otimes (x^{\mu})^{t}\right]_{ij} = \alpha\left(y_i^{\mu}, x_j^{\mu}\right) \mid \mu\epsilon\{1,2,...p\},\ i\epsilon\{1,2,...m\},\ j\epsilon\{1,2,...n\} \tag{3.12}$$

3.2.3.1. Algoritmo de la Memoria Heteroasociativa Alfa-Beta tipo Max ($\bigvee$)

Fase de Aprendizaje de la memoria heteroasociativa $\alpha\beta$ tipo ($\bigvee$)

Para la fase de aprendizaje se siguen dos pasos, en el primer paso se utiliza el operador $\otimes$, mientras que en el segundo paso se hace uso del operador máximo $\bigvee$.

Algoritmo 3.1 Algoritmo de memoria heteroasociativa Afa-Beta tipo Max ($\bigvee$)

Paso 1. Para cada $\mu = 1, 2, ...p$, a partir de la pareja (x^{μ}, y^{μ}) se construye la matriz

$$\left[y^{\mu} \otimes (x^{\mu})^{t}\right]_{mxn} \tag{3.13}$$

Paso 2. Aplicar el operador binario máximo $\bigvee_{p}$ a las matrices obtenidas en el paso 1:

$$M = \bigvee_{\mu=1}^{p} \left[y^{\mu} \otimes (x^{\mu})^{t}\right] \tag{3.14}$$

La entrada ij-ésima está dada por la siguiente expresión:

$$m_{ij} = \bigvee_{\mu=1}^{p} \alpha\left(y_i^{\mu}, x_j^{\mu}\right) \tag{3.15}$$

y de acuerdo con que $\alpha = AxA \rightarrow B$, se tiene que $m_{ij}\ \epsilon\ B$, $\forall i\ \epsilon\ \{1,2,..,m\}$, $\forall j\ \epsilon\ \{1,2,..,n\}$.

Fase de recuperación de la memoria heteroasociativa $\alpha\beta$ tipo ($\bigvee$)

Para la fase de recuperación de las memorias heteroasociativas presentan dos casos posibles. La diferencia fundamental es que el primer caso el patrón de entrada es un patrón fundamental x^{ω}; es decir, la entrada es un patrón $\omega\ \epsilon\ \{1,2,...,p\}$; mientras que en el segundo caso, el patrón de entrada NO es un patrón fundamental, sino la versión distorsionada de por lo menos uno de los patrones fundamentales; lo anterior significa que si el patrón de entrada es $\widetilde{x}$, debe existir al menos un valor de índice $\omega\ \epsilon\ \{1,2,...,p\}$, que corresponde al patrón fundamental respecto del cual $\widetilde{x}$ es una versión alterada con alguno de los tres tipos de ruido: aditivo, sustractivo o mezclado.

Lo primero que se hace es presentarle un patrón x^{ω}, donde $\omega\epsilon\{1,2,...p\}$, a la memoria heteroasociativa $\alpha\beta$ la cual es de tipo $\bigvee$, posteriormente se realiza la operación $\triangle_{\beta}$: $M\triangle_{\beta}x^{\omega}$.

Ahora bien, considerando las dimensiones de la matriz M son de mxn y el patrón que se desea recuperar x^{ω} es un vector columna de dimensión n, por lo que el resultado de la operación anterior debe ser un vector columna de dimensión m, cuya i-ésima componente es:

$$(M \triangle_{\beta} x^{\omega})_i = \bigwedge_{j=1}^{r} \beta\left(m_{ij}, x_j^{\omega}\right) \tag{3.16}$$

3.2.3.2. Algoritmo de la memoria heteroasociativa Alfa-Beta tipo Min ($\bigwedge$)

Fase de aprendizaje

Algoritmo 3.2 Algoritmo de la memoria heteroasociativa Alfa-Beta tipo Min ($\bigwedge$)

Paso 1. Para constituir la matriz se considera para cada $\mu = 1, 2, ...p$, a partir de la pareja $(x^{\mu},\, y^{\mu})$

$$\left[y^{\mu} \otimes \left(x^{\mu}\right)^{t}\right]_{mxt} \tag{3.17}$$

Paso 2. Después se aplica el operador binario mínimo $\bigwedge$ a las matrices obtenidas en el paso 1:

$$W = \bigwedge_{\mu=1}^{p} \left[y^{\mu} \otimes \left(x^{\mu}\right)^{t}\right] \tag{3.18}$$

Por lo que la entrada ij-ésima está dada por la siguiente expresión:

$$w_{ij} = \bigwedge_{\mu=1}^{p} \alpha\left(y_i^{\mu},\, x_j^{\mu}\right) \tag{3.19}$$

Fase de recuperación

En la fase de recuperación se le presenta un patrón x^{ω}, donde $\omega\, \epsilon\, \{1, 2, ...p\}$, a la memoria heteroasociativa $\alpha\beta$ tipo $\bigwedge$ y se realiza la operación $\triangle_{\beta}:\ W \nabla_{\beta} x^{\omega}$.

De forma que si se consideran las dimensiones de la matriz Min W (Construida en la fase de aprendizaje) son de mxn, y el patrón que se desea recuperar es x^{ω} el cual es un vector columna de dimensión n, el resultado de la operación anterior debe ser un vector columna de dimensión m, cuya i-ésima componente es:

$$(W \triangle_{\beta} x^{\omega})_i = \bigvee_{j=1}^{r} \beta\left(w_{ij}, x_j^{\omega}\right) \tag{3.20}$$

3.2.4. Memorias autoasociativas Alfa-Beta

Si a una memoria heteroasociativa se le impone la condición de que $y^{\mu} = x^{\mu} \forall\, \mu\, \epsilon\, \{1, 2, ..., p\}$, entonces deja de ser heteroasociativa y ahora se le denomina autoasociativa.

A continuación se enlistan algunas de las características de las memorias autoasociativas Alfa-Beta:

1. El conjunto fundamental toma la forma $\{(x^{\mu}, x^{\mu}) \mid \mu = 1, 2, ...p\}$

2. Los patrones fundamentales de entrada y salida son de la misma dimensión; denotada por n.
3. La memoria es una matriz cuadrada, para ambos tipos, $\bigvee$ y $\bigwedge$. Si $x^{\mu}\epsilon A^{n}$ entonces:

$$\bigvee = \lfloor v_{ij} \rfloor_{mxn} \; y \; \bigwedge = \lfloor \lambda_{ij} \rfloor_{mxn} \tag{3.21}$$

3.2.4.1. Algoritmo de la memoria autoasociativa Alfa-Beta tipo Max ($\bigvee$)

Las fases de aprendizaje y recuperación son similares a las memorias heteroasociativas Alfa-Beta, las cuales se presentan a continuación.

Fase de aprendizaje de la memoria autoasociativa tipo Max ($\bigvee$)

Algoritmo 3.3 Fase de aprendizaje de las memorias autoasociativas tipo Max ($\bigvee$)

Paso 1. Para cada $\mu = 1, 2, ...p$, a partir de la pareja (x^{μ}, x^{μ}) se construye la matriz

$$\left[x^{\mu} \otimes (x^{\mu})^{t}\right]_{mxn} \tag{3.22}$$

Paso 2. Se aplica el operador binario máximo $\bigvee$ a las matrices obtenidas en el paso 1:

$$M = \bigvee_{\mu=1}^{p} \left[x^{\mu} \otimes (x^{\mu})^{t}\right]_{mxn} \tag{3.23}$$

La entrada ij-ésima de la memoria está dada así:

$$m_{ij} = \bigvee_{\mu=1}^{p} \alpha\left(x_{i}^{\mu}, x_{j}^{\mu}\right) \tag{3.24}$$

y de acuerdo con que $\alpha = AxA \rightarrow B$, se tiene que $m_{ij} \epsilon B$, $\forall j \epsilon \{1, 2, .., n\}$, $\forall i \epsilon \{1, 2, .., m\}$.

Fase de recuperación de la memorias autoasociativa tipo Max ($\bigvee$)

Para la fase de recuperación de las memorias autoasociativas Alfa-Beta tipo $\bigvee$ presentan dos casos posibles. La diferencia fundamental es que en el primer caso el patrón de entrada es un patrón fundamental; es decir, la entrada es un patrón $\omega \epsilon \{1, 2, ..., p\}$; mientras que en el segundo caso, el patrón de entrada NO es un patrón fundamental, sino la versión distorsionada de por lo menos uno de los patrones fundamentales; lo anterior significa que si el patrón de entrada es $\widetilde{x}$, debe existir al menos un valor de índice $\omega \epsilon \{1, 2, ..., p\}$, que corresponde al patrón fundamental respecto del cual $\widetilde{x}$ es una versión alterada con alguno de los tres tipos de ruido: aditivo, sustractivo o mezclado. Analizando más a detalle ambos casos se tiene que:

Caso 1. En donde se considera un *patrón fundamental*. Se presenta un patrón x^{ω}, con $\omega \epsilon \{1, 2, ..., p\}$, a la memoria autoasociativa tipo $\alpha\beta$ y se realiza la operación $\triangle_{\beta}$:

$$\bigvee \triangle_{\beta} x^{\omega} \tag{3.25}$$

El resultado de la operación anterior será el vector columna de dimensión n.

$$\left(\bigvee \triangle_\beta x^\omega\right) = \bigwedge_{j=1}^{n} \beta\left(m_{ij}, x_j^\mu\right) \tag{3.26}$$

$$\left(\bigvee \triangle_\beta x^\omega\right) = \bigwedge_{j=1}^{n} \beta\left\{\left(\bigvee_{\mu=1}^{p} \alpha\left(x_i^\mu, x_j^\mu\right), x_j^\mu\right)\right\} \tag{3.27}$$

Lema 6. Memoria asociativa $\alpha\beta$ ***«Una memoria autoasociativa Alfa-Beta $\alpha\beta$ tipo Max $\bigvee$ tiene únicamente unos en su diagonal principal»***

Demostración. La ij-ésima de la memoria autoasociativa Alfa-Beta $\alpha\beta$ tipo Max $\bigvee$ está dada por $m_{ij} = \bigvee_{\mu=1}^{p} \alpha\left(x_i^\mu, x_j^\mu\right)$, de acuerdo con la fase de aprendizaje (Véase 3.2.4.1). Las entradas de la diagonal principal se obtiene de la expresión anterior haciendo que $i = j$. de tal forma que:

$$m_{ij} = \bigvee_{\mu=1}^{p} \alpha\left(x_i^\mu, x_i^\mu\right), \ \forall i \, \epsilon \, \{1, 2, .., n\} \tag{3.28}$$

Esto es posible gracias a la propiedad de α presentada en la tabla 3.1, se tiene que $\alpha(x_i^\mu, x_i^\mu) = 1$, entonces la expresión anterior se convierte en:

$$m_{ij} = \bigvee_{\mu=1}^{p} (1) = 1, \ \forall \, \epsilon \, \{1, 2, .., n\} \tag{3.29}$$

Teorema 5. Memoria asociativa $\alpha\beta$ y conjunto fundamental . ***«Una memoria asociativa $\alpha\beta$ tipo $\bigvee$, recupera de manera correcta el conjunto fundamental completo: además tienen máxima capacidad de aprendizaje ... »***

$$\bigvee \triangle_\beta x^\omega = x^\omega, \ \forall \omega \, \epsilon \, \{1, 2, ..., p\} \tag{3.30}$$

Demostración. Sea $\omega \, \epsilon \, \{1, 2, ..., p\}$ arbitrario y considerando el **Lema 6** descrito anteriormente, para cada $i \, \epsilon \, \{1, 2, ..., n\}$ escogida arbitrariamente:

$$m_{ii} = 1 = \alpha\left(x_i^\mu, x_i^\mu\right) \tag{3.31}$$

Lo cual nos indica que la memoria autoasociativa $\alpha\beta$ tipo $\bigvee$ recuperará de forma correcta todo el conjunto fundamental aprendido. Además es la demostración del teorema 5.

Caso 2. En donde se considera un *patrón alterado*. Se presenta el patrón binario $\tilde{x}$ (un patrón alterado de algún patrón fundamental x^ω) el cual es un vector columna de dimensión n, a la memoria autoasociativa Alfa-Beta tipo $\bigvee$ y después se realiza la operación mostrada a continuación.

$$\bigvee \triangle_\beta \widetilde{x} \tag{3.32}$$

Al igual que en el caso 1, el resultado de la operación anterior es un vector columna de dimensión n, cuya *i-ésima* componente se expresa de la siguiente manera:

$$(\bigvee \triangle_\beta \widetilde{x}) = \bigwedge_{j=1}^{n} \beta(v_{ij}, \widetilde{x}) \tag{3.33}$$

$$(\bigvee \triangle_\beta \widetilde{x}) = \bigwedge_{j=1}^{n} \beta \left\{ \left[\bigvee_{\mu=1}^{p} \alpha \left(x_i^\mu, x_j^\mu \right) \right], \widetilde{x_j} \right\} \tag{3.34}$$

De acuerdo con el Teorema 5, que puede consultarse en [58]:

$$\bigvee \triangle_\beta x^\omega = x^\omega, \forall \omega \in \{1, 2, \ldots, p\} \tag{3.35}$$

Esto significa que la memoria autoasociativa Alfa-Beta tipo $\bigvee$ recupera de manera correcta el conjunto fundamental completo. Además, en la demostración de este Teorema, en ningún momento aparece restricción alguna sobre p, que es la cardinalidad del conjunto fundamental; y esto quiere decir que el conjunto fundamental puede crecer tanto como se quiera. La consecuencia directa es que el número de patrones que puede aprender una memoria autoasociativa Alfa-Beta tipo $\bigvee$, con recuperación correcta, es máximo.

3.2.4.2. Algoritmo de la memoria autoasociativa Alfa-Beta tipo Min ($\bigwedge$)

Fase de Aprendizaje de la memoria autoasociativa tipo Min ($\bigwedge$)

Algoritmo 3.4 Fase de aprendizaje de la memoria autoasociativa Alfa-Beta tipo Min ($\bigwedge$)

Fase de Aprendizaje

Paso1. Para cada $\mu = 1, 2, ..., p$, a partir de la pareja (x^μ, x^μ) se construye la matriz

$$\left[x^\mu \otimes (x^\mu)^t \right]_{n \times n} \tag{3.36}$$

Paso2. Se aplica el operador binario máximo $\bigvee$ a las matrices obtenidas en el paso 1:

$$\bigwedge = \bigvee_{\mu=1}^{p} \left[x^\mu \otimes (x^\mu)^t \right] \tag{3.37}$$

La entrada ij-ésima está dada por la siguiente expresión:

$$\lambda_{ij} = \bigvee_{\mu=1}^{p} \alpha(x_i^\mu, x_j^\mu) \tag{3.38}$$

y de acuerdo con que α: $A \times A \to B$, se tiene que $\lambda_{ij} \in B$, $i \in \{1, 2, \ldots, n\}$. $\forall j \in \{1, 2, \ldots, n\}$.

Fase de Recuperación de la memoria autoasociativa tipo Min($\bigwedge$)

La fase de recuperación de las memorias autoasociativas Alfa- Beta tipo $\bigwedge$ tiene dos casos posibles. En el primer caso el patrón de entrada es un patrón fundamental; es decir, la entrada es un patrón x^ω, con $\omega \in \{1, 2, \ldots, p\}$. En el segundo caso, el patrón de entrada NO es un patrón fundamental, sino la versión distorsionada de por lo menos uno de los patrones fundamentales; lo anterior significa que si el patrón de entrada es $\widetilde{x}$, debe existir al menos un valor de índice $\omega \in \{1, 2, \ldots, p\}$, que corresponde al patrón fundamental respecto del cual $\widetilde{x}$ es una versión alterada con alguno de los tres tipos de ruido: aditivo, sustractivo o mezclado.

Caso 1. *Patrón fundamental.* Se presenta un patrón x^ω, con w $\in \{1, 2, ..., p\}$, a la memoria autoasociativa $\alpha\beta$ tipo $\bigvee$ y se realiza la operación ∇_β:

$$\bigwedge \nabla_\beta x^\omega. \tag{3.39}$$

El resultado de la operación anterior será el vector columna de dimensión n.

$$(\bigwedge \nabla_\beta x^\omega) = \bigvee_{j=1}^{n} \beta(\lambda_{ij}, x_j^\mu) \tag{3.40}$$

$$(\bigwedge \nabla_\beta x^\omega) = \bigwedge_{j=1}^{n} \beta \left\{ \left[\bigvee_{\mu=1}^{p} \alpha \left(x_i^\mu, x_j^\mu \right) \right], \, x_j^\omega \right\} \tag{3.41}$$

Caso 2. *Patrón alterado.* Se presenta el patrón binario $\widetilde{x}$ (patrón alterado de algún patrón fundamental x^ω) que es un vector columna de dimensión n, a la memoria autoasociativa Alfa-Beta tipo $\bigwedge$ y se realiza la operación

$$\bigwedge \nabla_\beta \widetilde{x} \tag{3.42}$$

Al igual que en el caso 1, el resultado de la operación anterior es un vector columna de dimensión n, cuya *i-ésima* componente se expresa de la siguiente manera:

$$(\bigwedge \nabla_\beta \widetilde{x}) = \bigvee_{j=1}^{n} \beta(\lambda_{ij}, \widetilde{x}) \tag{3.43}$$

$$(\bigwedge \nabla_\beta \widetilde{x}) = \bigwedge_{j=1}^{n} \beta \left\{ \left[\bigvee_{\mu=1}^{p} \alpha \left(x_i^\mu, x_j^\mu \right) \right], \, \widetilde{x_j} \right\} \tag{3.44}$$

De acuerdo con el Teorema 4.7, que puede consultarse en [58]:

$$\bigvee \nabla_\beta x^\omega = x^\omega, \, \forall \omega \in \{1, 2, \ldots, p\} \tag{3.45}$$

Esto significa que la memoria autoasociativa Alfa-Beta tipo V recupera de manera correcta el conjunto fundamental completo. Además, en la demostración de este Teorema, en ningún momento aparece restricción alguna sobre p, que es la cardinalidad del conjunto fundamental; y esto quiere decir que el conjunto fundamental puede crecer tanto como se quiera. La consecuencia directa es que el número de patrones que puede aprender una memoria autoasociativa Alfa-Beta tipo $\bigwedge$, con recuperación correcta, es máximo.

3.3. Descripción del entorno en Microsoft Robotics Developer Studio

Microsoft Robotics Developer Studio (MRDS) es una plataforma de programación y simulación de robots que trabaja bajo el sistema operativo de Windows, nace bajo la creciente necesidad de un entorno versátil de programación para robots, con potentes herramientas de fácil uso e intuitivas funciones, que facilitan el aprendizaje de la robótica y que a su vez sean capaz de proveer soluciones a los diversos problemas existentes en esta área. Es por ello que en el año 2006, Microsoft incursionó en el desarrollo de Robotics Studio, el cual es producto del arduo trabajo de profesionales en el área, que proponen este entorno de programación, como una solución robusta a los problemas y carencias más comunes de los que adolecen programas similares.

MRDS proporciona un módulo de tiempo de ejecución orientado a servicios, además de las herramientas necesarias para diseñar e implementar aplicaciones basadas en robótica. Incluye herramientas, tutoriales y documentación para crear elementos visuales, diseñados para introducir rápidamente a los diseñadores con poca experiencia en el mundo de la robótica. Los desarrolladores comerciales deberán adquirir el kit de herramientas por una pequeña cantidad, pero los desarrolladores aficionados e investigadores académicos podrán descargarlo y usarlo de forma gratuita.

Algunas ventajas importantes de esta plataforma son:

1. Permite el control de una importante gama de robots físicos que a su vez pueden ser simulados utilizando el mismo código fuente del controlador.

2. Simulación en entornos virtuales en tres dimensiones utilizando la tecnología AGEIA PhysX. Gracias a ésta, se puede simular con alto grado de precisión la dinámica e interacciones físicas entre los elementos del entorno simulado.

3. Arquitectura orientada a servicios, que hacen uso de protocolos denominados Decentralized Software Services (DSS). Estos permiten manejar un robot como un conjunto de servicios que se ejecutan en paralelo y administración de recursos que gestiona la entrada/salida asíncrona, la concurrencia y la distribución de servicios

4. Manejo sencillo en programación en paralelo concurrente, esto se debe al uso de bibliotecas *Concurrency and Coordination Runtime* (CCR), las cuales permiten gestionar la *concurrencia y coordinación de los servicios* (DSS). También proveen directivas sencillas que permiten el uso de la programación asíncrona, las cuales son de vital importancia para coordinar los diferentes procesos que integran a un robot.

5. Proporciona un lenguaje de programación denominado *Visual Programming Language* (VPL), el cual brinda una forma sencilla de programación basada en bloques de los diferentes servicios de los que hace uso un robot móvil para el control de sus recursos. Este lenguaje de programación lógico-visual resulta bastante intuitivo, además de que ayuda en gran medida al aprendizaje de la programación orientada a servicios utilizando DSS.

6. Control de un dispositivo físico a través de una red inalámbrica o directamente desde un robot con PC integrada, el cual incluya una versión actual de Microsoft Windows.

7. Es una plataforma diseñada para usarse con gran diversidad de hardware y fabricantes de robots, ya que existe una importante variedad de servicios que pueden ser empleados directamente con los modelos de robots más populares.

8. MRDS soporta lenguajes de programación incluidos en Microsoft Visual Studio y Microsoft Visual Studio Express sea (C# y VB.NET), así como lenguajes de scripting como Microsoft Iron Python.

Todas estas ventajas y muchas otras hacen que MRDS sea en esencia una útil y versátil herramienta de programación y simulación de robots. Sin lugar a dudas una de sus principales aportaciones de MRDS es acercar el mundo de la robótica a los investigadores, profesores, estudiantes y comunidad interesada, los cuales en algunos casos no poseen profundos conocimientos en el manejo y control de los recursos físicos de un robot, y el beneficio también es inverso para personas que no poseen grandes conocimientos de programación pero si poseen amplios conocimientos de hardware, ya que tienen la posibilidad de programar de forma fácil un robot.

Una vez enmarcadas las ventajas de MRDS, es de importancia ampliar el marco conceptual para saber cómo es que esta plataforma brinda una extensa posibilidad de aplicaciones:

Programación para el control de dispositivos integrados en robots móviles. La plataforma permite programar en C# y VB.NET, JScript y Python, pero también incluye una herramienta de desarrollo visual (Programación a bloques), la cual permite crear y depurar de forma sencilla servicios modulares de hardware y software para interactuar con robots como Pioneer P3DX, Lego Mindstorms NXT, iRobot, Roomba, entre otros robots disponibles. Todo ello bajo el sistema operativo Windows y agregando la versatilidad de los servicios web. Además MRDS permite crear entornos virtuales de simulación en 3D para probar el software sin disponer de robots reales, esto es gracias a que Microsoft ha licenciado el motor de simulación física de AEGIA.

Bibliotecas de ejecución orientadas a servicios. Gracias a su programación orientada a servicios permite desarrollar aplicaciones asíncronas (propias de este tipo de desarrollos), ya que mantiene un alto nivel de abstracción, al basarse en una arquitectura de mensajes convierte en algo simple el acceso al estado de los sensores y actuadores de los robots, incluso desde aplicaciones web de control remoto, lo cual permite deslindarse de las complejidades inherentes del control de motores, sensores y demás dispositivos que forman a un robot, así como del control del tráfico en la utilización de los recursos.

Plataforma extensible. MRDS soporta una amplia variedad de hardware de robots, brindando la viabilidad de reutilización de código para ser empleado en proyectos con robots diferentes. Además te permite ampliar el alcance de la plataforma robótica escribiendo extensiones propias en forma de bibliotecas y servicios.

Ahora bien, la simulación es una parte esencial de esta plataforma ya que permite experimentar con robots nuevos incluso antes de construirlos físicamente. También se pueden desarrollar aplicaciones para robots existentes y luego probar los resultados en los robots físicos. Además, nos permite eliminar errores y problemas antes de probar la aplicación en el robot físico.

3.3.1. Entorno de simulación

Las opciones de configuración de las características físicas se reducen a la aceleración de la gravedad (modificable) y las opciones de simular en tiempo real o a intervalos, opción útil para PC´s poco potentes sin tarjeta de aceleración AGEIA.

El entorno de simulación es un entorno 3D donde tenemos 4 opciones de visualización:

- Visual: Renderiza una imagen completamente en 3D, con una iluminación y shaders de forma realista (Soporta Píxel Shaders 3.0).
- Sin Renderizar: Recrea la imagen de forma que sólo se ven la estructura de cada entidad, sin texturizar, de esta forma puedes tener una idea de cuantos polígonos está hecha cada entidad y donde están los vértices.
- Físico: Muestra los vectores de cada entidad, de esta forma se puede ver el modelo físico durante una simulación.
- Combinado: Renderiza la imagen 3D con los vectores.

Ventajas y Desventajas de la simulación
Todo programador que domine diversos entornos de programación, sabe que la simulación es un elemento clave a la hora de desarrollar una aplicación. En este caso, la relevancia que adquiere disponer de un simulador avanzado es vital.

Las ventajas de tener un simulador son múltiples:

- Se puede probar en situaciones que comprometerían la funcionalidad de un prototipo.
- El personal para probar el prototipo simulado es menor.
- El aprendizaje es rápido y permite modificaciones sobre la marcha.
- Se puede trabajar sobre modelos que todavía no están diseñados completamente.
- El coste del hardware final es costoso.

No obstante, siempre que se trabaja con emuladores/simuladores existen ciertos inconvenientes al no aplicar el software sobre el aparato final:

- Algunos efectos secundarios propios de trabajar con robots no se ven reflejados en la simulación. Un ejemplo puede ser la supresión de ruido, el ruido suele crear valores poco fidedignos en las lecturas de los sensores, pero en una simulación el robot no se ve afectado por el ruido.
- Al trabajar con un modelo simulado, siempre aparecen problemas a la hora de llevar el proyecto a una fase de desarrollo.
- Requiere la creación del Modelo y pueden surgir múltiples inconvenientes técnicos.

Simulation Engine Service
Microsoft Robotics Studio proporciona ***Simulation Engine Service***, es la herramienta que permitirá simular y recrear las condiciones establecidas para poder determinar las reacciones del robot.

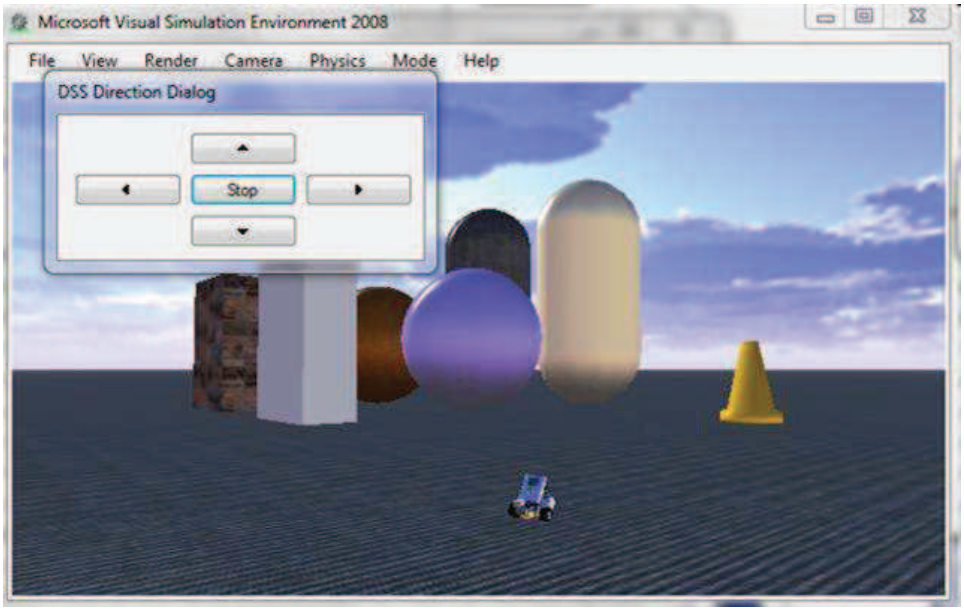


Figura 3.2: Ejemplo de entorno de simulación en MRDS.

Las características del Simulator Engine Service son las siguientes:

- Está implementado como un servicio.
- Mantiene el estado de la aplicación.
- Los desarrolladores pueden simular las aplicaciones, utilizando modelos 3D realistas que generan condiciones reales con la tarjeta AGEIA™ PhysX™ Technology. AGEIA es la única empresa que tiene en el mercado una solución hardware para cálculos físicos.
- El renderizado 3D utiliza XNA, que implementa Direct X. XNA es una API para el desarrollo de videojuegos para PC.
- Controla dispositivos de entrada, como son webcam, teclado, controles Xbox entre otros.
- Tiene un editor para modelar objetos del mundo real, y depurarlos.
- Permite importar recreaciones 3D de programas destinados al diseño.
- Dentro del Simulador se manejan entidades, que son objetos en el mundo de la simulación. Diversos ejemplos pueden ser cámaras, cielo, fondos, componentes de un robot, motores y sensores, construcciones, muebles, o bien cualquier elemento que sea físico y visible.

Una descripción más detallada sobre el uso de MRDS puede verse en el apéndice D.

Capítulo 4

Experimentos y Resultados

Resultados y validación del sistema propuesto

La finalidad este trabajo de investigación es probar las memorias asociativas Alfa-Beta en la tarea de localización de un robot móvil, para ello se han diseñado dos experimentos que permiten evaluar el desempeño de las memorias asociativas en la solución del presente problema.

A continuación de describen detalladamente las condiciones y características de cada experimento, por lo que a partir de este momento se hará referencia a dichos experimentos como "Experimento n. 1" y "Experimento n. 2".

Experimento n.1. La finalidad es brindar una primera aproximación de los resultados que se obtienen haciendo uso de las memorias asociativas Alfa-Beta en la localización de un robot móvil. Para ello se ha dispuesto emplear una memoria asociativa Alfa-Beta utilizando uno de los bancos de datos llamado «Wall-Following Robot Navigation Data», disponibles en el repositorio de UCI el cual pertenece a la Universidad de California del Center for Machine Learning and Intelligent Systems. En donde se mantienen 211 bancos de datos disponibles, los cuales pueden ser usados por la comunidad interesada en la aplicación del aprendizaje automático[1] . De manera que en este primer experimento se obtendrán resultados que permitirán visualizar una tendencia preliminar del empleo de este tipo de memorias en un simulador durante el "Experimento n.2".

Experimento n.2. En este experimento se propone la utilización de un simulador de robots para visualizar el comportamiento de las memorias asociativas Alfa-Beta. En este experimento se hace uso de las herramientas de simulación que proporciona la plataforma de Microsoft Robotics Developer Studio (MRDS) (Véase la sección 3.3 y el apéndice D). El robot seleccionado es un Lego Mindstorm NXT que es un robot compatible con MRDS.

Ambos experimentos (Experimento n.1 y Experimento n.2) serán descritos a continuación:

4.1. Experimento n.1. Empleado banco datos de UCI

Como ya se mencionó en la introducción del capítulo, para la realización de este experimento se hará una preselección considerando los 211 bancos de datos disponibles en el repositorio de UCI de la

[1] Consulte en http://archive.ics.uci.edu/ml/

Universidad de California, por lo tanto para este experimento se han propuesto los siguientes pasos a seguir:

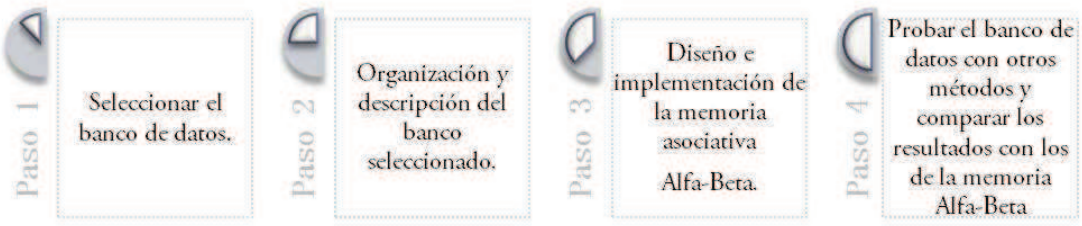
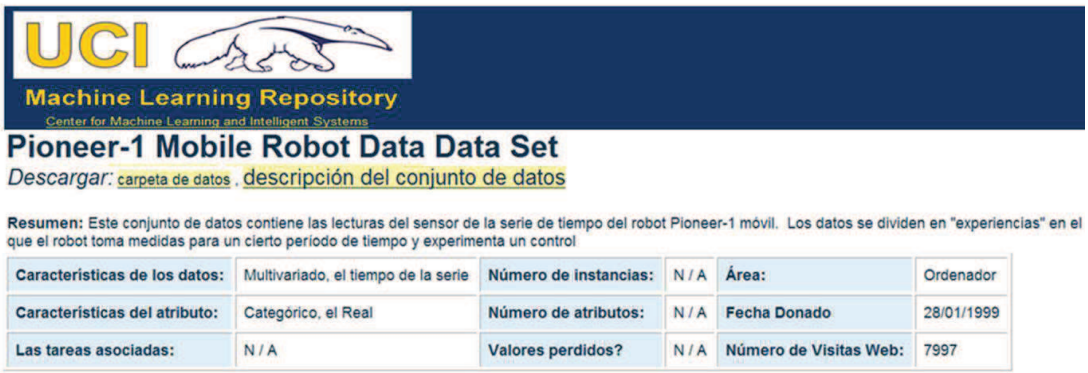


Figura 4.1: Secuencia de pasos a seguir durante el Experimento n.1.

4.1.1. Paso 1. Seleccionar el banco de datos

Durante el desarrollo de esta primera etapa de experimentación se realiza una revisión de los bancos de datos disponibles que son concernientes a robots móviles y puedan ser empleados en el problema de localización. A continuación se muestra la preselección de los bancos de datos disponibles en el repositorio de UCI, los cuales están enfocados en la recopilación de datos que provienen de los diferentes sensores que equipan el robot móvil.

Primera opción. Banco de datos "Pioneer-1 Mobile Robot".


UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

Pioneer-1 Mobile Robot Data Data Set

Descargar: carpeta de datos , descripción del conjunto de datos

Resumen: Este conjunto de datos contiene las lecturas del sensor de la serie de tiempo del robot Pioneer-1 móvil. Los datos se dividen en "experiencias" en el que el robot toma medidas para un cierto período de tiempo y experimenta un control

Características de los datos:	Multivariado, el tiempo de la serie	**Número de instancias:**	N / A	**Área:**	Ordenador
Características del atributo:	Categórico, el Real	**Número de atributos:**	N / A	**Fecha Donado**	28/01/1999
Las tareas asociadas:	N / A	**Valores perdidos?**	N / A	**Número de Visitas Web:**	7997

Figura 4.2: Banco de datos de Pioner-1 Mobile Robot extraído del Repositorio UCI.

Información de relevancia

- **Donada por:** Matthew D. Schmill, Paul R. Cohen; Experimental Knowledge Systems Laboratory; Department of Computer Science; University of Massachusetts, Amherst.
- **Descripción:** Los datos fueron recogidos durante una serie de pruebas diseñadas específicamente. Las pruebas cubren con la mayoría de los tipos de interacciones sensoriales que un robot «Pioneer» podría enfrentar bajo circunstancias controladas: por ejemplo evadir objetos visibles, empujando a los objetos visibles, estrellándose contra las paredes, entre otros; muchas de estas interacciones

se repiten en todo el conjunto de datos. El banco de datos incluye seis sensores ultrasónicos, información de encoders que brindan datos de la velocidad de cada rueda del robot, información del griper y de imágenes de las cámaras.

Segunda opción. Banco de datos "Mobile Robots"

Figura 4.3: Banco de datos de «Mobile Robots» extraído del Repositorio UCI.

Información de relevancia

- **Donada por:** Volker Klingspor, Katharina J. Morik, Anke D. Rieger; Computer Science Dept. LS VIII; University of Dortmund, Germany.
- **Descripción:** Se presenta una serie de conjuntos de datos, donde cada conjunto de datos corresponde al aprendizaje de conceptos inconexos en un nivel. Consideran una serie de aspectos para constituir los segmentos de datos, tales como el tiempo, datos de sensor, orientación, distancias, coordenadas, movimiento diagonal, movimiento direccional, características perceptivas, entre otros.

Tercera y última opción. Banco de datos "Wall-Following Robot Navigation"


UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

Wall-Following Robot Navigation Data Data Set

Descargar: carpeta de datos , descripción del conjunto de datos

Resumen: Los datos fueron recolectados como el G5 SCITOS robot se desplaza por la sala siguiendo la pared en el sentido de las agujas del reloj, de 4 rondas, con 24 sensores de ultrasonidos colocados circularmente en torno a su "cintura".

Características de los datos:	Multivariado, secuencial	**Número de instancias:**	5456	**Área:**	Ordenador
Características del atributo:	Real	**Número de atributos:**	24	**Fecha Donado**	04/08/2010
Las tareas asociadas:	Clasificación	**Valores perdidos?**	N / A	**Número de Visitas Web:**	12258

Figura 4.4: Banco de datos de «Wall-Following Robot Navigation»" extraído del Repositorio UCI.

Información de relevancia

- **Donada por:** Ananda Freire, Marcus Veloso and Guilherme Barreto, Department of TeleinformaticsEngineering, Federal University of Ceará, Fortaleza, Ceará, Brazil.
- **Descripción:** Los archivos proporcionados comprenden tres conjuntos de datos diferentes. El primero consta de lecturas de 24 sensores ultrasónicos que forman parte del robot, con tasa de muestreo de 9 tomas por segundo. El segundo y tercero son un compilado que solo considera a 4 y 2 sensores respectivamente.

4.1.1.1. Banco de datos seleccionado "Wall-Following Robot Navigation"

Hasta este punto se ha realizado un análisis general, considerando aspectos como: el tipo de datos proporcionado por cada uno de los tres bancos de datos, revisar si se incluía información sobre el o los mapas de la navegación del robot en el momento de las tomas de lecturas de los sensores, así como también una revisión rápida de los artículos relacionados con los bancos de datos, los cuales proporcionan información más rica y significativa, finalmente se ha concluido que el banco de datos que más se acerca a las características que son necesarias para lograr la localización de un robot móvil, es el tercer banco "Wall-Following Robot Navigation".

A continuación se presenta un estudio más detallado del banco de datos seleccionado:

- Los datos fueron recolectados mientras el robot G5 SCITOS navegaba a través de la sala. El robot esta equipado por 24 sensores ultrasónicos colocados circularmente en torno a su "cintura". La numeración de los sensores ultrasónicos inicia en el frente del robot y va aumentando 15° en sentido de las manecillas del reloj.
- Los archivos que se proporcionan están integrados por tres conjuntos de datos.

- El primero contiene los valores brutos de las mediciones de los veinticuatro sensores de ultrasónicos además incluye una etiqueta que indica la clase correspondiente. Las lecturas del sensor fueron obtenidas con una tasa de 9 muestras por segundo.
- El segundo contiene cuatro lecturas de los sensores llamados “distancias simplificadas” y también incluye la etiqueta de la clase a la que pertenece cada patrón. Estas distancias simplificadas corresponden a la “distancia frontal”, “distancia del costado izquierdo”, “distancia del costado derecho” y “distancia trasera”.
- Por último en el tercer archivo solo se encuentran las lecturas de los sensores que corresponden a la “distancia frontal” y a la “distancia de costado izquierdo”, incluyendo la etiqueta de la clase correspondiente.
- Un aspecto importante que se menciona en la información de este banco de datos es que las lecturas de 24 sensores ultrasónicos y las distancias simplificadas han sido lecturas que fueron tomadas en el mismo paso de tiempo, por lo que cada archivo tiene el mismo número de filas (una para cada paso de tiempo de muestreo).

≻ El banco originalmente contiene 5,456 instancias y el número de atributos se encuentran distribuidos según cada archivo de la siguiente manera:

- Sensor_readings_24.data: 24 atributos numéricos y la clase.
- Sensor_readings_4.data: 4 atributos numéricos y la clase.
- Sensor_readings_2.data: 2 atributos numéricos y la clase.

Originalmente este banco de datos es empleado en el artículo [1], en el que se compara la precisión de los clasificadores Multilayer Perceptrón (MLP), Mixture of Local Experts (ME) y Elman Recurrent Network (obtenida del original MLP) empleados en el problema de navegación, en el artículo se hace uso de un robot G5 SCITOS, el cual fue probado bajo el mapa mostrado en la figura 4.5.

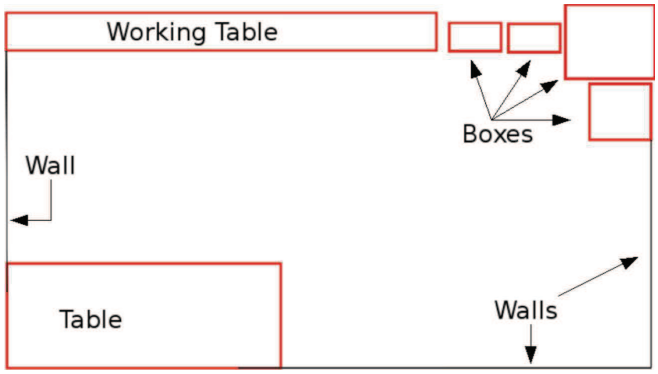


Figura 4.5: Esquema de navegación del robot del banco de datos «Wall-Following Robot Navigation» [1].

El banco de datos pertenece al conjunto de información extraída por los 24 sensores en forma de anillo con una frecuencia de 9 muestras por segundo, bajo un algoritmo de navegación sencillo, basado en reglas condicionales (IF =>THEN).

En la investigación de Freire et al.[1] se presenta el algoritmo 4.1, el cual se basa en reglas condicionales simples empleadas para la navegación del robot, en el algoritmo se considera la medición de la distancia frontal y de la distancia del costado izquierdo, es importante mencionar que durante la navegación del robot se almacenan todas las mediciones tomadas de los sensores, independientemente de las consideraciones de navegación del mismo.

Algoritmo 4.1 Algoritmo básico con lógicas condicionales para la navegación del robot G5 SCITOS [1]

```
If distancia_izquierda> 0.9
        then
            if distancia_frontal<= 0.9
                then <<Para y da vuelta a la derecha>>
                else <<Desacelerar y dar vuelta a la Izquierda>>
        else
            if distancia_frontal<= 0.9
                then<<Para y da vuelta a la derecha>>
                else
                    if distancia_izquierda< 0.55
                        then<<Desacelerar y dar vuelta a la derecha>>
                        else<<Seguir adelante>>
```

El esquema de la figura 4.6 es también extraído de [1] y muestra la percepción del ambiente de navegación y su trayectoria, el cual permite apreciar el punto de inicio (señalado con un circulo) cabe señalar que el recorrido lo hace en sentido de las manecillas del reloj, y el primer archivo del banco de datos (Sensor_readings_24.data) pertenece a el recorrido siguiendo las reglas del algoritmo 4.1, además los datos son el resultado de 4 vueltas completas bajo el mismo esquema.

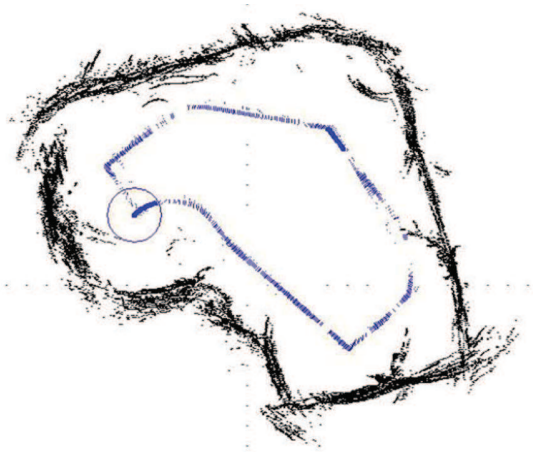


Figura 4.6: Esquema de la percepción del ambiente a partir de los sensores ultrasónicos y la trayectoria de SCITOS G5 con el Algoritmo IV.1.[1].

Ahora bien, ya que originalmente los datos son empleados para el problema de navegación se deben hacer algunos ajustes con las clases originales que son asignadas a cada uno de los datos, ya que el objetivo no es determinar cual es la respuesta del robot mientras este explora el ambiente de navegación, el principal interés radica en que el robot sea capaz de determinar su localización a partir de la experiencia (datos previos).

4.1.2. Paso 2. Organización y descripción del banco seleccionado

Durante este paso es importante considerar algunas modificaciones al banco de datos original, esto se debe a que existen algunas diferencias entre el problema de la navegación y la localización; por que se debe recordar que el banco de datos original fue empleado con fines de navegación del robot móvil, entonces es necesario determinar algunos aspectos que se deben adaptar tales como la asignación de tipos de clases y determinación de números de patrones que pertenecen a cada una de las clases definidas, y finalmente determinar cuales son los rasgos de cada patrón que son de mayor utilidad en el problema de localización del robot móvil.

4.1.2.1. Adaptación de clases para el problema de localización

La principal diferencia entre la navegación y la localización de un robot es que en la navegación, la información que proviene de los sensores es empleada generalmente para determinar las acciones necesarias para el eficiente desplazamiento del robot ante ciertas circunstancias o metas definidas, mientras que en la localización los datos de los sensores son utilizados como un recurso de referencia para conocer la posición del robot, es decir, en la localización la información de los sensores deben dar indicios, que permitan identificar la ubicación del robot en un determinado instante.

Datos por Vuelta. Para la adaptación de clase fue necesario en primer lugar determinar del conjunto de datos, cuales de ellos pertenecían a cada vuelta, recuérdese que el banco de datos ésta integrado por las lecturas de los 24 sensores disponibles bajo una tasa de 9 muestras por segundo y el recorrido consta de un total de 4 vueltas completas.

De forma que se realizó un análisis para comprobar la coincidencia del patrón de secuencias de las acciones del robot durante cada una de las vueltas considerando el algoritmo 4.1, y como consecuencia se han relacionado las 4 vueltas, la toma de mediciones por cada sensor y las acciones llevadas acabo por el robot en el escenario de la figura 4.5. Después se destinó un color para cada una de las cuatro acciones que el robot realizó en el algoritmo, durante su navegación, tal como se muestra en la siguiente tabla (véase la tabla 4.1).

De modo que cada uno de los 5,456 patrones del banco de datos aparece como se muestra a continuación.

Tabla 4.1: Asignación de colores dependiendo de las acciones de navegación del robot.

Naranja	Desacelerar y dar vuelta a la izquierda (giro suave a la izquierda).
Verde	Seguir adelante (avanzar).
Azul	Desacelerar y dar vuelta a la derecha (giro suave a la derecha).
Rosa	Parar y dar vuelta a la derecha.

N.Patro	Sensor 180° front	Sensor r -165°	Sensor r -150°	Sensor r -135°	Sensor r -120°	Sensor r -105°	Sensor r -90°	Sensor r -75°	Sensor r -60°	Sensor r -45°	Sensor r -30°	Sensor r -15°	Sensor r 0° Trasero	Sensor r 15°	Sensor r 30°	Sensor r 45°	Sensor r 60°	Sensor r 75°	Sensor r 90°	Sensor r 105°	Sensor r 120°	Sensor r 135°	Sensor r 150°	Sensor r 165°	Accion
1	0.438	0.5	3.63	3.65	5	2.32	5	2.35	2.33	2.64	1.7	1.69	1.7	1.72	1.74	0.59	0.5	0.49	0.5	0.45	0.43	0.44	0.44	0.43	Vuelta Suave a la DERECha
2	0.438	0.5	3.63	3.65	5	2.32	5	2.64	2.33	2.65	1.7	1.69	1.7	1.72	1.74	0.59	0.5	0.49	0.5	0.45	0.43	0.44	0.44	0.43	Vuelta Suave a la DERECha
3	0.438	0.5	3.63	3.63	5	2.32	5	2.64	2.33	2.64	1.7	1.69	1.7	1.72	1.74	0.59	0.5	0.49	0.5	0.45	0.43	0.44	0.45	0.43	Vuelta Suave a la DERECha
4	0.437	0.5	3.63	3.63	5	2.32	5	2.35	2.33	2.64	1.73	1.69	1.7	1.72	1.74	0.59	0.5	0.49	0.5	0.45	0.43	0.44	0.44	0.43	Vuelta Suave a la DERECha
5	0.438	0.5	3.63	3.63	5	2.32	5	2.64	2.33	2.64	1.7	1.69	1.7	1.72	1.74	0.59	0.5	0.49	0.5	0.45	0.43	0.44	0.44	0.43	Vuelta Suave a la DERECha
6	0.439	0.5	3.63	3.63	5	2.32	5	2.63	2.33	2.65	1.71	1.69	1.69	1.72	1.74	0.59	0.5	0.49	0.5	0.45	0.43	0.44	0.44	0.43	Vuelta Suave a la DERECha
7	0.44	5	3.63	3.63	5	2.32	3.03	2.35	2.33	2.64	1.73	1.68	1.69	1.71	1.75	0.59	0.5	0.49	0.5	0.45	0.43	0.45	0.44	0.43	Vuelta Suave a la DERECha
8	0.444	5.02	3.63	3.63	5	2.32	5	2.63	2.33	2.64	1.7	1.68	1.69	1.71	1.74	0.6	0.5	0.49	0.5	0.45	0.44	0.45	0.44	0.44	Vuelta Suave a la DERECha
9	0.451	5.03	3.64	3.64	5	2.32	3.03	2.62	2.32	2.63	1.71	1.67	1.68	1.7	0.75	0.6	0.5	0.49	0.5	0.46	0.44	0.45	0.45	0.44	Vuelta a la derecha
10	0.458	5.02	3.64	3.64	5	2.32	5	2.35	2.32	2.63	1.69	1.67	1.67	1.7	0.74	0.59	0.5	0.49	0.5	0.46	0.44	0.46	0.46	0.45	Vuelta a la derecha
11	0.465	0.53	3.65	3.67	5	2.32	5	2.61	2.32	2.63	1.67	1.66	1.67	1.69	0.74	0.59	0.5	0.49	0.5	0.47	0.45	0.46	0.47	0.46	Vuelta a la derecha
12	0.473	0.53	3.65	3.68	5	2.33	5	2.61	2.31	2.62	1.67	1.65	1.66	1.68	0.73	0.58	0.5	0.49	0.5	0.47	0.45	0.47	0.48	0.47	Vuelta a la derecha
13	0.481	0.54	3.66	3.68	5	2.33	5	2.61	2.3	2.62	1.67	1.64	1.65	1.28	0.72	0.58	0.5	0.49	0.5	0.48	0.46	0.47	0.48	0.47	Vuelta a la derecha
14	0.484	0.54	3.66	3.67	5	2.33	5	2.32	2.3	5.02	1.65	1.64	1.65	1.27	0.76	0.62	0.49	0.48	0.5	0.48	0.46	0.47	0.49	0.48	Vuelta a la derecha
15	0.484	0.53	3.67	3.66	2.35	2.33	5	2.33	2.31	2.62	1.65	1.64	1.65	1.27	0.76	0.53	0.49	0.48	0.49	0.51	0.46	0.47	0.49	0.48	Vuelta a la derecha
16	0.482	0.52	3.69	3.66	2.35	2.33	5	2.98	2.31	2.33	1.65	1.64	1.64	1.27	0.76	0.53	0.49	0.48	0.49	0.51	0.46	0.47	0.49	0.47	Vuelta a la derecha
17	0.481	0.52	3.68	3.66	2.36	2.33	2.36	2.98	1.7	2.62	1.65	1.64	1.65	1.27	0.76	0.53	0.49	0.48	0.49	0.51	0.46	0.49	0.48	0.47	Vuelta a la derecha
18	0.48	0.53	3.69	3.66	2.35	2.33	2.35	3	1.7	2.62	1.65	1.64	1.65	1.27	0.76	0.53	0.49	0.48	0.49	0.52	0.46	0.49	0.48	0.47	Vuelta a la derecha
19	0.481	0.52	5.02	3.66	2.36	2.33	2.35	2.99	1.7	2.62	1.65	1.64	1.65	1.27	0.77	0.59	0.49	0.48	0.5	0.53	0.46	0.5	0.48	0.47	Vuelta a la derecha
20	0.479	5	3.7	3.66	2.35	2.33	2.34	2.99	1.7	2.62	1.65	1.64	1.65	1.27	0.81	0.59	0.49	0.48	0.49	0.51	0.46	0.48	0.48	0.47	Vuelta a la derecha
21	0.48	0.51	5.02	3.67	2.34	2.33	2.35	2.98	1.7	2.62	1.65	1.64	1.65	1.27	0.81	0.64	0.49	0.48	0.49	0.52	0.46	0.47	0.48	0.47	Vuelta a la derecha
22	0.479	5	5.02	3.66	2.34	2.33	2.34	3	1.7	2.31	1.65	1.64	1.64	1.27	0.81	0.62	0.49	0.48	0.49	0.58	0.46	0.47	0.48	0.47	Vuelta a la derecha
23	0.479	5	5.02	3.66	2.35	2.33	2.34	3	1.7	2.31	1.65	1.64	1.64	1.27	0.31	0.62	0.49	0.48	0.49	0.5	0.46	0.46	0.48	0.47	Vuelta Suave a la DERECha
24	0.481	0.51	5.02	3.66	2.35	2.34	2.35	2.97	1.71	2.31	1.65	1.64	1.64	1.27	1.69	0.68	0.49	0.48	0.48	0.5	0.47	0.46	0.48	0.47	Vuelta Suave a la DERECha
25	0.486	0.51	0.52	3.66	2.36	2.34	2.34	2.63	1.71	2.31	1.66	1.64	1.64	1.67	1.7	0.7	0.52	0.48	0.48	0.55	0.47	0.46	0.47	0.47	Vuelta Suave a la DERECha
26	0.479	0.52	0.52	3.66	2.35	2.34	2.34	2.96	1.71	2.31	1.66	1.64	1.64	1.66	1.76	0.69	0.53	0.48	0.48	0.49	0.46	0.46	0.47	0.47	Vuelta Suave a la DERECha
27	0.48	0.55	0.52	3.66	2.35	2.34	2.34	2.63	1.72	2.31	1.66	1.64	1.64	1.66	1.76	0.69	0.55	0.48	0.47	0.56	0.46	0.45	0.47	0.48	Vuelta Suave a la DERECha
28	0.481	0.51	5.02	3.67	2.35	2.34	2.34	2.63	1.71	2.31	1.66	1.64	1.64	1.66	1.76	0.69	0.55	0.48	0.48	0.5	0.46	0.46	0.47	0.48	Vuelta Suave a la DERECha
29	0.482	0.53	0.52	3.67	2.35	2.34	2.34	2.63	1.71	2.31	1.66	1.64	1.64	1.65	1.76	0.57	0.55	0.48	0.48	0.51	0.46	0.46	0.47	0.48	Vuelta Suave a la DERECha
30	0.482	0.53	0.52	3.67	2.36	2.34	2.34	2.63	1.71	2.31	1.66	1.64	1.64	1.65	1.76	0.57	0.56	0.48	0.48	0.5	0.46	0.46	0.47	0.48	Vuelta Suave a la DERECha
31	0.482	0.53	0.52	3.67	2.35	2.34	2.34	2.63	1.71	2.31	1.66	1.64	1.64	1.65	1.76	0.56	0.54	0.48	0.48	0.49	0.46	0.46	0.47	0.48	Vuelta Suave a la DERECha
32	0.482	0.51	0.52	3.67	2.35	2.34	2.34	2.63	1.71	2.31	1.66	1.64	1.64	1.65	1.76	0.56	0.55	0.48	0.48	0.49	0.46	0.46	0.47	0.48	Vuelta Suave a la DERECha
33	0.484	0.51	0.53	3.67	2.35	2.34	2.34	2.96	1.71	2.31	1.66	1.64	1.63	1.65	1.75	0.68	0.54	0.48	0.48	0.54	0.47	0.46	0.47	0.48	Vuelta Suave a la DERECha
34	0.487	0.53	0.53	3.67	2.36	2.34	2.34	2.96	1.71	2.31	1.65	1.64	1.63	1.65	1.75	0.54	0.54	0.47	0.47	0.53	0.47	0.46	0.47	0.48	Vuelta Suave a la DERECha
35	0.492	0.54	0.53	3.67	2.36	2.34	2.34	3.47	1.7	2.3	1.65	1.63	1.62	1.65	1.74	0.53	0.55	0.47	0.47	0.49	0.47	0.46	0.47	0.49	Vuelta Suave a la DERECha
36	0.494	0.54	0.54	3.68	2.36	2.34	2.34	3.05	1.7	2.3	1.65	1.62	1.62	1.64	1.78	0.53	0.53	0.47	0.47	0.54	0.48	0.47	0.48	0.5	Vuelta Suave a la DERECha
37	0.501	0.54	5.02	3.68	2.36	2.34	2.34	3.74	2.6	2.29	1.64	1.62	1.61	1.64	1.73	0.52	0.54	0.47	0.47	0.5	0.48	0.47	0.48	0.51	Vuelta Suave a la DERECha
38	0.509	0.55	5.03	3.69	5	2.34	2.34	2.96	2.59	2.29	1.63	1.61	1.61	1.62	1.66	0.53	0.54	0.48	0.47	0.49	0.49	0.48	0.49	0.51	Vuelta Suave a la DERECha
39	0.517	0.55	5.03	3.7	2.37	2.35	2.34	2.99	1.68	2.28	1.63	1.6	1.6	1.61	1.65	0.52	0.55	0.49	0.47	0.48	0.49	0.48	0.49	0.53	Vuelta Suave a la DERECha
40	0.526	0.56	5.03	3.7	5	2.35	2.34	2.6	1.67	2.27	1.63	1.6	1.59	1.6	0.56	0.71	0.55	0.48	0.47	0.48	0.5	0.49	0.5	0.53	Vuelta a la derecha
41	0.535	0.57	0.67	3.71	5	2.35	2.34	2.59	1.66	2.27	5	1.59	1.58	1.59	1.62	0.7	0.54	0.48	0.47	0.48	0.5	0.5	0.51	0.54	Vuelta Suave a la DERECha
42	0.544	0.56	5.03	3.71	5	2.35	2.34	2.58	1.65	1.65	5	1.58	1.57	1.58	1.61	0.53	0.57	0.48	0.47	0.48	0.51	0.5	0.51	0.55	Vuelta Suave a la DERECha
43	0.553	0.57	5.03	3.72	5	2.35	2.34	2.58	1.65	1.64	5	1.57	1.56	1.57	1.6	0.53	0.59	0.48	0.47	0.48	0.52	0.51	0.52	0.57	Vuelta Suave a la DERECha
44	0.562	0.57	5.03	3.73	3.76	2.35	2.34	1.66	1.64	1.63	5	1.56	1.55	1.56	1.59	0.7	0.56	0.48	0.47	0.48	0.53	0.52	0.53	0.57	Vuelta Suave a la DERECha
45	0.572	0.58	5.02	3.73	3.78	2.35	2.34	2.36	1.63	1.63	5	1.55	1.54	1.55	1.58	0.69	0.58	0.48	0.47	0.48	0.54	0.53	0.53	0.58	Vuelta Suave a la DERECha
46	0.582	0.59	5.03	3.74	3.79	2.36	2.34	2.35	2.54	2.23	1.56	1.54	1.53	1.54	1.17	0.74	0.57	0.48	0.47	0.48	0.55	0.54	0.55	0.59	Vuelta Suave a la DERECha
47	0.591	0.6	5	3	3.79	2.36	2.34	2.36	2.54	2.22	1.55	1.53	1.52	1.53	1.16	0.73	0.53	0.48	0.47	0.48	0.56	0.54	0.55	0.6	Vuelta Suave a la DERECha
48	0.6	0.61	5.03	3	3.77	2.37	2.34	2.35	2.53	2.22	5	1.52	1.51	1.52	1.15	0.73	0.57	0.48	0.47	0.48	0.58	0.55	0.58	0.61	Vuelta Suave a la DERECha
49	0.604	0.62	5.03	3	3.77	5	2.34	2.35	2.53	2.21	1.53	1.51	1.5	1.51	1.14	0.82	0.56	0.48	0.47	0.48	0.6	0.56	0.58	0.62	Vuelta Suave a la DERECha
50	0.601	0.63	5	3	3.77	5	2.34	2.35	2.52	2.2	1.52	1.5	1.5	1.5	1.14	1.62	0.58	0.48	0.47	0.48	0.61	0.57	0.58	0.63	Vuelta Suave a la DERECha
51	0.617	0.63	5	3.77	3.77	5	2.34	2.36	2.52	1.57	1.52	1.49	1.49	1.49	1.13	1.62	0.58	0.48	0.47	0.48	0.51	0.57	0.59	0.64	Vuelta Suave a la DERECha
52	0.623	0.64	5	3.78	3.77	5	2.34	3.67	2.51	1.56	1.51	1.48	1.48	1.48	1.59	1.61	0.58	0.48	0.47	0.48	5	0.58	0.59	0.65	Vuelta Suave a la DERECha

Figura 4.7: Banco de datos "Wall-Following Robot Navigation" coloreado en correspondencia con la tabla 4.1.

Una vez que se tiene la asignación de colores, se establece la siguiente premisa "*El robot debe realizar acciones similares bajo muestras de mediciones tomadas en un mismo espacio*", dicho en otras palabras, cada que el robot pase por un mismo punto, se debe suponer que sus acciones deben ser similares cada vez que pase por el mismo lugar. Lo cual se confirmó con una inspección a simple vista, como se muestra en la figura 4.7. Algunas observaciones de importancia son el hecho de que se han dividido las instancias del banco de datos original en las 4 vueltas que el robot realiza, otra consideración fue

el hecho de establecer el punto de inicio de navegación justo cuando el robot gira hacia la izquierda en la única vuelta prolongada que hace durante su recorrido (el punto de inicio se encuentra señalado con un circulo en la figura 4.6), esto con el propósito de tener un punto de referencia.

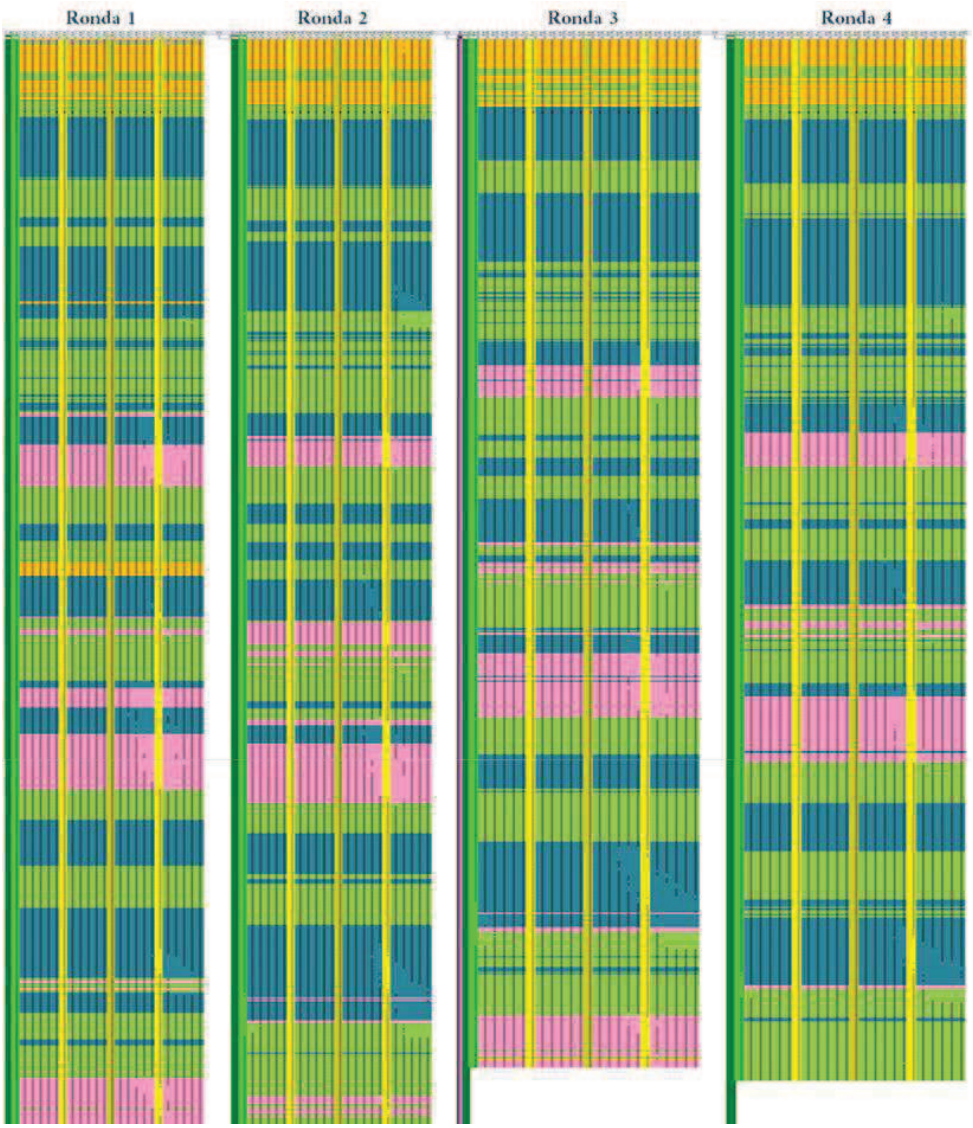


Figura 4.8: Banco de datos “Wall-Following Robot Navigation” segmentado entre las 4 rondas del robot en el escenario de la figura 4.5.

Ahora bien, dentro de la secuencia de las cuatro rondas de la figura 4.8 se pueden observar ciertas similitudes, las cuales permiten realizar una correspondencia con el espacio físico que el robot recorre, si se observa la figura 4.9, la cual es una combinación de las figuras 4.5 y 4.6 del artículo [1]. Dentro de la figura también se pueden observar claramente 7 secciones que han sido numeradas, con ello se pretende correlacionar las secciones con la toma de lecturas del robot durante las 4 rondas.

La figura 4.9 también se encuentra correlacionada con las acciones del robot que se muestran en la tabla 4.1, ya que el color de las flechas en el recorrido del robot esta relacionado con los colores de la tabla.

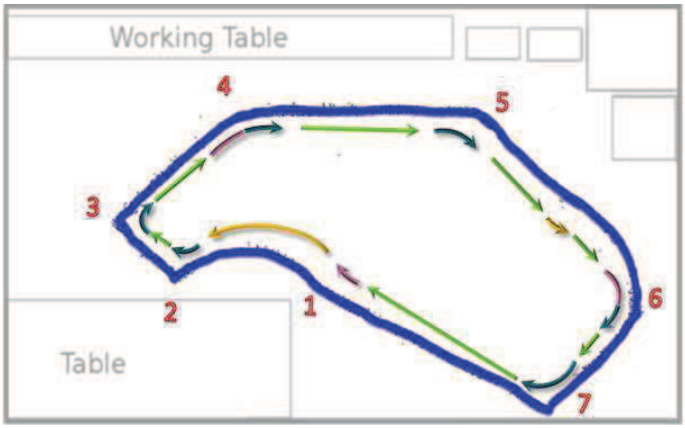


Figura 4.9: Esquema producto de la unión del escenario mostrado en la figura 4.5 y la trayectoria del robot mostrado en la figura 4.6.

4.1.2.2. Determinación de patrones

Ahora bien, gracias a que se tienen 4 rondas de navegación del robot móvil bajo el mismo escenario se ha decidido utilizar solo una de las rondas, esto con el propósito de no saturar el banco de datos con información duplicada de una misma sección. Como se muestra en la figura 4.10 se ha empleado la primera ronda que el robot G5 SCITOS ha realizado en el escenario mostrado en la figura 4.5, los patrones que integran esta ronda van desde el 763 hasta el 1,675.

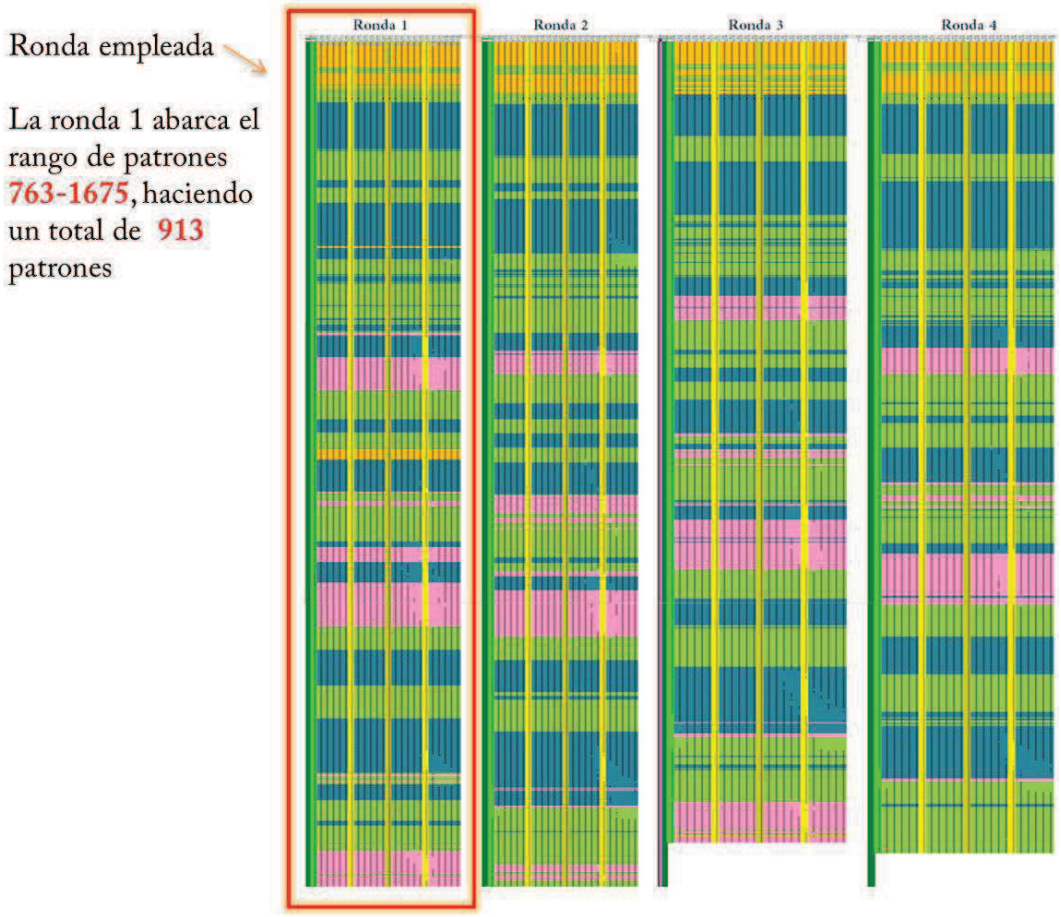


Figura 4.10: Extracto del banco de datos "Wall-Following Robot Navigation" en donde se muestra en el recuadro rojo la ronda seleccionada para la fase de aprendizaje y prueba de la memoria asociativa Alfa-Beta.

4.1.2.3. Determinación de rasgos

Cabe mencionar que durante esta fase de experimentación se tomó en cuenta un filtro al banco de datos (recuérdese que el banco de datos originalmente está integrado por las lecturas de los 24 sensores disponibles bajo una tasa de 9 muestras por segundo). Para el filtro se realizaron los cálculos de la media entre las lecturas de los sensores, de modo que de las muestras de los 24 sensores disponibles solo se seleccionan un conjunto de ellos que son de relevancia significativa durante la navegación del robot.

Entonces dentro de las mediciones del robot que son de relevancia se encuentran 4 sensores principalmente: ***Sensor Frontal*** (180°), ***Sensor Izquierdo*** (-90°), ***Sensor Posterior*** (0°) y ***Sensor Derecho*** (90°); bajo esta premisa se puede ajustar el error que alguno de estos sensores pudieran tener durante su recorrido, para el caso particular de este trabajo de tesis se ha decidido ampliar el espectro de medición de los sensores principales agrupando los dos sensores contiguos de cada lado de sensores principales y posteriormente extraer la media, tal como se muestra en la figura 4.11.

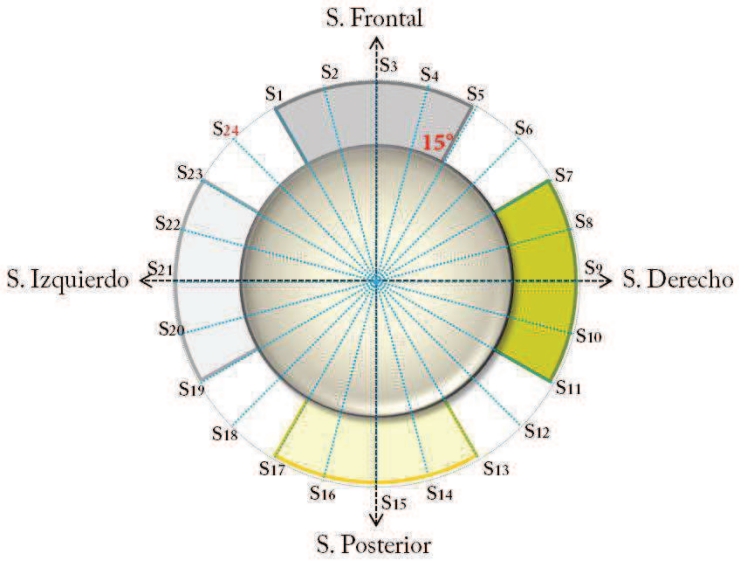


Figura 4.11: Esquema de distribución de los sensores del robot, en donde el sector sombreados representa la agrupación de sensores relacionados con sus respectivos sensores principales (Sensor Frontal (180°), Sensor Izquierdo (-90°), Sensor Posterior (0°), Sensor Derecho (90°)).

Como puede observarse en la figura 4.11 se han agrupado los sensores contiguos a los sensores principales con la finalidad de contar con un muestreo de 4 grupos que constan del arco de 5 sensores de cada grupo, considerando así, 30° grados hacia la derecha y 30° hacia la izquierda de cada sensor principal haciendo un total de 60°, tal como se muestra en la siguiente tabla. Ahora bien, es necesario realizar los cálculos que corresponden a las medias de los sensores, tal como se muestra en la tabla 4.2.

Tabla 4.2: Asignación de sensores en los cuatro grupos para calcular la media de cada grupo.

Grupo I	Media de los sensores contiguos a el Sensor Frontal (S1, S2, S3, S4, S5).
Grupo II	Media de los sensores contiguos a el Sensor Derecho (S7, S8, S9, S10, S11).
Grupo III	Media de los sensores contiguos a el Sensor Posterior (S13, S14, S15, S16, S17).
Grupo IV	Media de los sensores contiguos a el Sensor Izquierdo (S19, S20, S21, S22, S23).

Entonces se obtiene el valor de media de los rasgos que pertenecen a cada grupo (los cuatro grupos de la tabla 4.2), esta operación se repite para cada patrón; en la figura 4.12se puede observar un recuadro marcado en color rojo que presenta un extracto de los valores obtenidos.

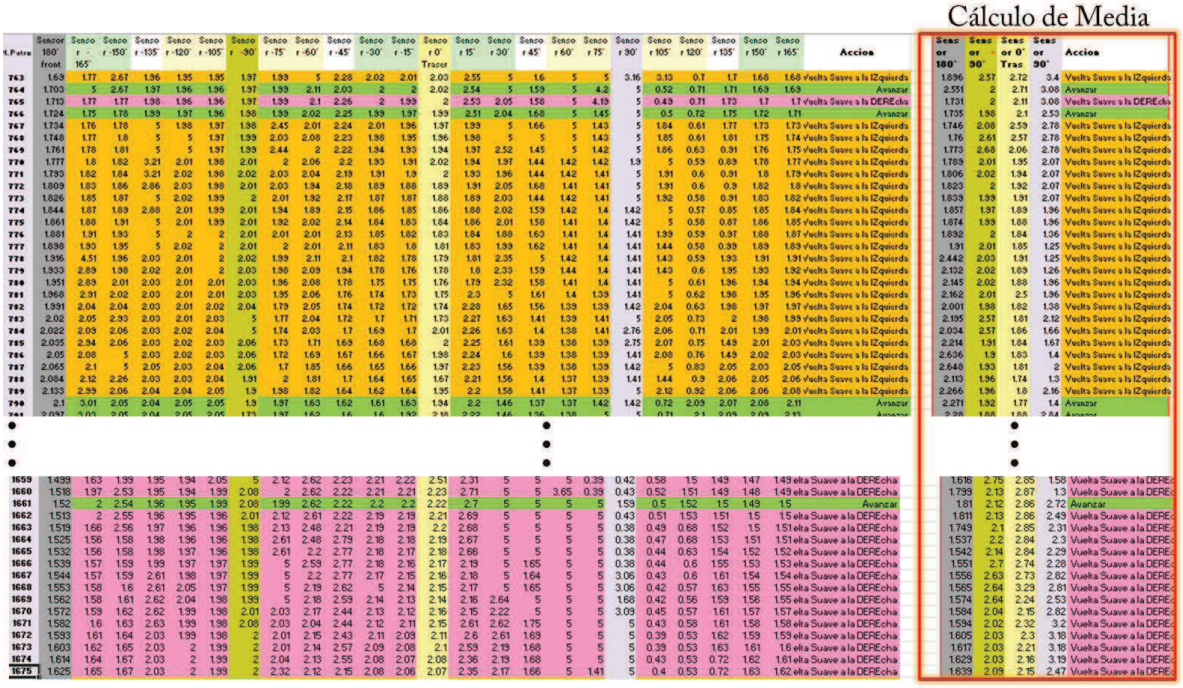


Figura 4.12: Extracto del Banco de datos “Wall-Following Robot Navigation” en donde se muestra en el recuadro rojo la media de cada patrón dividido en cuatro grupos.

4.1.2.4. Determinación de clases

Por conveniencia se ha decidido dividir el escenario en partes (cada una representa una clase), de modo que haciendo una relación de las acciones del robot durante la navegación con el escenario se obtienen las clases que formarán los lugares que la memorias asociativa Alfa-Beta tiene que ser capaz de reconocer (véase la figura 4.13).

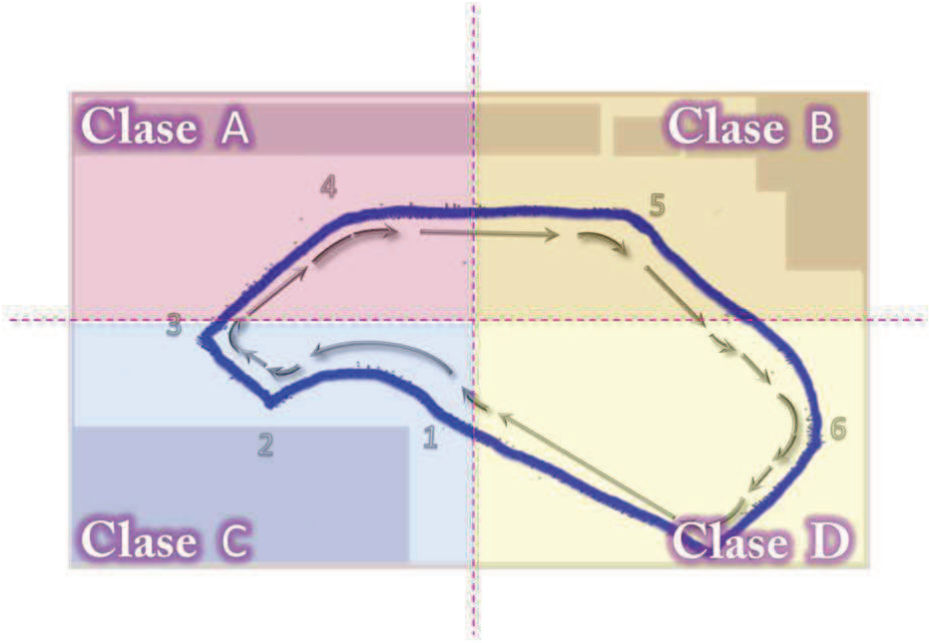


Figura 4.13: Escenario de navegación del robot señalando las cuatro clases (Clase A, Clase B, Clase C y Clase D) que desean que la memoria asociativa Alfa-Beta sea capaz de reconocer.

Entonces si ya se han decidido las clases se debe determinar los patrones que pertenecen a cada clase, así que en la tabla 4.3 se ha determinado esta relación. Se debe recordar que el total de patrones utilizados para este trabajo de investigación van desde el patrón 763 hasta el 1675, tal como se muestra en la figura 4.10.

Tabla 4.3: Asignación de patrones a cada clase.

Clase A	Integrada por los patrones 763-993 y 1636-1675	231+40=271
Clase B	Integrada por los patrones 994-1161	168
Clase C	Integrada por los patrones 1162-1278	117
Clase D	Integrada por los patrones 1279-1635	357
4 Clases	**Desde el patrón 763 hasta el 1675**	**913**

Resumen
Hasta este punto se cuenta con un banco de datos filtrado por medio del cálculo de la medias obtenidas a partir de la formación de cuatro grupos formados por 5 sensores, los cuales equivalen a un arco de 60°, considerando como punto medio los sensores principales (Sensor Frontal (180°), Sensor Izquierdo (-90°), Sensor Posterior (0°) y Sensor Derecho (90°)), de esta forma se debe determinar que las cuatro medias calculadas forman los **«rasgos»** de cada patrón, aclarando que el **«patrón»** esta formado por los cuatro rasgos que integran una fila. Además se cuenta con la relación de los datos y el espacio físico que representan dentro del escenario de navegación, es decir, se han determinado las secciones (véase la figura 4.13) que la memoria asociativa debe ser capaz de aprender y clasificar, por lo tanto

las áreas representadas por las letras ***A, B, C y D*** conforman las nuestras **«clases»**; mientras que los patrones que pertenecen a cada clase pueden observarse en la tabla 4.3. De modo que se cuenta con la información necesaria para el aprendizaje de la memoria asociativa Alfa-Beta, así que se puede comenzar con el diseño de la memoria para probarla con el banco que datos que ya hemos filtrado.

4.1.3. Paso 3. Diseño e implementación de la memoria asociativa Alfa-Beta.

En la presente tesis se hace uso de las memorias asociativas Alfa-Beta, las cuales fueron descritas en la sección 3.1, de modo que el paso tres tienen como propósito el diseñar e implementar un algoritmo basado en los modelos matemáticos de las memorias previamente descritos, el objetivo es determinar la asertividad de las memorias asociativas Alfa-Beta cuando son empleadas en el problema de localización de robots móviles. Las memorias asociativas Alfa-Beta presentan dos fases generales, la primera fase esta encargada del aprendizaje de la memoria asociativa, por lo que hace uso del conjunto fundamental o de entrenamiento, cabe mencionar que durante la fase de experimentación se utilizó un método llamado K-fold Cross Validation, el cual nos permite separar al conjunto de entrenamiento o fundamental del conjunto de prueba.

Existen diferentes métodos de validación como son el método H (*Holdout*), validación cruzada (*K- fold cross-validation*), LOO (*Leave One Out*), entre otros. A continuación se presenta una breve descripción de los tres métodos de validación mencionados.

Holdout. El método holdout (método H) divide aleatoriamente el conjunto de datos en dos subconjuntos. El primer grupo se integra por 2/3 del total del conjunto de patrones y corresponde al denominado conjunto de entrenamiento, el cual es utilizado para que el clasificador aprenda, y el segundo grupo contiene el restante de los patrones es decir 1/3 y forma parte del grupo de prueba y con estos últimos se determina la asertividad del clasificador.

K-fold-cross-validation. En el método de k-particiones validación-cruzada (k-fold cross-validation) [59], funciona de la siguiente manera: la base de datos es particionada aleatoriamente en k subconjuntos, aproximadamente del mismo tamaño, en donde k-1 subconjuntos constituyen el conjunto de entrenamiento y el restante el conjunto de prueba.

Leave-One-Out. Es un caso especial del método de K-fold-cross-validation. En este método de validación la base de datos es particionada k veces, siendo k el número de casos originales n. Se forma un conjunto de entrenamiento con los n-1 casos dejando un caso fuera, el cual será utilizado en la clasificación. Cabe señalar que este método es el menos sesgado pero presenta una varianza alta, la cual en algunos casos presenta una estimación de la precisión del clasificador no deseada. Este tipo de validación, generalmente es aplicado a bases de datos con relativamente pocos casos.

Para el presente trabajo de tesis se ha empleado el método de validación K-fold-cross-validation con k=10.

De modo que el conjunto de entrenamiento será empleado en la primera fase de la memoria asociativa Alfa-Beta (fase de aprendizaje) y el conjunto de prueba será utilizado en la segunda fase de la memoria (fase de prueba). Ambas fases serán descritas a continuación.

Fase de Aprendizaje. El propósito de esta fase es el de aprender a partir de un conjunto de patrones verdaderos. En esta fase se utiliza un conjunto de patrones al cual se denomina conjunto fundamental o conjunto de entrenamiento, la determinación del tamaño de este conjunto suele ser producto de la utilización de algún método de validación.

Fase de Prueba. El objetivo es determinar la asertividad del clasificador (en este caso la memoria asociativa Alfa-Beta), de modo que se debe presentar al clasificador un patrón nuevo y la tarea del clasificador es la de determinar a que clase pertenece. Durante esta fase se suele hacer uso de un conjunto que no se le ha enseñado al clasificador el cual es llamado «conjunto de prueba», el tamaño de este conjunto suele determinarse por medio del método de validación.

Estas dos fases son parte del algoritmo de la memoria asociativa, no obstante existe otra fase que pocas veces es referenciada en los clasificadores, se le llama Fase de Recuperación.

Fase de Recuperación. La fase de recuperación tiene como propósito determinar si todos los patrones que se le enseñaron al clasificador durante la «Fase de Aprendizaje» fueron aprendidos correctamente. De modo que una vez que la memoria asociativa ha pasado la fase de aprendizaje se le muestran nuevamente todos y cada uno de los patrones que previamente ha aprendido, para determinar si aprendió correctamente. Parece lógico que un clasificador debe ser capaz de aprender correctamente antes de intentar clasificar un patrón nuevo, sin embargo muchos clasificadores no son capaces de recordar todos los patrones que aprendieron, y es justamente esa una de las ventajas de las memorias asociativas Alfa-Beta, ya que es capaz de recordar todo lo aprendido (100 % en la fase de recuperación) y también es capaz de clasificar eficientemente patrones nuevos que no han sido aprendidos.

4.1.3.1. Algoritmo de memoria autoasociativa Alfa-Beta tipo Max

Se ha propuesto emplear la memoria autoasociativa Alfa-Beta tipo Max, la cual es explicada en la sección 3.2.4, el algoritmo original ha sido adaptado al problema de localización de robots móviles empleando el banco de datos filtrado en el la sección 4.1.2.

Algoritmo 4.2 Fase de aprendizaje de la memoria autoasociativa Alfa-Beta tipo Max adaptada a la base de datos descrita en la sección 4.1.2.

Memoria Autoasociativa **Alfa – Beta** **Fase de Aprendizaje**

1. Obtener el conjunto de entrenamiento *(C.E.)* utilizando el método de validación seleccionado.
2. Convertir a código binario cada uno de los rasgos.
3. Concatenar todas las cadenas binarias de los rasgos. De modo que cada patrón esta representado por la concatenación de las cadenas binarias de todos los rasgos que forman parte de una sola fila. Entonces el número de patrones es igual al número de filas que pertenecen al conjunto de entrenamiento.
4. Para cada patrón $\mu = 1, 2, ..., n.imágenes\, del\, C.E.$, se construye la matriz.

$$\left[x^{\mu} \otimes (x^{\mu})^{t}\right]_{n\times n} \tag{4.1}$$

   Por lo tanto se construye una matriz por cada patrón, de modo que se tiene un número de matrices= al número de filas del C.E.

5. Se aplica el operador binario máximo $\bigvee$ a las matrices obtenidas en el paso 4 y se obtiene la matriz M, de tal forma que se crea la memoria autoasociativa $\alpha\beta$ tipo $\bigvee$

$$M = \bigvee_{\mu=1}^{p} \left[x^{\mu} \otimes (x^{\mu})^{t}\right] \tag{4.2}$$

   La entrada ij-ésima está dada por la siguiente expresión:

$$\lambda_{ij} = \bigvee_{\mu=1}^{p} \alpha(\, x_i^{\mu},\, x_j^{\mu}) \tag{4.3}$$

   y de acuerdo con que α: $A \times A \to B$, se tiene que $\lambda_{ij} \in B$, $i \in \{1, 2, \ldots, n\}$. $\forall j \in \{1, 2, \ldots, n\}$

Algoritmo 4.3 Fase de recuperación de la memoria autoasociativa Alfa-Beta tipo Max adaptada a la base de datos descrita en la sección 4.1.2.

Memoria Autoasociativa **Alfa − Beta** **Fase de Recuperación**

1. Se le presenta cada uno de los patrones aprendidos a la memoria autoasociativa $\alpha\beta$ tipo $\bigvee$ y se realiza la operación $\triangle_\beta$:

$$\bigwedge \triangle_\beta x^\omega. \tag{4.4}$$

   El resultado de la operación anterior será el vector columna de dimensión n.

$$(\bigwedge \triangle_\beta x^\omega) = \bigvee_{j=1}^{n} \beta(\lambda_{ij}, x_j^\mu) \tag{4.5}$$

$$(\bigwedge \triangle_\beta x^\omega) = \bigvee_{j=1}^{n} \beta \left\{ \left[\bigvee_{\mu=1}^{p} \alpha \left(x_i^\mu, x_j^\mu \right) \right], \, x_j^\omega \right\} \tag{4.6}$$

➢ **Primera Evaluación.** De este modo se determina la asertividad de recuperación de patrones de la memoria autoasociativa $\alpha\beta$ tipo $\bigvee$.

Algoritmo 4.4 Fase de prueba de la memoria autoasociativa Alfa-Beta tipo Max adaptada a la base de datos descrita en la sección 4.1.2.

Memoria Autoasociativa **Alfa − Beta** **Fase de Prueba**

1. Se le presenta cada uno de los patrones $\hat{x}$ que forman parte del conjunto de prueba (C.P.) a la memoria Autoasociativa $\alpha\beta$ tipo $\bigvee$ y se realiza la operación $\triangle_\beta$:

$$\bigwedge \nabla_\beta \hat{x} \tag{4.7}$$

Al igual que en la fase de recuperación el resultado es un vector columna de dimensión n expresada:

$$(\bigvee \triangle_\beta \hat{x}) = \bigwedge_{j=1}^{n} \beta(v_{ij}, \hat{x}) \tag{4.8}$$

$$(\bigvee \triangle_\beta \hat{x}) = \bigwedge_{j=1}^{n} \beta \left\{ \left[\bigvee_{\mu=1}^{p} \alpha \left(x_i^\mu, x_j^\mu \right) \right], \, \hat{x_j} \right\} \tag{4.9}$$

➢ **Segunda Evaluación.** De este modo se determina la asertividad de clasificación de patrones distintos a los aprendidos por la memoria Autoasociativa $\alpha\beta$ tipo $\bigvee$.

Ahora bien, el algoritmo ha sido programado en Visual Basic .NET (VB.NET) utilizando el entorno de desarrollo Microsoft Visual Studio, instalado en una notebook Asus G74SX bajo las siguientes características:

Intel® Core™ i7 720QM Procesador: 1.6 GHz con un turbo arriba de los 2.8 GHz.
Sistema Operativo: Windows® 7 Ultimate.

Memoria Principal: DDR3 1066/1333 MHz de 8GB (para plataforma Intel® Core™ i7).
Gráficos y Memoria de Vídeo: NVIDIA® GeForce® GTS 360M, 1GB GDDR5 VRAM.
Disco duro: 500GB 5400rpm/7200rpm.

Se diseñó e implementó una memoria asociativa Alfa-Beta Autoasociativa tipo Max (Véase el algoritmo 4.2), el banco de datos empleados durante el experimento corresponde a «Wall-Following Robot Navigation», el cual esta disponible en el repositorio de UCI[a]. Para la selección del conjunto de entrenamiento y de prueba se empleo el método K-fold-cross-validation con k=10. Los resultados obtenidos son:

- **Fase de Aprendizaje:** El conjunto de entrenamiento esta integrado por 822 patrones, con 4 clases que representan las secciones (véase la figura 4.13) en las que el robot debe ser capaz de ubicarse.
- **Fase de Recuperación**: 100 % de recuperación de patrones aprendidos, obteniendo **0 %** de factor de olvido (se empleó el conjunto de entrenamiento).
- **Fase de Prueba:** 95.20 % de asertividad considerando 91 patrones en el conjunto de prueba.

[a]El repositorio UCI el cual pertenece a la Universidad de California del Center for Machine Learning and Intelligent Systems. En donde se mantienen 211 bancos de datos disponibles, los cuales pueden ser usados por la comunidad interesada en la aplicación de aprendizaje automático. Consulte en http://archive.ics.uci.edu/ml/

4.1.4. Paso 4. Probar el banco de datos con otros métodos y comparar los resultados con los de la memoria Alfa-Beta

Una vez estructurado y definido el banco de datos se propone probarlo con otros modelos (redes bayesianas, vecino más cercano, arboles binarios, entre otros, véase la sección 3.1.1), dedicados en la tarea de reconocimiento de patrones, al igual que las memorias asociativas Alfa-Beta. El propósito es tener un panorama de cuales son los métodos que presentan un mayor índice de asertividad, y por lo tanto saber cuales son los porcentajes que se deben obtener con las memorias asociativas Alfa-Beta, para determinar que este tipo de memorias son un método competitivo en la solución del problema de localización de robots móviles. Por lo que en un primer experimento se utilizara una herramienta llamada Weka[60], la cual permitirá obtener resultados de otros métodos; Weka soporta los métodos de clasificación y predicción más citados en la literatura, por lo que Weka proporcionará resultados preliminares que indicaran de forma anticipada que tan asertivos se deben comportar los algoritmos de las memorias asociativas Alfa-Beta, considerando como objetivo el igualar o mejorar los resultados obtenidos por otros métodos.

Se debe considerar que el banco de datos empleado fue el mismo que en la memoria asociativa Alfa-Beta el cual se describe en la sección 4.1.2, además de que los métodos de clasificación que a continuación se presentan son una selección de los métodos que mostraron el mejor resultado en Weka, empleando el k-fold cross-validation con un k=10 como método de validación.

En la figura 4.14 se muestra la entrada y análisis que Weka realiza con los datos de la base filtrada, en donde se pueden ver las clases y el número de patrones.

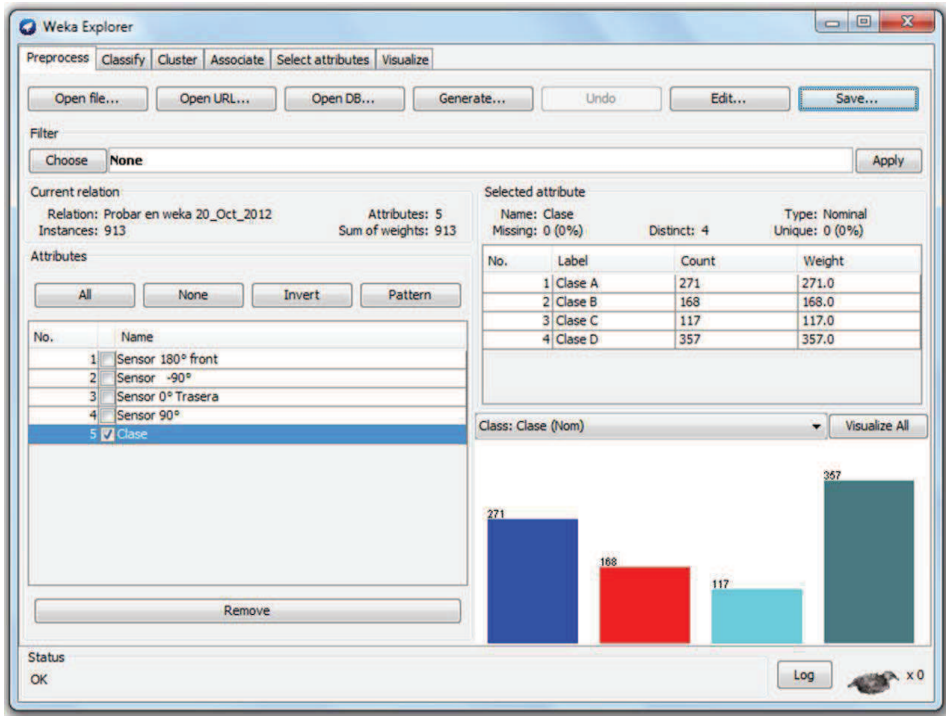


Figura 4.14: Imagen de pantalla de Weka donde se muestra el análisis del banco de datos.

Resultados obtenidos con los clasificadores de Weka Se han impreso las pantallas de los mejores resultados obtenidos por cada una de las siete familias de clasificadores integradas en Weka (véase la figura 4.15).

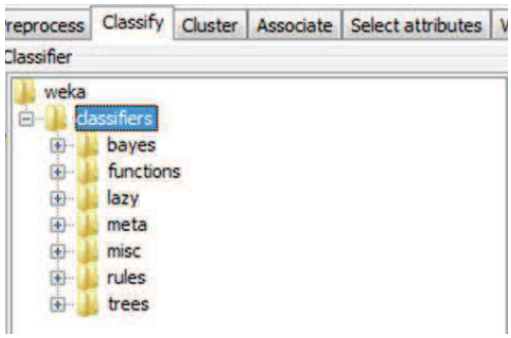


Figura 4.15: Familia de clasificadores contenidos en Weka.

➢ **Clasificador Network Classifier Bayes= 77.437 % de asertividad.**

Es un modelo probabilístico multivariado que relaciona un conjunto de variables aleatorias de modo que es capaz de modelar un fenómeno mediante un conjunto de variables y las relaciones de dependencia entre ellas.

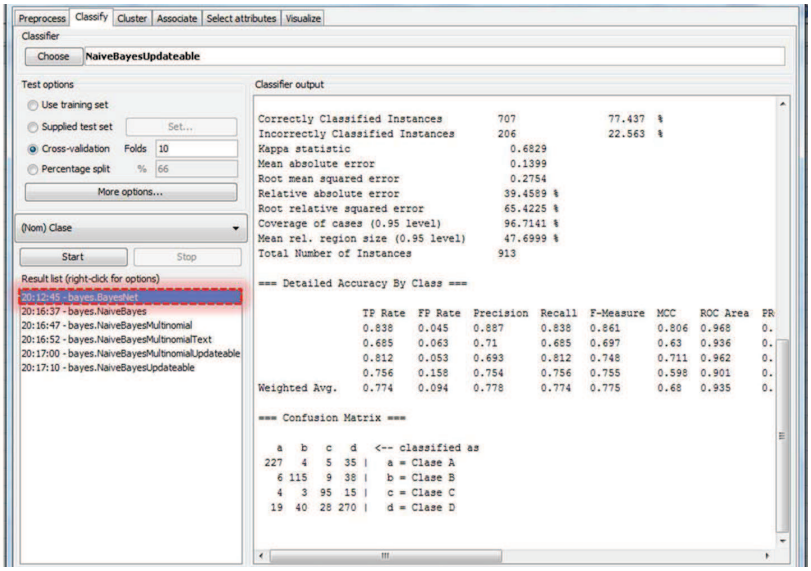


Figura 4.16: Imagen de pantalla de Weka en el procesamiento del banco de datos que ha sido filtrado con medias y clasificado bajo el método **«Network Classifier Bayes»** y k-fold cross validation con k=10.

➢ **Clasificador Multilayer Perceptron Functions= 72.289 % de asertividad.**

Es un modelo matemático inspirado en el comportamiento biológico de las neuronas y en la estructura del cerebro. Se basa en funciones de activación siendo una red neuronal artificial (RNA) formada por múltiples capas.

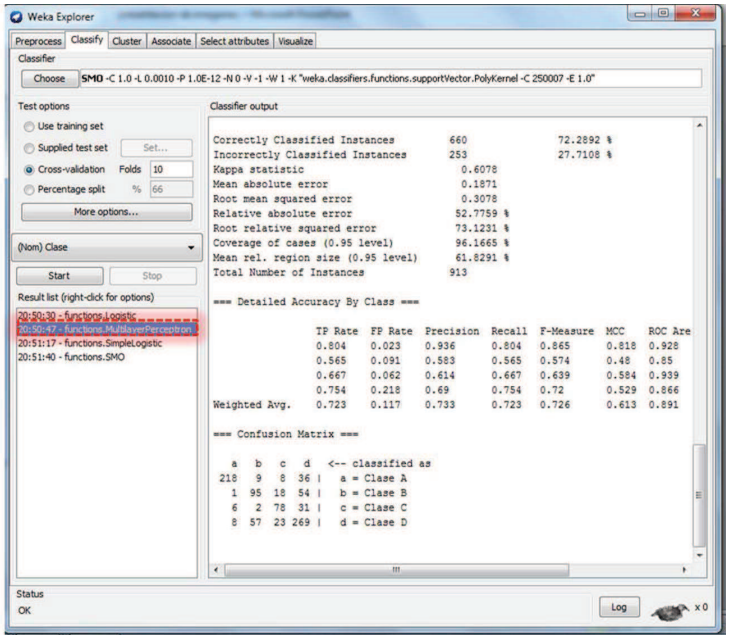


Figura 4.17: Imagen de pantalla de Weka en el procesamiento del banco de datos que has ido filtrado con medias y clasificado bajo el método «**Multilayer Perceptron Functions**» y k-fold cross validation con k-10.

➢ **Clasificador Lazy KStar =89.594 % de asertividad.**

Es un clasificador basado en instancias, determina las semejanzas entre las instancias por medio de una función de similitud. Se diferencia de otros modelos de instancias por que emplea una función de distancia basada en la entropía.

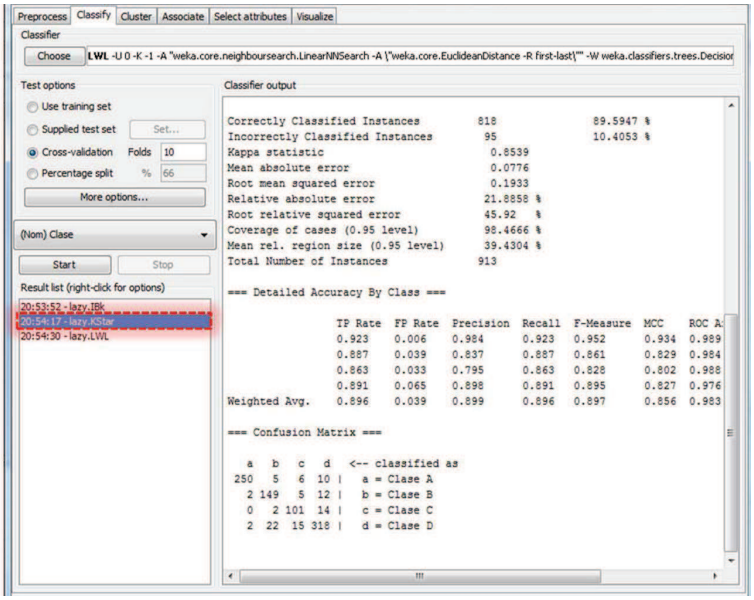


Figura 4.18: Imagen de pantalla de Weka en el procesamiento del banco de datos que ha sido filtrado con medias y clasificado bajo el método «**Lazy KStar**» y k-fold cross validation con k=10.

➢ **Clasificador Random Committee = 88.061 % de asertividad.**

Es un clasificador basado que crea un comité de clasificadores aleatorios.

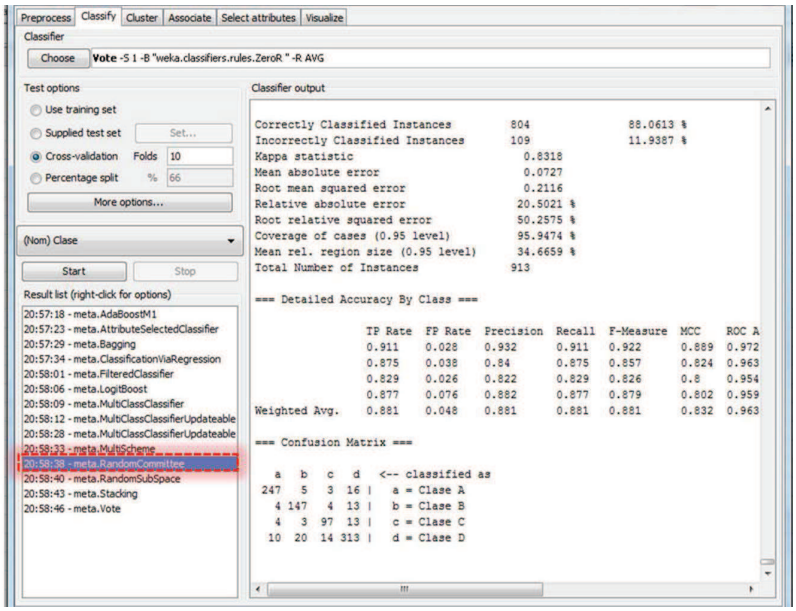


Figura 4.19: Imagen de pantalla de Weka en el procesamiento del banco de datos que ha sido filtrado con medias y clasificado bajo el método **«Random Committe »** y k-fold cross validation con k=10.

➢ **Clasificador Rules PART = 85.651 % de asertividad.**

Es un clasificador basado en reglas de decisión, genera clases por medio de una lista de decisiones PART. Empleando el lema "divide y vencerás", de modo que en cada integración se construye un modelo de árbol utilizando M5 seleccionando la mejor hoja según la regla.

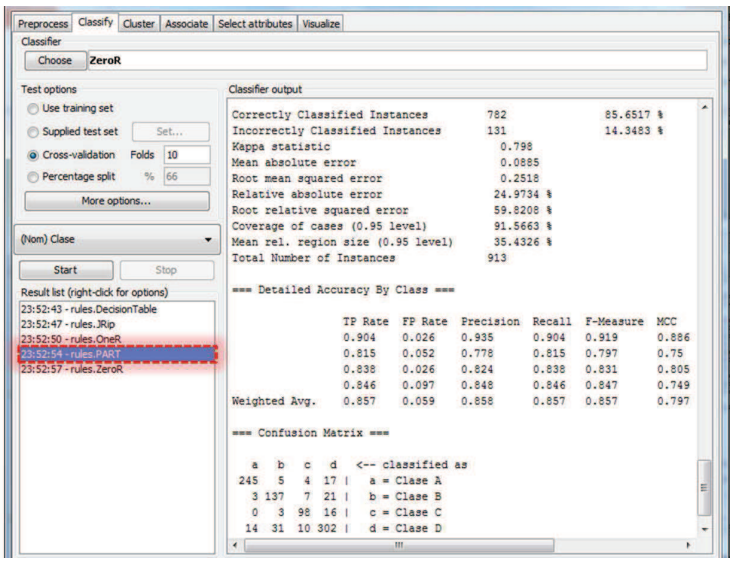


Figura 4.20: Imagen de pantalla de Weka en el procesamiento del banco de datos que ha sido filtrado con medias y clasificado bajo el método **«Rules Decision Table»** y k-fold cross validation con k=10.

➢ **Clasificador Trees Random Forest = 88.499 % de asertividad.**

Es un clasificador que crea un bosque de arboles aleatoriamente.

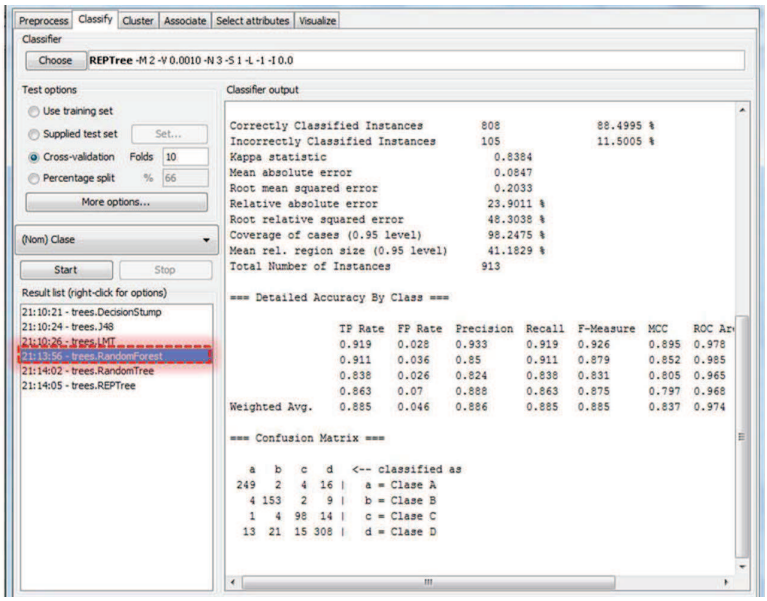


Figura 4.21: Imagen de pantalla de Weka en el procesamiento del banco de datos que ha sido filtrado con medias y clasificado bajo el método **«Trees Random Forest»** y k-fold cross validation con k=10.

La gráfica 4.22 muestra los valores de obtenidos con Weka con cada uno de los clasificadores agrupados por familias, en un circulo rojo se encuentran los valores más altos obtenidos por cada familia de clasificadores.

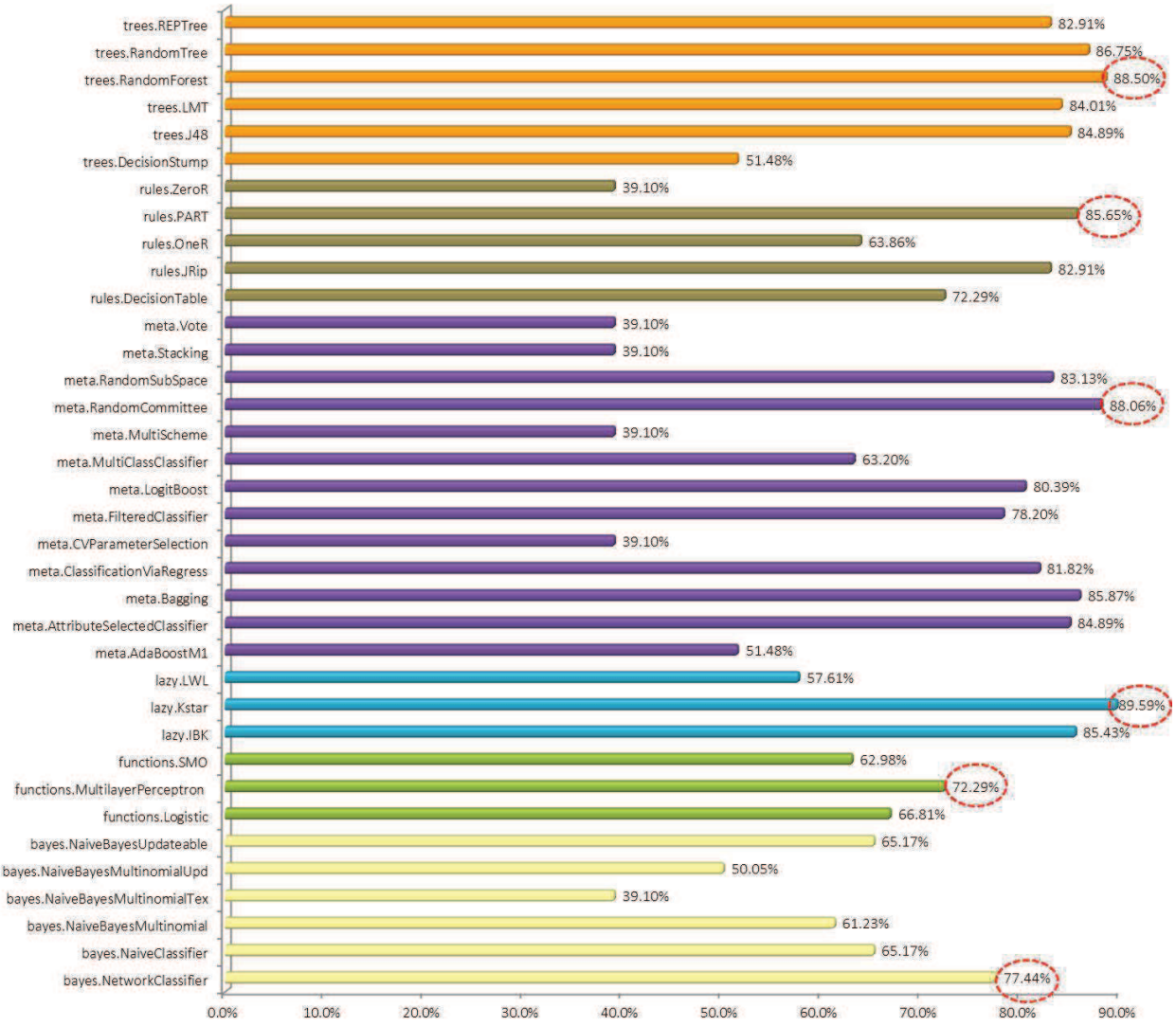


Figura 4.22: Gráfica que muestra los valores de asertividad obtenidos con cada uno de los clasificadores de Weka, los cuales están agrupados por familias.

Gráfica Comparativa

Una vez que se tienen los resultados tanto de la memoria asociativa Alfa-Beta como de los métodos de Weka que presentaron los mejores porcentajes de asertividad, es conveniente establecer un marco comparativo en el cual se perciba de forma gráfica el rendimiento de cada método por familia, tal como se muestra en la figura 4.23.

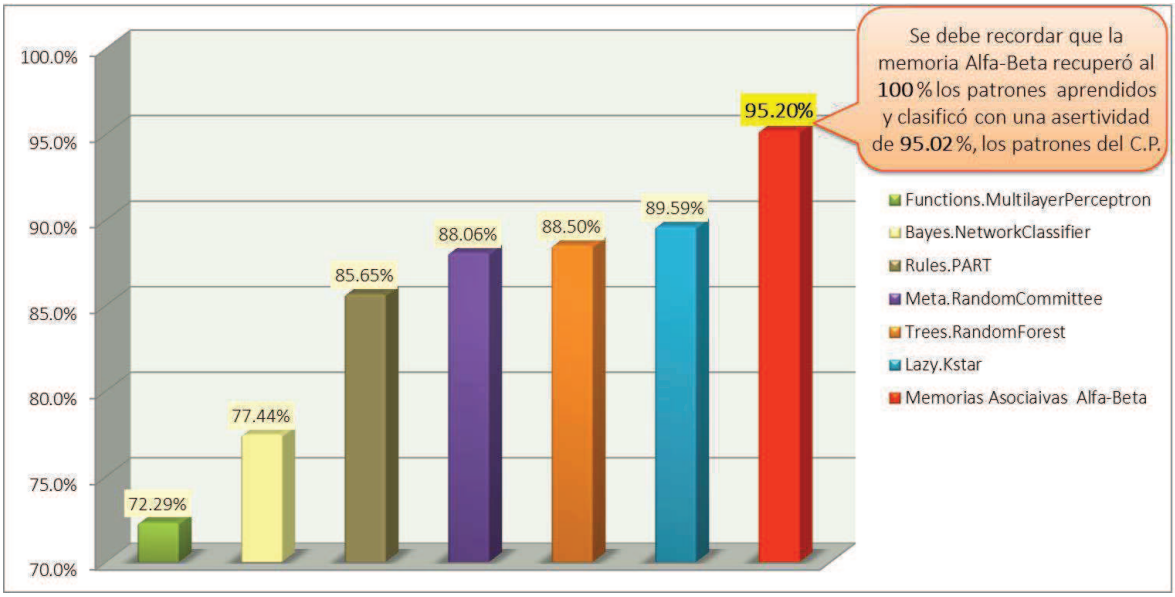


Figura 4.23: Gráfica comparativa que presenta el método con mayor porcentaje de asertividad de cada familia contenida en Weka, en el que se incluye el resultado obtenido en el presente trabajo de investigación empleando las memorias asociativas Alfa-Beta.

4.2. Experimento n.2. Utilizando simulador

Como ya se mencionó en la introducción del presente capítulo, uno de los propósitos es implementar las memorias asociativas Alfa-Beta en un robot móvil simulado, para ello se hace uso de un simulador de robots, de esa forma se puede determinar el comportamiento de las memorias asociativas Alfa-Beta en el problema de localización. De modo que se ha seleccionado la plataforma de Microsoft Robotics Developer Studio (MRDS), ya que dispone de un potente simulador de robots, bajo diferentes escenarios. Además se ha seleccionado el robots Lego Mindstorm NXT dentro del conjunto de robot con los que MRDS es compatible (para saber más de MRDS véase la sección 3.3 y el apéndice D).

Para este experimento se han propuesto una serie de pasos a seguir:

1. ***Diseño del robot y escenario***. Determinar las características del robot y del escenario de navegación, después realizar su modelado en 3D.
2. ***Obtención del banco de datos***. Diseño e implementación de los algoritmos necesarios para que el robot sea capaz de navegar por el entorno de navegación, todo ello con el fin de acumular datos suficientes, los cuales son proporcionados por los sensores durante las rondas de navegación. Los datos de los sensores serán agrupados para conformar un banco de datos considerable.
3. ***Implementación del algoritmo de la memoria asociativa Alfa-Beta***. El objetivo es presentar los resultados de las memorias asociativas Alfa-Beta tipo Max, empleando el mismo algoritmo del experimento número 1 presentado en la sección 4.1.3.1, con la diferencia de estar trabajando en un robot y entorno simulado.

4.2.1. Paso 1. Diseño del robot y escenario

En primera instancia es necesario elegir el tipo de escenario en el que se pretende que el robot navegue y por lo tanto también que se localice. Existen una serie de características a considerar cuando se hace la selección de escenarios de navegación tales como:

Estático-Dinámico. Depende del nivel de dinamismo dentro del entorno, en otras palabras que tanto cambia el entorno mientras el robot navega por este.

Exterior-Interior. Es una característica tomada en cuenta principalmente para elegir el tipo de sensores, ya que en ambientes cerrados se presentan ciertas condiciones idóneas para la utilización de sensores con poco alcance y medianamente sensibles a los ruidos, mientras que en exteriores es necesario considerar los espacios abiertos en donde por ejemplo la utilización de un sensor ultrasónico seria poco factible, además de lidiar constantemente con el ruido proveniente de diversos factores.

Para esta etapa de experimentación se ha seleccionado un escenario estático y controlado, además de que el robot navegará en un espacio cerrado bajo un entorno simulado en MRDS.

Previamente en el capítulo III se ha seleccionado la plataforma de simulación así como también el tipo de robot a utilizar y los diferentes sensores de los que haremos uso. Por lo que es importante señalar que el entorno de simulación de Microsoft Robotics Developer Studio es en tercera dimensión, además por default trae ya precargados algunos escenarios de los que podemos hacer uso, no obstante también se pueden diseñar escenarios propios bajo las necesidades de los usuarios.

Se ha elegido trabajar con uno de los escenarios que trae precargado, el cual es la simulación de un departamento tal como se muestra en la figura 4.24.

Figura 4.24: Escenario predefinido de la galería de Microsoft Robotics Developer Studio (MRDS).

No obstante, los elementos de este espacio son en su mayoría innecesarios para las pruebas de localización, por lo que se ha eliminado una cantidad importante de elementos, dejando el escenario final como se muestra en la figura 4.25.

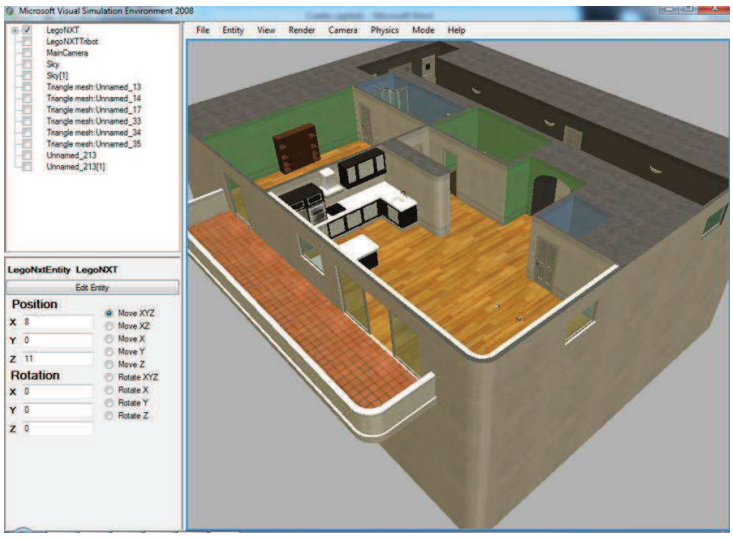


Figura 4.25: Escenario de la galería de Microsoft Robotics Developer Studio (MRDS) en la que se han eliminado de forma intencional elementos.

Otro aspecto de importancia a considerar, con respecto al mapa original son la selección de los espacios en los que el robot puede navegar y aquellos en los que no, para lo cual se diseño un mapa en 2D con el fin de vislumbrar de forma más clara los espacios en los que se realizaran las diferentes mediciones, cabe señalar que se han considerado en el mapa 2D, solo los espacios en los que el robot móvil tomará las mediciones de los correspondientes sensores, esto con el fin de contar con un banco de datos maleable y que libere carga de procesamiento, sin embargo se han considerado en trabajos a futuro ampliar el espectro de navegación y por lo tanto también el espacio de obtención de datos (Véase la figura 4.26).

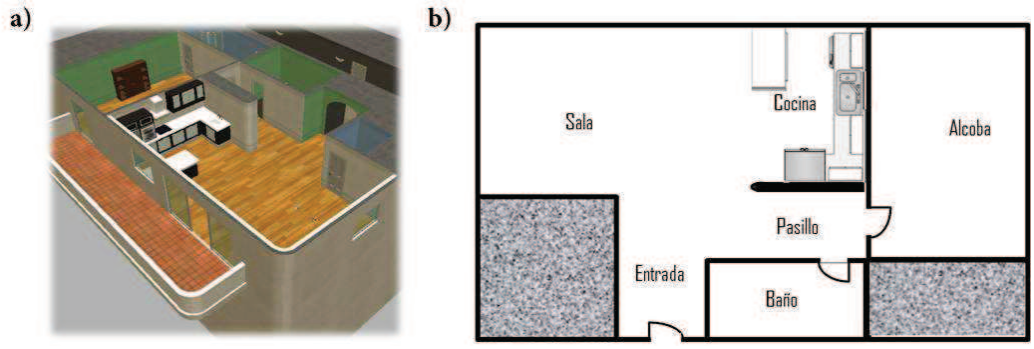


Figura 4.26: a) Escenario de la galería de MRDS en modelado 3D. b) Mapa en 2D creado a partir de el escenario en 3D, con modificaciones que limitan el acceso del robot a áreas cerradas o bien al exterior (terraza y pasillo principal).

4.2.1.1. Modelado de Lego Mindstorm

Ahora bien, un caso particular fue el modelado del robot móvil Lego Mindstorm NXT, ya que en un principio no se considero la realización de un modelado para este robot, porque MRDS es capaz de soportar un modelo de este robot predefinido por la herramienta, no obstante el robot Mindstorm que viene por default en MRDS solo está integrado por los sensores básicos. Por lo que fue necesario implementar el modelado de un robot móvil capaz de ser simulado en MRDS y que considerará cada uno de los elementos incluidos en el kit básico del modelo Tribot, tales como sensores, ruedas, brick, entre otros, y sumar los sensores complementarios como el sensor ultrasónico, brújula y el acelerómetro (véase las especificaciones de los sensores implementados en el robot en el apéndice D). En la figura 4.27 se puede observar el modelo predefinido por default de MRDS a la izquierda (a), mientras que a la derecha (b) se encuentra el modelado del robot que ya tiene integrado los sensores extras.

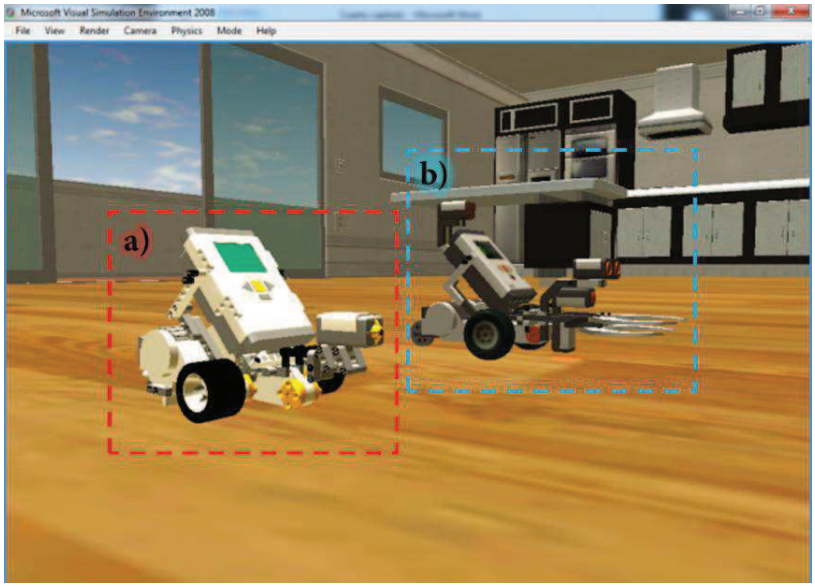


Figura 4.27: Modelos de simulación del Robot Lego Mindstorm NXT de Microsoft Robotics Developer Studio. a) Robot predefinido por MRDS. b) Robot modelado con sensores complementarios.

4.2.2. Paso 2. Obtención del banco de datos.

Con base en el mapa (b) de la figura 4.26 se han seleccionando las zonas que constituirán la parte medular de la clasificación del robot, es decir que el robot debe ser capaz de distinguir su localización bajo el esquema de sección por zonas. A cada zona se le ha asignado una área determinada la cual se encuentra representada por una letra mayúscula tal como se muestra en la figura 4.28.

4.2.2.1. *Determinación de Clases.*

En este punto es posible identificar las clases, ya que cada una de las zonas formaran parte de las 6 clases, de las cuales se pretende que el robot sea capaz de identificar, es decir que el robot a partir de un conjunto de información extraída de sus sensores debe ser capaz de determinar en qué zona se encuentra del escenario.

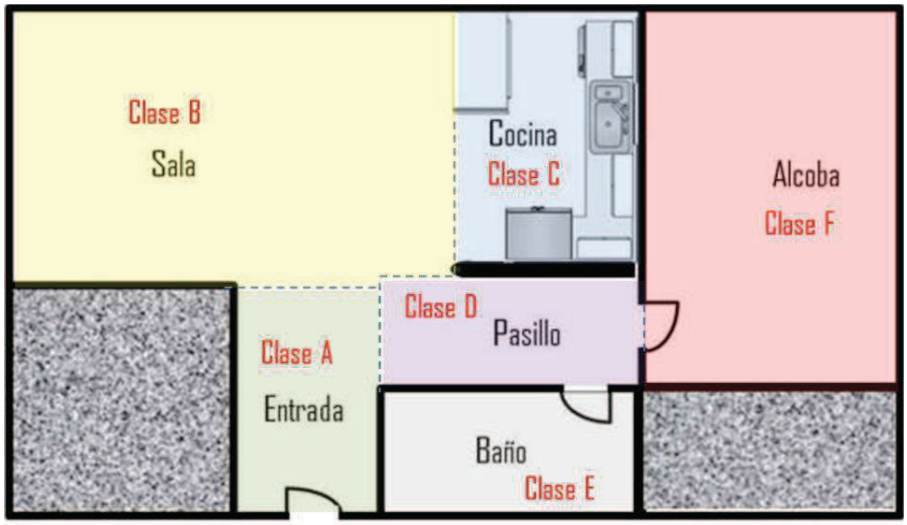


Figura 4.28: Mapa en 2D creado a partir de el escenario en 3D, Etiquetado con letras mayúsculas por zonas. Asignando: Clase A (Entrada), Clase B (Sala), Clase C (Cocina), Clase D (Pasillo), Clase E (Baño), Clase F (Alcoba).

4.2.2.2. *Selección de Rasgos.*

Para ello se han elegido los datos obtenidos de 3 sensores (sensor brújula, acelerómetro y ultrasónico) acoplados al robot Tribot, los cuales trabajan bajo una tasa de 100 muestras por segundo, para el experimento n.2 se ha bajado el rango de frecuencia a la mitad, de modo que los sensores trabajaran a una tasa de 50 muestras por segundo (véase la sección D para la descripción de los sensores empleados en esta fase de experimentación).
Ahora bien, ya que el robot primero debe recolectar un conjunto de muestras con los sensores que han sido acoplados a él, se debe establecer un mecanismo de navegación con la finalidad de almacenar las muestras de los sensores, las cuales posteriormente servirán durante la fase de aprendizaje y prueba de la memoria asociativa Alfa-Beta.

Algoritmo de navegación Para representar el camino que el robot debe seguir para trasladarse de un nodo a otro nodo es necesario considerar una serie de aspectos para que el robot pueda ser capaz de cumplir con su objetivo; el primer aspecto es el tipo de algoritmo, para este caso se ha implementado el procedimiento getPath (origen, destino). Este procedimiento permite generar un camino en el que los distintos puntos que lo conforman distan entre sí distancias similares. Estos puntos se almacenan en un array cuya última posición contiene la posición destino. En el código, el camino a seguir se encuentra almacenado en la variable Path. Ahora se debe determinar las distintas posiciones, las cuales se ven representadas durante el entorno de simulación como las coordenadas X y Z, ya que en el entorno de simulación de MRDS no es necesario el desplazamiento en el eje Y, ya que los valores en este eje permiten al robot flotar o hundirse en el piso.
El siguiente código fue empleado para la navegación del robot de nodo a nodo:

Algoritmo 4.5 Código de navegación programado en lenguaje C# para la navegación del robot Lego Mindstorm Tribot modelado en la sección 4.2.1.1.

```
private ArrayList getPath(comlink.Position from, comlink.Position to)
      {
         ArrayList result = new ArrayList();
                        // //Cada paso que se mueve (como máximo) a 5 centímetros en X y Z.
                        int steps = (int)(Distance(from, to) / .05); // Número de pasos intermedios
                       int xlength = (int) (Math.Abs(from.X - to.X)); //Distancia entre cada paso en el
eje X
                        int zlength = (int)(Math.Abs(from.Z - to.Z)); //Distancia entre cada paso en el
eje Z
                        bool fixedX = false;
                        bool fixedZ = false;

                        if (steps == 0)
                        result.Add(to);
                        return result;
                        }
                        if (xlength < .025) {
//Si la distancia en el eje X es pequeña entonces desplazarse a este valor de coordenadas
                             fixedX = true;
                        }
                        if (zlength < 5) {
//Si la distancia en el eje Z es pequeña entonces desplazarse a este valor de coordenadas
                             fixedZ = true;
                        }
                        if (!fixedX) { xlength /= steps; }
                        if (!fixedZ) { zlength/= steps; }
                        //Calcular la posición de cada paso y agregarla a la ruta.
                        for (int i = 0; i < (steps-1); i++){
                             comlink.Position point = new comlink.Position();
                             if (!fixedX) {
                                  if (to.X > from.X) {
                                       point.X = from.X + (i + 0.01) * xlength;
                                  }else{
                                       point.X = from.X - (i + 0.01) * xlength;
                                  }
                             }else{
                                  point.X = to.X;
                             }
                             if (!fixedZ) {
                                  if (to.Z > from.Z){
                                       point.Z = from.Z + (i + 1) * zlength;
                                  }
                                  else{
                                        point.Z = from.Z - (i + 1) * zlength;
                                  }
                             } else { point.Z = to.Z;
                             }
                        result.Add(point);
                        }
                        comlink.Position finish = new comlink.Position();
//Final de paso: Se ha punto destino
                        finish.X = to.X;
                        finish.Z = to.Z;
                        result.Add(finish);
                        return result;
      }
```

Una vez que el robot es capaz de crear caminos es necesario un algoritmo que nos indique las correspondientes acciones que son necesarias para su desplazamiento a cada uno de los puntos o nodos de dicho camino. Para ello, el primer paso consiste en determinar la posición relativa del siguiente punto con respecto a sí mismo. Esta posición relativa corresponde con las combinaciones de los cuatro puntos cardinales: Norte, Sur, Este y Oeste. Esta posición relativa se consigue a través de la obtención de la distancia entre el robot y el punto destino con respecto a los ejes X y Z.

Por tanto, se pueden dar dos situaciones durante los cálculos odométricos para conocer las distancias:

- ≻ Si el robot no está orientado hacia el punto cardinal que corresponde a la posición relativa en que se encuentra el punto al que se dirige, éste realiza un movimiento de rotación que lo sitúe en la orientación adecuada.
- ≻ Si se encuentra orientado en la misma dirección, se realiza un proceso de orientación más preciso, con el fin de que el robot se desplace en la dirección adecuada.

En el primer caso, la rotación se realiza de distinta manera dependiendo de la orientación que se desee conseguir. Si se desea que el robot se oriente hacia el Norte, Sur, Este y Oeste, se realiza una llamada al método lookUp, lookDown, lookRight y lookLeft, respectivamente. De esta manera, el robot rotará hasta orientarse en la dirección deseada, con una precisión de 5 grados. Por ejemplo, si la localización destino se encuentra inmediatamente al sur del robot, la posición relativa será Sur. Si el robot no está orientado en esta dirección, correspondiente al valor 0 de Yaw, se invoca el método lookDown(). Este método hará que el robot rote sobre sí mismo en el sentido que rápidamente permita obtener un valor de Yaw comprendido en el intervalo [-0.1, 0.1].

Por otro lado, si la localización destino está en dirección Noroeste, Noreste, Sureste o Suroeste, el proceso de orientación se realiza en dos pasos. Primero, se hace rotar al robot hasta que se encuentre orientado en la dirección deseada y, a continuación, se rota al robot más despacio hasta conseguir una orientación próxima a la de la recta que une su posición y el destino.

Para problemas de evasión de obstáculos que son críticos se utilizó un control de la consola de Xbox, con la que es posible mover el robot Tribot manualmente sin perder las constantes muestras de los sensores.

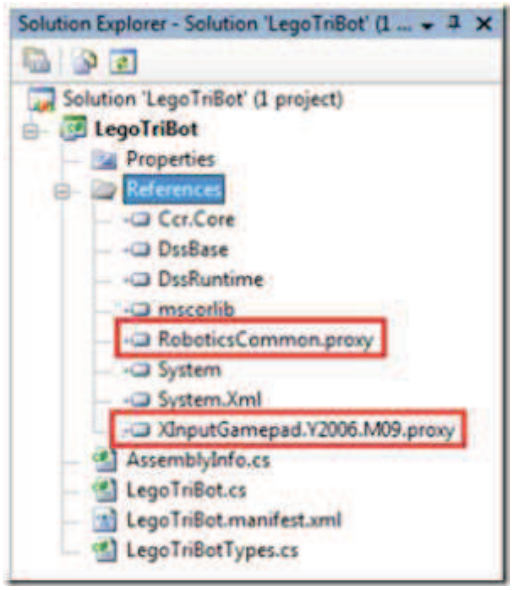


Figura 4.29: Esquema en el que se agrega un periferico externo para la navegación del robot en puntos críticos. Se ha empleado un control de Xbox acoplado a los motores del robot Tribot.

Resumiendo

Hasta este punto se hizo navegar al robot Tribot en el escenario de navegación (véase la sección 4.2.1), producto del modelado de dicho escenario se generan seis clases (Véase la sección 4.2.2.1) bajo un algoritmo de navegación (véase la sección 4.2.2.2) y el empleo de un control de la consola Xbox, los tres sensores empleados en la navegación han funcionado en sincronización a una frecuencia de 50 muestras por segundo (véase la sección 4.2.2.2). La tabla 4.4 muestra los patrones que se han tomado en cada clase.

Tabla 4.4: Asignación de patrones con cada clase, durante la navegación del robot Tribot en el escenario de navegación simulado. Entre todas las clases se generan un total de **33,755** patrones.

Clase	**N. Patrones**	**Tiempo de Navegación**
Clase A	Integrada por 2,600 patrones	52 segundos
Clase B	Integrada por 7,800 patrones	156 segundos
Clase C	Integrada por 4,455 patrones	89 segundos
Clase D	Integrada por 7,250 patrones	145 segundos
Clase E	Integrada por 4,750 patrones	95 segundos
Clase F	Integrada por 6,900 patrones	138 segundos

Cada patrón esta integrado por tres rasgos que pertenecen a las muestras de cada sensor en un instante dado de navegación.

4.2.3. Resultado de la implementación de la memoria asociativa Alfa-Beta tipo Max

El algoritmo mostrado en la sección 4.1.3.1, así como el método «K-fold-cross-validation» fueron programados para este experimento en el lenguaje de programación C#, bajo la plataforma de MRDS, bajo un robot modelado con tres sensores (brújula, acelerómetro y ultrasónico). Los resultados obtenidos en la implementación de la memoria son:

Se diseñó e implemento una memoria asociativa Alfa-Beta Autoasociativa tipo Max (Véase el algoritmo 4.2), el banco de datos de entrenamiento y de prueba empleados en la memoria están descritos en la tabla 4.4. Para la selección del conjunto de entrenamiento y de prueba se empleo el método K-fold-cross-validation con k =10. Los resultados obtenidos son:

- ≻ **Fase de Aprendizaje:** El conjunto de entrenamiento esta integrado por 30,380 patrones, con 6 clases que representan las secciones (véase la figura 4.28) en las que el robot debe ser capaz de ubicarse.
- ≻ **Fase de Recuperación**: **100 %** de recuperación de patrones aprendidos, obteniendo **0 %** de factor de olvido (se empleó el conjunto de entrenamiento).
- ≻ **Fase de Prueba:** ***91.15 %*** de asertividad considerando 3,375 patrones en el conjunto de prueba.

Capítulo 5

Conclusiones y Trabajos a Futuro

Este capítulo consta de tres secciones que en conjunto tienen la finalidad de sintetizar y reflexionar sobre el desarrollo del presente trabajo de investigación, además de proponer nuevas líneas de investigación que propicien el fortalecimiento de esta área de estudio. De modo que la primera sección presenta una recapitulación del trabajo de investigación descrito a lo largo de los cuatro capítulos anteriores, después en una segunda sección se enlistan las conclusiones derivadas de un minucioso análisis sobre el comportamiento de las memorias asociativas Alfa-Beta en el problema de localización de robots móviles. Finalmente en la última sección se formula una serie de ideas que servirán como guia para trabajos futuros que se deslinden de la presente tesis.

5.1. Síntesis

Hasta este punto el desarrollo del presente trabajo de tesis ha cumplido exitosamente con los objetivos planteados en la sección 1.4.1 y 1.4.2, por lo que es de importancia realizar una breve revisión del conjunto de esfuerzos que ha permitido concluir de forma satisfactoria con la investigación propuesta.

La primera meta a superar se trataba de realizar una detallada búsqueda y adquisición de conocimientos científicos y tecnológicos que permitieran abordar correctamente el problema de la localización de robots móviles, estos esfuerzos fueron vertidos en el capítulo II, por lo tanto en él se encuentra una descripción de los aspectos generales y particulares que hacen de este problema una línea de investigación compleja y fascinante a la vez.

Para el capítulo III el reto fue seleccionar, conocer y comprender los métodos y herramientas que serían usados durante la fase de experimentación, por lo que fue necesario conocer a profundidad los principios matemáticos de las memorias asociativas de tipo Alfa-Beta, esto con el propósito de emplear correctamente los algoritmos necesarios para su implementación. Después se realizó un arduo estudio de los simuladores de robots disponibles así como una recopilación de sus principales características, derivado de ello se decidió emplear la plataforma de Microsoft Robotics Developer Studio (MRDS). Por lo tanto el capítulo III es producto de los esfuerzos invertidos en el análisis e investigación de las memorias asociativas Alfa-Beta y MRDS.

Finalmente se tiene la fase de experimentación, la cual representa la parte medular del trabajo de investigación y también es la fase más interesante; el desafío fue determinar el alcance, las condiciones y el impacto de cada experimento, el objetivo era proponer experimentos que permitieran probar la asertividad de las memorias asociativas Alfa-Beta en el problema de localización de robots móviles. Por lo tanto en el capítulo IV se redacta a detalle los dos experimentos que prueban la eficacia de las memorias.

5.2. Conclusiones

Una vez realizados los experimentos propuestos se obtuvieron resultados de los que se concluye que:

1. Se demostró que el uso de las memorias asociativas Alfa-Beta en el problema de localización en robots móviles representan una solución innovadora, eficaz y competitiva.
2. Se comprobó que las memorias Autoasociativas Alfa-Beta recuperan de manera óptima el conjunto de entrenamiento aprendido por la memoria.
3. Se observó que el tipo de configuración de un robot influye directamente en la complejidad de navegación y por consecuencia en la toma de lecturas de los sensores lo que facilita o dificulta su localización.
4. Se generó un robusto banco de datos a partir de las pruebas realizadas en el experimento simulado con MRDS, con el cual se puede trabajar en futuras líneas de investigación.

5.3. Trabajo a futuro

1. Probar las memorias asociativas con las rondas de navegación del robot SCITOS G5 que no fueron empleadas durante el experimento 1.
2. Emplear las diferentes variantes de los modelos de memorias asociativas Alfa-Beta, para lograr resultados mejores a los obtenidos en este trabajo de tesis.
3. Probar las memorias asociativas Alfa-Beta en el entorno de simulación de MRDS empleando un robot distinto al utilizado en la presente tesis.
4. Realizar el diseño e implementación de un sistema de localización de un robot móvil real, con la finalidad de probar su eficiencia bajo entornos y factores no controlados.

Referencias

[1] A.L. Freire, G.A. Barreto, M. Veloso, and A.T. Varela. Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study. In *Robotics Symposium (LARS), 2009 6th Latin American*, pages 1–6. IEEE, 2009.

[2] M. Hazas, J. Scott, and J. Krumm. Location-aware computing comes of age. *Computer*, 37(2):95–97, 2004.

[3] K. Capek. Rossum's universal robots. *Prague, CZ*, 1920.

[4] K.S. Fu. *Robótica: Control, detección, visión e inteligencia*. McGraw-Hill Interamericana de España, 1988.

[5] R.S. Ortigoza, J.R.G. Sánchez, V.R.B. Sotelo, MA Molina, V.M.H. Guzmán, and G.S. Ortigoza. Una panorámica de los robots móviles. *Télématique*, (003):1–14, 2007.

[6] TM Knasel. Mobile robotics-state of the art review. *Robotics*, 2(2):149–155, 1986.

[7] TM Knasel. Fifth international conference on assembly automation. *Robotics*, 1(1):49–62, 1985.

[8] J.J. Leonard and H.F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *Robotics and Automation, IEEE Transactions on*, 7(3):376–382, 1991.

[9] J. Borenstein, HR Everett, and L. Feng. Where am i? sensors and methods for mobile robot positioning. *University of Michigan*, 119:120, 1996.

[10] S.P. Engelson. *Passive map learning and visual place recognition*. PhD thesis, Yale University, 2000.

[11] A.B. Chatfield. *Fundamentals of high accuracy inertial navigation*, volume 174. AIAA (American Institute of Aeronautics & Astronautics), 1997.

[12] B. Barshan and H.F. Durrant-Whyte. Inertial navigation systems for mobile robots. *Robotics and Automation, IEEE Transactions on*, 11(3):328–342, 1995.

[13] K.R. Britting. *Inertial navigation systems analysis*. 2010.

[14] B.S. Cho, W. Moon, W.J. Seo, and K.R. Baek. A study on localization of the mobile robot using inertial sensors and wheel revolutions. *Intelligent Robotics and Applications*, pages 575–583, 2011.

[15] A. Georgiev and P.K. Allen. Localization methods for a mobile robot in urban environments. *Robotics, IEEE Transactions on*, 20(5):851–864, 2004.

[16] L. Ojeda and J. Borenstein. Methods for the reduction of odometry errors in over-constrained mobile robots. *Autonomous Robots*, 16(3):273–286, 2004.

[17] J. Park, S. Choi, and J. Lee. Beacon scheduling algorithm for localization of a mobile robot. *Intelligent Robotics and Applications*, pages 594–603, 2011.

[18] T.K. Yang, J.H. Park, and J.M. Lee. Multi block localization of multiple robots. *Progress in Robotics*, pages 293–299, 2009.

[19] S. Genchev, P. Venkov, and B. Vidolov. Trilateration analysis for movement planning in a group of mobile robots. *Artificial Intelligence: Methodology, Systems, and Applications*, pages 353–364, 2008.

[20] M. Piaggio, A. Sgorbissa, and R. Zaccaria. Navigation and localization for service mobile robots based on active beacons. *SYSTEMS SCIENCE-WROCLAW-*, 27(4):71–84, 2001.

[21] E. Brassart, C. Pegard, and M. Mouaddib. Localization using infrared beacons. *Robotica*, 18(2):153–161, 2000.

[22] S. Frintrop, P. Jensfelt, and H. Christensen. Simultaneous robot localization and mapping based on a visual attention system. *Attention in Cognitive Systems. Theories and Systems from an Interdisciplinary Viewpoint*, pages 417–430, 2007.

[23] U. Frese. Treemap: An o (log n) algorithm for simultaneous localization and mapping. *Spatial Cognition IV. Reasoning, Action, Interaction*, pages 455–477, 2005.

[24] A. Bais, R. Sablatnig, J. Gu, and S. Mahlknecht. Active single landmark based global localization of autonomous mobile robots. *Advances in Visual Computing*, pages 202–211, 2006.

[25] S. Thrun. Bayesian landmark learning for mobile robot localization. *Machine Learning*, 33(1):41–76, 1998.

[26] I. Shimshoni. On mobile robot localization from landmark bearings. *Robotics and Automation, IEEE Transactions on*, 18(6):971–976, 2002.

[27] M. Betke and L. Gurvits. Mobile robot localization using landmarks. *Robotics and Automation, IEEE Transactions on*, 13(2):251–263, 1997.

[28] A. Arleo, J. del R Millan, and D. Floreano. Efficient learning of variable-resolution cognitive maps for autonomous indoor navigation. *Robotics and Automation, IEEE Transactions on*, 15(6):990–1000, 1999.

[29] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *Robotics, IEEE Transactions on*, 23(1):34–46, 2007.

[30] J.H. Lim and C.U. Kang. Grid-based localization of a mobile robot using sonar sensors. *Journal of Mechanical Science and Technology*, 16(3):302–309, 2002.

[31] S. Thrun, W. Burgard, and D. Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Autonomous Robots*, 5(3):253–271, 1998.

[32] GC Anousaki and KJ Kyriakopoulos. Simultaneous localization and map building for mobile robot navigation. *Robotics & Automation Magazine, IEEE*, 6(3):42–53, 1999.

[33] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 116–121. IEEE, 1985.

[34] A. Elfes. Sonar-based real-world mapping and navigation. *Robotics and Automation, IEEE Journal of*, 3(3):249–265, 1987.

[35] M. Weigl, B. Siemiatkowska, KA Sikorski, and A. Borkowski. Grid-based mapping for autonomous mobile robot. *Robotics and Autonomous Systems*, 11(1):13–21, 1993.

[36] B.J. Kuipers. *Representing Knowledge of Large-Scale Space1*. PhD thesis, Massachusetts Institute of Technology, 1977.

[37] T. Goedemé, T. Tuytelaars, and L.V. Gool. Visual topological map building in self-similar environments. *Informatics in Control Automation and Robotics*, pages 195–205, 2008.

[38] Ó. Mozos, C. Stachniss, A. Rottmann, and W. Burgard. Using adaboost for place labeling and topological map building. *Robotics Research*, pages 453–472, 2007.

[39] W.H. Huang and K.R. Beevers. Topological mapping with sensing-limited robots. *Algorithmic Foundations of Robotics VI*, pages 235–250, 2005.

[40] R. Berry, K. Loebbaka, and E. Hall. Sensors for mobile robots. 449(2):584–588, 1984.

[41] R. Siegwart and I.R. Nourbakhsh. *Introduction to autonomous mobile robots*. MIT press, 2004.

[42] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. John Wiley & Sons, 2001.

[43] A. Webb. *Statistical Pattern Recognition*. Oxford University Press, 1999.

[44] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine learning. Springer*, 29(2):131–163, 1997.

[45] C. Yáñez-Márquez and J.L. Díaz de León Santiago. Memorias autoasociativas morfológicas min: condiciones suficientes para convergencia, aprendizaje y recuperación de patrones. *México: IT-177, Serie Azul, CIC-IPN*, 2003.

[46] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

[47] JL Díaz'de'León, C. Yáñez'Márquez, and FA Sánchez'Garfias. Reconocimiento de patrones. *Enfoque probabilístico'estadístico. IT*, 83, 2003.

[48] A. Kandel. *Introduction to pattern recognition: statistical, structural, neural, and fuzzy logic approaches*, volume 32. World Scientific Publishing Company Incorporated, 1999.

[49] R. Schalkoff. Pattern recognition: Statistical. *Structural and Neu*, 1992.

[50] R.C. Gonzalez and M.G. Thomason. Syntactic pattern recognition: An introduction. 1978.

[51] D. Michie, D.J. Spiegelhalter, C.C. Taylor, and J. Campbell. Machine learning, neural and statistical classification. 1994.

[52] A. Argüelles, C. Yáñez, I. López, and O. Camacho. Prediction of co and no x levels in mexico city using associative models. *Artificial Intelligence Applications and Innovations*, pages 313–322, 2011.

[53] A. Arguelles-Cruz, I. López-Yáñez, M. Aldape-Pérez, and N. Conde-Gaxiola. Alpha-beta weightless neural networks. *Progress in Pattern Recognition, Image Analysis and Applications*, pages 496–503, 2008.

[54] M. Aldape-Pérez, C. Yánez-Márquez, and LO Lopez Leyva. Feature selection using a hybrid associative classifier with masking techniques. In *Artificial Intelligence, 2006. MICAI'06. Fifth Mexican International Conference on*, pages 151–160. IEEE, 2006.

[55] R. Santiago-Montero. *Clasificador híbrido de patrones basados en la Lernmatrix de Steinbuch y Linear Associator de Anderson-Kohonen.* Tesis de la maestria de ciencias de la computación, Centro de Investigación en Computación del Instituto Politécnico Nacional (CIC-IPN), 2003.

[56] C. Yáñez-Márquez and J.L. Díaz de León Santiago. Memorias asociativas basadas en relaciones de orden y operaciones binarias. *Computación y Sistemas (Revista Iberoamericana de Computación incluida en el índice de CONACyT)*, 6(4):300–311, 2003.

[57] C. Yáñez-Márquez and J.L. Díaz de León Santiago. Memorias autoasociativas morfológicas min: condiciones suficientes para convergencia, aprendizaje y recuperación de patrones. Technical report, México„ 2003.

[58] C. Yáñez-Márquez. *Memorias Asociativas basadas en Relaciones de Orden y Operadores Binarios*. Phd., Instituto Politécnico Nacional-CIC, México, 2002.

[59] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 111–147, 1974.

[60] I.H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[61] M.J. Caruso. Applications of magnetic sensors for low cost compass systems. In *Position Location and Navigation Symposium, IEEE 2000*, pages 177–184. IEEE, 2000.

[62] J.M. O'Kane and S.M. LaValle. Localization with limited sensing. *Robotics, IEEE Transactions on*, 23(4):704–716, 2007.

[63] C.C. Tsai. A localization system of a mobile robot by fusing dead-reckoning and ultrasonic measurements. In *Instrumentation and Measurement Technology Conference, 1998. IMTC/98. Conference Proceedings. IEEE*, volume 1, pages 144–149. IEEE, 1998.

[64] H. Myung, H.K. Lee, K. Choi, and S. Bang. Mobile robot localization with gyroscope and constrained kalman filter. *International Journal of Control, Automation and Systems*, 8(3):667–676, 2010.

[65] H. Chung, L. Ojeda, and J. Borenstein. Accurate mobile robot dead-reckoning with a precision-calibrated fiber-optic gyroscope. *Robotics and Automation, IEEE Transactions on*, 17(1):80–84, 2001.

[66] J. Borenstein and L. Feng. Gyrodometry: A new method for combining data from gyros and odometry in mobile robots. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 1, pages 423–428. IEEE, 1996.

[67] H.H.S. Liu and G.K.H. Pang. Accelerometer for mobile robot positioning. *Industry Applications, IEEE Transactions on*, 37(3):812–819, 2001.

[68] B.S. Cho, W. Moon, W.J. Seo, and K.R. Baek. A dead reckoning localization system for mobile robots using inertial sensors and wheel revolution encoding. *Journal of mechanical science and technology*, 25(11):2907–2917, 2011.

[69] R.B. Langley. Why is the gps signal so complex. *GPS world*, 1(3):56–59, 1990.

[70] T.A. Herring. The global positioning system. *Scientific American*, 274(2), 1996.

[71] I.A. Getting. Perspective/navigation-the global positioning system. *Spectrum, IEEE*, 30(12):36–38, 1993.

[72] A. Pozo-Ruz, A. Ribeiro, MC García-Alegre, L. García, D. Guinea, and F. Sandoval. Sistema de posicionamiento global (gps): Descripción, análisis de errores, aplicaciones y futuro. *ETS Ingenieros de Telecomunicaciones. Universidad de Málaga*, 2000.

[73] L. Jurisica, A. Vitko, F. Duchon, and D. Kastan. Statistical approach to gps positioning of mobile robot. *Journal of Control Engineering and Applied Informatics*, 12(2):44–51, 2010.

[74] H. Chae, S. Choi, W. Yu, J. Cho, et al. Autonomous navigation of mobile robot based on dgps/ins sensor fusion by ekf in semi-outdoor structured environment. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1222–1227. IEEE, 2010.

[75] K. Ohno, T. Tsubouchi, B. Shigematsu, and Shin'ichi Yuta. Differential gps and odometry-based outdoor navigation of a mobile robot. *Advanced Robotics*, 18(6):611–635, 2004.

[76] A.R. Jiménez, F. Seco, C. Prieto, and J. Roa. Tecnologías sensoriales de localización para entornos inteligentes. In *Proc. Simposio UCAmI (Granada)*, pages 75–86, 2005.

[77] M.J. Segura, F.A. Auat Cheein, J.M. Toibero, V. Mut, and R. Carelli. Ultra wide-band localization and slam: A comparative study for mobile robot navigation. *Sensors*, 11(2):2035–2055, 2011.

[78] J. Gonzalez, JL Blanco, C. Galindo, A. Ortiz-de Galisteo, JA Fernández-Madrigal, FA Moreno, and JL Martínez. Mobile robot localization based on ultra-wide-band ranging: A particle filter approach. *Robotics and autonomous systems*, 57(5):496–507, 2009.

[79] M.J. Segura, V.A. Mut, and H.D. Patiño. Mobile robot self-localization system using ir-uwb sensor in indoor environments. In *Robotic and Sensors Environments, 2009. ROSE 2009. IEEE International Workshop on*, pages 29–34. IEEE, 2009.

[80] D.I. Tapia, J.R. Cueli, O. García, J.M. Corchado, J. Bajo, and A. Saavedra. Identificación por radiofrecuencia: Fundamentos y aplicaciones. *Proceedings de las primeras Jornadas Científicas sobre RFID. Ciudad Real, Spain*, 2007.

[81] B.S. Choi, J.W. Lee, J.J. Lee, and K.T. Park. A hierarchical algorithm for indoor mobile robot localization using rfid sensor fusion. *Industrial Electronics, IEEE Transactions on*, 58(6):2226–2235, 2011.

[82] A. Codas, M. Devy, and C. Lemaire. Robot localization algorithm using odometry and rfid technology. In *Intelligent Autonomous Vehicles*, volume 7, pages 569–574, 2010.

[83] S. Park and S. Hashimoto. Autonomous mobile robot navigation using passive rfid in indoor environment. *Industrial Electronics, IEEE Transactions on*, 56(7):2366–2373, 2009.

[84] K. Kodaka, H. Niwa, Y. Sakamoto, M. Otake, Y. Kanemori, and S. Sugano. Pose estimation of a mobile robot on a lattice of rfid tags. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1385–1390. IEEE, 2008.

[85] S. Han, H.S. Lim, and J.M. Lee. An efficient localization scheme for a differential-driving mobile robot based on rfid system. *Industrial Electronics, IEEE Transactions on*, 54(6):3362–3369, 2007.

Apéndice A

Cálculos cinemáticos

Este anexo tiene la finalidad de describir detalladamente la fundamentación matemática derivada de los diferentes cálculos cinemáticos y estocásticos de los robots móviles con ruedas, la cual incluye las típicas configuraciones (ackerman, triciclo, diferencial y síncrona).

Para comenzar a realizar el estudio de cálculos de odometría para los robots móviles con ruedas, es necesario definir algunos conceptos básicos que son utilizados a lo largo de la descripción de los diferentes tipos de configuraciones.

La odometría tiene como propósito estimar la posición del robot móvil con ruedas en un instante determinado. La palabra odometría proviene de las palabras griegas "hodos" que significa trayecto y "metron" que significa medida. Entonces la odometría se usa para estimar la posición de un robot móvil con ruedas calculando la distancia recorrida por el robot con respecto a un punto inicial y al tiempo consumido en el desplazamiento. El cálculo de forma general se realiza tomando en consideración el perímetro de la rueda y el número de vueltas (revoluciones) que el robot realizó durante el desplazamiento. Actualmente se utilizan encoders incrementales para realizar una acumulación gradual de los datos relacionados con el conteo de vueltas [9].

La idea fundamental es la integración de información incremental del movimiento a lo largo del tiempo, por lo que la acumulación de errores es una de las desventajas de esta clase de método para la estimación de la posición de un robot.

Este tipo de método para saber la posición de un robot es muy utilizado ya que su implementación no es compleja y presenta buena precisión en desplazamientos relativamente cortos, además actualmente existen numerosas técnicas para corregir la acumulación de errores. Razón por la cual se muestra un modelo de estimación de posición basado en odometría de cada una de las configuraciones que se presentan en este trabajo tesis.

Para ello es necesario conocer el número de revoluciones de la rueda durante el desplazamiento del robot, como ya se menciono actualmente se usan encoders (incrementales o absolutos) que son colocados sobre los motores de tracción.

El empleo del encoder es sencillo ya que se emplea un factor que convierte los pulsos del encoder en una descomposición lineal con lo que se puede calcular la velocidad de la rueda. El factor para dicha conversión se determina por medio de la ecuación A.1.

$$C_m = \frac{\pi D_n}{nC_e} \tag{A.1}$$

Donde:

C_m =Factor de conversión.
D_n =Diámetro nominal de la rueda (mm).
C_e =Resolución del encoder por revolución (dato proporcionado por el fabricante).
n =Relación de engranes, engranes de reducción entre el motor y la rueda de tracción.

Suponiendo que el espacio de muestreo está representado por i y el encoder de la rueda de tracción izquierda presenta un incremento de pulsos N_L, mientras que el encoder de la rueda de tracción derecha presenta un incremento de pulsos N_R.

Entonces el cálculo del incremento de la distancia recorrida por la rueda izquierda $\triangle U_{L,i}$ y el de la derecha $\triangle U_{R,i}$ se puede calcular como:

$$\triangle U_{\frac{L}{R,i}} = C_m N_{\frac{L}{R,i}} \tag{A.2}$$

Mientras que el incremento lineal de la distancia recorrida por el centro C del robot simbolizado $\triangle U_i$ se calcula por medio de la ecuación A.3.

$$\triangle U_i = \frac{\triangle U_R + \triangle U_L}{2} \tag{A.3}$$

Mientras que el incremento del cambio de orientación esta dado por la ecuación A.4.

$$\triangle \theta_i = \frac{\triangle U_R - \triangle U_L}{b} \tag{A.4}$$

En donde:
b =Es la distancia entre los puntos de contacto de las ruedas con respecto al piso.

De tal manera que la orientación puede ser calculada por la ecuación A.5.

$$\theta_i = \theta_{i-1} + \triangle \theta_i \tag{A.5}$$

Ahora bien la posición (x, y) considerada a partir del punto central del robot puede ser expresada como

$$x_i = x_{i-1} + \triangle U_i cos\,(\theta_i) \tag{A.6}$$

$$y_i = y_{i-1} + \triangle U_i sen\,(\theta_i) \tag{A.7}$$

En donde:
(x_i, y_i) =son la posición con respecto al punto central de robot en un instante dado

Una vez que se han definidos algunos conceptos básicos, se comenzarán a establecer los cálculos cinemáticos para estimar la posición del robot.

Ahora bien la cinemática *se encarga del estudio geométrico de los cuerpos en movimiento y en robots móviles con ruedas lo cual es de ayuda para determinar la posición del robot con respecto a su posición inicial y final, así como su velocidad y aceleración en un instante dado.*

Para comenzar con los cálculos cinemáticos se deben establecer las ecuaciones a las que se desea llegar con el uso de los cálculos cinemáticos las cuales son las ecuacionesA.8 y A.9.

$$x_c = f_1(t) \tag{A.8}$$

$$y_c = f_2(t) \tag{A.9}$$

Cálculos Configuración Diferencial

Ya que en la configuración diferencial cada rueda está controlada por un motor y el punto de contacto de la superficie con las ruedas solo es uno. De modo que se considera que no existirá deslizamiento de la rueda y el piso en dirección al movimiento.

Se recuerda que para obtener un sistema cinemático de ecuaciones es necesario establecer gráficamente todos aquellos factores involucrados que afecten directamente en el movimiento del robot. Como se muestra en la figura A.1.a.

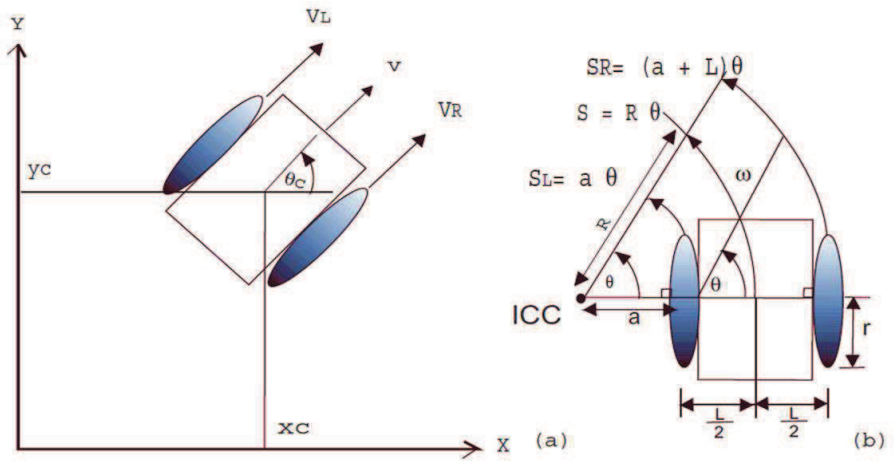


Figura A.1: Configuración de ruedas tipo triciclo clásico considerando el modelo cinemático de los ángulos de dirección.

El la figura A.1 se ilustra el modelo cinemático de una configuración diferencial en donde:

d= Es la distancia longitudinal entre el centro de la rueda delantera y las ruedas traseras.
$\omega(t)$= Es la velocidad angular de rotación con respecto al punto de ICC.
θ= Es el ángulo de rotación del robot.
θ_c= Es el ángulo del robot con respecto a las coordenadas absolutas (x, y).
r= Es el radio de las ruedas.
ICC= Centro Instantáneo de Curvatura.
a= Es la distancia del centro entre la rueda y el punto ICC.
S= Es la distancia de curvatura alrededor de θ.
v= Es la velocidad de traslación del robot.
V_R= Es la velocidad del motor de la rueda derecha.

V_L= Es la velocidad del motor de la rueda izquierda.
L= Es la medición de la separación lateral entre las ruedas.
P_s= Es el vector $[x_c, y_c, \theta_c]$ el cual representa la posición y orientación del robot.
R= Es la distancia del radio de curvatura.

Si se considera que la velocidad angular de la rueda está dada por:

$$velocidad = (radio * \omega) \tag{A.10}$$

Se sustituye entonces V_R, y V_L, en la ecuaciónA.10.

$$V_R, = (radio * \omega_R) \tag{A.11}$$

$$V_L, = (radio * \omega_L) \tag{A.12}$$

La distancia de la curvatura alrededor de θ está representada con la literal S tal como como se puede observar en la figura A.1.b., por lo tanto S_R y S_L son las distancias recorridas por la rueda derecha e izquierda respectivamente.

$$S_L = a\theta \tag{A.13}$$

$$S_R = (a + L)\,\theta \tag{A.14}$$

$$S = \theta R \tag{A.15}$$

Igualando y despejando las ecuaciones A.13 y A.14 se obtiene:

$$S_R = S_L + L\theta \tag{A.16}$$

Se recuerda que al obtener la primera derivada del desplazamiento se obtiene la velocidad, de tal forma que sí se deriva la ecuación A.16 resultara la ecuación A.17, la cual representa la velocidad.

$$V_R = V_L + L\omega \tag{A.17}$$

Despejando ω se obtiene:

$$\omega = \frac{V_R - V_L}{L} \tag{A.18}$$

Ahora se determina la ecuación para S, se hace sumando S_R y S_L :

$$S_R + S_L = 2a\theta + L\theta = 2\theta\left(a + \frac{L}{2}\right) \tag{A.19}$$

En donde

$$R = a + \frac{L}{2} \tag{A.20}$$

Por lo tanto

$$S_R + S_L = 2a\theta + L\theta = 2\theta\,(R) \tag{A.21}$$

Si se despeja R de la ecuación A.15 y sustituye en la ecuación A.21.

$$S_R + S_L = 2a\theta + L\theta = 2\theta\left(\frac{S}{\theta}\right) \tag{A.22}$$

De modo que simplificando resulta la ecuación A.23

$$S = \frac{S_R + S_L}{2} \tag{A.23}$$

Ahora se saca la primera derivada a la ecuación A.23 obteniendo la velocidad como resultante la cual puede verse expresada como:

$$v = \frac{V_R + V_L}{2} \tag{A.24}$$

Para el cálculo de R:

$$v = R\omega \tag{A.25}$$

Después se remplazan las ecuaciones A.24 y A.18 en ecuación A.26.

$$R = \frac{L}{2}\left(\frac{V_R + V_L}{V_R - V_L}\right) \tag{A.26}$$

Ahora bien, realizando un análisis de los valores que puede tomar R en la ecuación A.26 se concluye que sí V_R=V_L entonces $R = \infty$, lo cual significa que el robot ira en línea recta. Ahora bien sí V_R=$-V_L$ entonces $R = 0$, lo que quiere decir que el robot está girando alrededor de su centro.

Si se representa de forma matricial las ecuaciones A.24 y A.18 se obtiene:

$$S = \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{V_R+V_L}{2} \\ \frac{V_R-V_L}{L} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{-1}{L} & \frac{1}{L} \end{bmatrix} \begin{bmatrix} V_L \\ V_R \end{bmatrix} \tag{A.27}$$

Ahora se obtiene $\dot{p_s}$ a partir del producto del a matriz jacobiana $J(\theta)$ con s.

$$\dot{p_s} = \begin{bmatrix} \dot{x_c} \\ \dot{y_c} \\ \dot{\theta_c} \end{bmatrix} = \begin{bmatrix} cos(\theta_c) & 0 \\ sen(\theta_c) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = J(\theta_c)\, s \tag{A.28}$$

Finalmente para obtener la posición del robot se integra la ecuación A.28, de la siguiente forma:

$$x_c = \int_o^t v(T)\, cos(\theta_c)\, dT \tag{A.29}$$

$$y_c = \int_o^t v(T)\, sen(\theta_c)\, dT \tag{A.30}$$

$$\theta_c = \int_o^t \omega(T)\, dT \tag{A.31}$$

Mientras ICC esta dado por:

$$ICC = (x_c - Rsen(\theta),\ y + Rcos(\theta)) \tag{A.32}$$

Cálculos Triciclo Clásico

Para esta configuración se consideran se consideran las dimensiones que se muestra en la figura A.2, en donde el radio de curvatura instantáneo, esta siempre a lo largo de los ejes extendidos de las ruedas traseras.

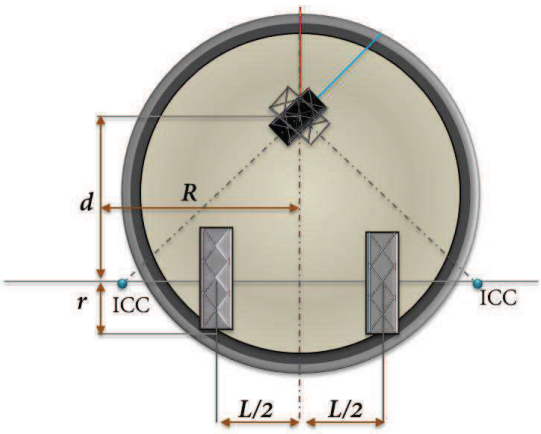


Figura A.2: Sistema de rotación en configuración tipo triciclo clásico.

Una vez definidos los conceptos básicos de la configuración de triciclo clásico, lo siguiente que se tiene que determinar es la posición relativa del robot el cual está localizado en determinado punto de inicio, para ello se hace uso del análisis cinemático del robot.

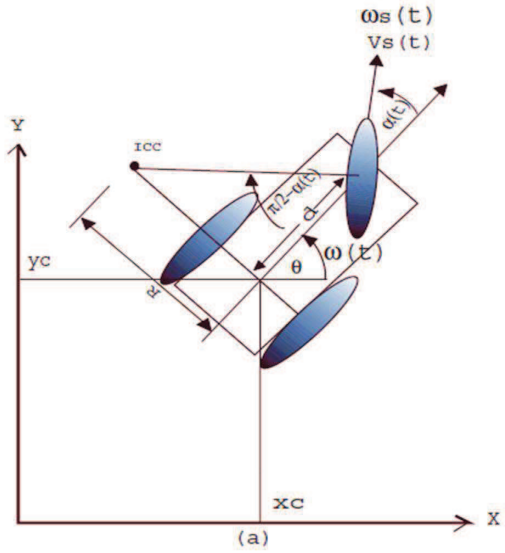


Figura A.3: Configuración de ruedas tipo triciclo clásico considerando los ángulos de dirección.

El la figura A.3 se ilustra la configuración de triciclo clásico en donde:

d= Es la distancia longitudinal entre el centro de la rueda delantera y las ruedas traseras.
$\alpha(t)$= Es el ángulo de la dirección de la rueda con respecto a la dirección en línea recta del robot.
$\omega(t)$= Es la velocidad angular de rotación con respecto al punto de ICC.
θ= Es el ángulo de rotación del robot.
ICC= Centro Instantáneo de Curvatura.
R= Es la distancia del radio de curvatura.
S= Es la distancia de la curvatura alrededor de θ.

Para comenzar se deben definir las ecuaciones a las que se desea llegar mediante el uso de cálculos cinemáticos, las cuales se presentan a continuación.

$$x_c = f_1(t) \tag{A.33}$$

$$y_c = f_2(t) \tag{A.34}$$

La velocidad lineal de la rueda delantera se obtiene por medio de la ecuación A.35, se debe tener en cuenta que el *radio* es el de las ruedas.

$$velocidad = (\omega * radio) \tag{A.35}$$

Si se sustituyen los datos presentados en la figura A.3 queda la ecuación

$$v_s(t) = \omega_s(t) * r \tag{A.36}$$

Se calcula el radio de curvatura de la siguiente forma:

$$R = d * tan\left(\frac{\pi}{2} - \alpha(t)\right) \tag{A.37}$$

Sí se sabe que S es la distancia de la curvatura con respecto al ángulo θ entonces:

$$S = d\theta \tag{A.38}$$

Después si se saca la primera derivada a la ecuación A.38 queda:

$$V = d\omega(t) \tag{A.39}$$

Ahora bien hasta el momento se puede decir que la rueda delantera tiene una componente de velocidad (V) y un ángulo $\alpha(t)$ con respecto a la línea horizontal del robot. De forma que la componente V está en el eje perpendicular del robot.

Por lo tanto:

$$V = r\omega(t)\, sen(\alpha(t)) = V_s(t)\, sen(\alpha(t)) \tag{A.40}$$

Si se igualan las ecuaciones A.39 y A.40 se obtiene:

$$V_s(t)\, sen(\alpha(t)) = d\omega(t) \tag{A.41}$$

Ahora se despeja $\omega(t)$ de la ecuación A.41 :

$$\omega(t) = \frac{V_s(t)\, sen(\alpha(t))}{d} = \dot{\theta} \tag{A.42}$$

Entonces la velocidad $v(t)$queda expresada en la ecuación A.43

$$v(t) = V_s cos(\alpha(t)) \tag{A.43}$$

Las ecuaciones que finalmente indican la posición del robot, están expresadas de la siguiente forma:

$$\dot{x_c} = v(t)\, cos(\theta(t)) = V_s cos(\alpha(t))\, cos(\theta(t)) \tag{A.44}$$

$$\dot{y_c} = v(t)\, sen(\theta(t)) = V_s cos(\alpha(t))\, sen(\theta(t)) \tag{A.45}$$

Si se realiza una representación matricial de las ecuaciones A.44, A.45 y A.42 queda de la siguiente forma:

$$\begin{bmatrix} \dot{x_c}(t) \\ \dot{y_c}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} cos(\theta(t)) & 0 \\ sin(\theta(t)) & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} \tag{A.46}$$

De modo que para conocer la posición del robot solo se tiene que integrar la matriz A.46 de la siguiente forma:

$$x_c = \int_0^t v(T)\, cos(\theta(T))\, dT \tag{A.47}$$

$$y_c = \int_0^t v(T)\, sin(\theta(T))\, dT \tag{A.48}$$

$$\theta_c = \int_0^t \omega(T)\, dT \tag{A.49}$$

Cálculos Ackerman

A continuación se presentan las ecuaciones necesarias para obtener la posición del robot con respecto al tiempo.

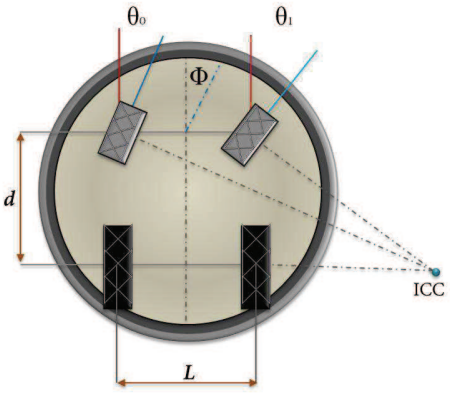


Figura A.4: Sistema de rotación en configuración tipo Ackerman.

La figura A.4 ilustra este tipo de configuración y cómo podemos darnos cuenta los ángulos θ_0 y θ_1 son ligeramente diferentes más específicamente $\theta_0 > \theta_1$, y la razón de la variación entre estos ángulos ayuda a eliminar el deslizamiento de las ruedas, también se puede observar la prolongación de los ejes de cada rueda de forma que todos se intersecan en un punto y este es llamado Centro instantáneo de Rotación (ICR) o Centro Instantáneo de Curvatura (ICC), el cual hemos definido previamente.

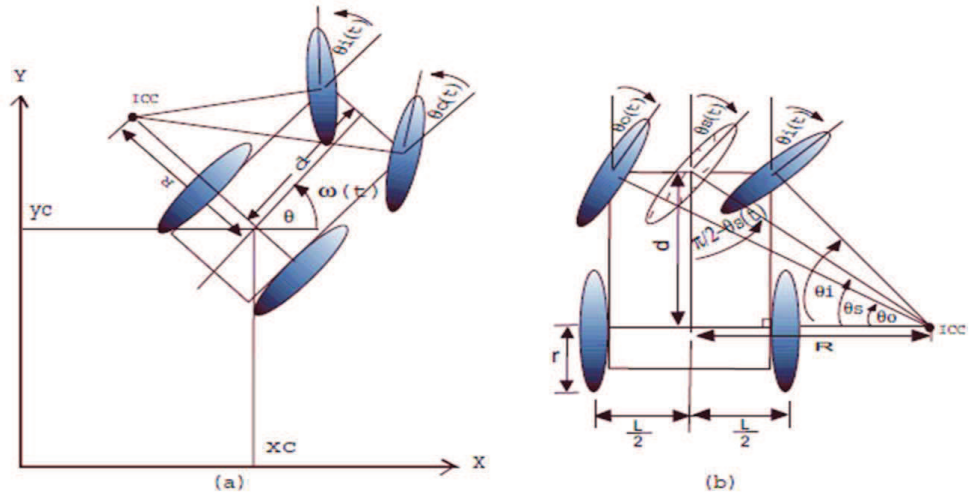


Figura A.5: Configuración de ruedas tipo ackerman considerando los ángulos de dirección.

El la figura A.5 se ilustra la configuración de Ackerman en donde:

L= Es la medición de la separación lateral entre las ruedas.
d= La distancia longitudinal entre las ruedas.
ω= Es la Velocidad Angular del robot.
v= Es la velocidad lineal del robot.
θ= Ángulo de rotación del robot.

r= Es el radio de las ruedas.
ICC= Centro Instantáneo de Curvatura.
R= Es la distancia del radio de curvatura.

Por medio de cálculos cinemáticas se desea determinar:

$$X_c = f_1(t) \tag{A.50}$$

$$X_y = f_2(t) \tag{A.51}$$

Analizando la ecuación anterior se obtiene:

$$cot\left(\theta_i\left(t\right)\right) = \frac{\left(R - \frac{L}{2}\right)}{d} \tag{A.52}$$

$$cot\left(\theta_o\left(t\right)\right) = \frac{\left(R - \frac{L}{2}\right)}{d} \tag{A.53}$$

Igualando R que como se definió es el radio de curvatura en las ecuaciones A.52 y A.53.

$$cot\left(\theta_o\left(t\right)\right) - cot\left(\theta_i\left(t\right)\right) = \frac{L}{d} \tag{A.54}$$

Ya que la configuración Ackerman es una variante de la configuración triciclo, supóngase que existe una rueda imaginaria alineada justo en medio de las dos ruedas delanteras, tal como se muestra en la figura A.5, después se calcula el ángulo de la rueda imaginaria entonces nos queda la ecuación A.55,

$$cot\left(\theta_s\left(t\right)\right) = \frac{R}{d} \tag{A.55}$$

Ahora se despeja R de la ecuación A.55 y después se iguala con las ecuaciones A.52 y A.53, dando como resultado las ecuaciones A.56 y A.57.

$$cot\left(\theta_s\left(t\right)\right) = cot\left(\theta_i\left(t\right)\right) + \frac{L}{2d} \tag{A.56}$$

$$cot\left(\theta_s\left(t\right)\right) = cot\left(\theta_o\left(t\right)\right) - \frac{L}{2d} \tag{A.57}$$

Si se retoma el análisis cinemático realizado en la configuración de triciclo 2.5.2 se puede observar que $\theta_s\left(t\right)$, es equivalente al ángulo de α_s que es presentada en la configuración de triciclo.

Para calcular la velocidad del robot v la velocidad ejercida por las ruedas traseras las cuales son las que ejercen tracción al robot.

$$V_s = \frac{v}{cos\left(\theta_s\left(t\right)\right)} \tag{A.58}$$

Y finalmente para obtener la posición del robot se utilizan las siguientes ecuaciones A.42, A.43, A.44, A.45, A.46 , A.47, A.48y A.49, las cuales habían sido descritas previamente en la sección 2.5.2.

Cálculos Síncrono

La posición del robot móvil de este tipo de configuración está determinada por las ecuaciones A.59, A.60 y A.61.

$$x_c = \int_0^t v\,(T)\,cos\,(\theta\,(T))\,dT \tag{A.59}$$

$$y_c = \int_0^t v\,(T)\,sin\,(\theta\,(T))\,dT \tag{A.60}$$

$$\theta_c = \int_0^t \omega\,(T)\,dT \tag{A.61}$$

Apéndice B

Descripción de los sensores típicamente empleados en la localización

Encoders rotatorios

Los Encoders son sensores que convierten un movimiento rotatorio o lineal en señales digítales. Cuando los encoders trabajan conjuntamente con engranes, ruedas de medición o las flechas de los motores, los encoders pueden ser utilizados para medir movimientos lineales, velocidad y posición. Dependiendo de su tecnología de fabricación los encoders pueden ser magnéticos, mecánicos, electromecánicos, ópticos, entre otros, siendo estos últimos los más empleados.

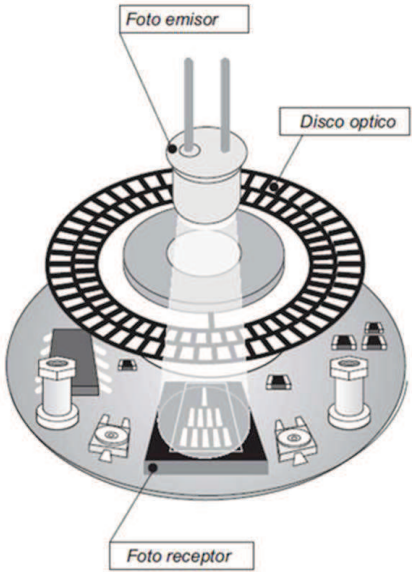


Figura B.1: Encoder Rotatorio capaz de transformar un movimiento angular en una serie de pulsos digítales.

Los encoders rotatorios (véase figura B.1) son muy utilizados en los robots móviles con ruedas, siendo esenciales en el uso de cálculos odométricos. Como se ha explicado en el la sección 2.6.1.1, la odometría es una técnica que permite estimar la posición de un robot móvil con ruedas sí es que se conoce su posición inicial, ya que hace uso del número de revoluciones de cada rueda del robot y la dirección en que se mueven las ruedas.

Con base en la salida de los datos se puede decir que existen dos tipos generales de encoders: los Encoders Incrementales y los Encoders Absolutos.

Encoders Incrementales. Este tipo de encoders entregan un número específico de pulsos por revolución (PPR), por pulgada o milímetro en movimiento lineal, los encoders básicos proporcionan tres salidas de pulsos (Canal A, B y Z), los canales A y B son señales de onda cuadrada defesadas 90° entre sí, con la lectura de un solo canal se obtiene información correspondiente a la velocidad de rotación, pero si se considera la lectura de ambos canales es posible conocer el sentido de rotación con base en la secuencia de datos que son producidos por ambas señales. El canal Z o también llamado Cero, proporciona la posición absoluta del eje cero del encoder, de forma que el número de pulsos acumulados con respecto a esta marca proporciona el desplazamiento.

Encoders Absolutos. Su funcionamiento es básicamente el mismo que los encoders incrementales, la diferencia es que los absolutos generan mensajes digitales que representan la posición actual del encoder, así como su velocidad y dirección de movimiento. Además si la energía se pierde, su salida será corregida cada vez que la energía sea restablecida y no es necesario ir a una posición referencial como los encoders de tipo incremental.

Ultrasónicos

Los sensores ultrasónicos detectan la proximidad de objetos, trabajan a través de la emisión y reflexión de ondas acústicas. El funcionamiento básicamente consiste en la emisión de un pulso ultrasónico, cuando este pulso choca con un objeto entonces el pulso es reflejado, ahora el sensor mide el tiempo desde que se emitió el pulso hasta que el sensor recibe el pulso reflejado; este tiempo se transforma en una distancia y de acuerdo con los parámetros de respuesta envía una señal eléctrica digital o analógica, que puede ser interpretada en unidades de distancia (véase la figura B.2). Estos sensores trabajan solamente en el aire y pueden detectar objetos con diferentes formas, colores, superficies y de diferentes materiales.

En los robots móviles con frecuencia los sensores ultrasónicos son distribuidos de forma que se encuentren separados en intervalos uniformes en la periferia o contorno del robot. Otra estrategia es hacer que un motor este continuamente rotando al sensor, obteniendo así lecturas omnidireccionales.

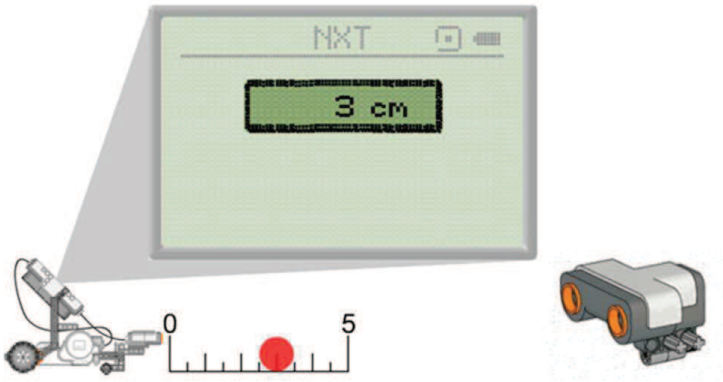


Figura B.2: Sensor ultrasónico NXT, para robot lego Mindstorms NXT, tiene una precisión de +/- 3 centímetros y un alcance de 2.5 metros.

Infrarrojos

Este tipo de sensores son ampliamente utilizados en la robótica móvil para detectar la aproximación de objetos y así evadir los obstáculos, ya que los sensores infrarrojos proporcionan una precisión en el orden de los milímetros, otras ventajas son su bajo costo y su fácil implementación.

Los rayos infrarrojos se encuentran dentro del espectro de los rayos que el ser humano no es capaz de visualizar, pero son visibles para las cámaras; estos sensores básicamente se componen de un emisor y un receptor, el emisor es típicamente un LED emisor de infrarrojos, el cual envía un rayo infrarrojo para posteriormente ser recibido por el receptor que suele ser un fototransistor o un fotodiodo. La señal enviada por el emisor puede ser codificada para distinguirla de otra y así identificar varios sensores a la vez, esto es muy utilizado en la robótica en casos en los que se requiere tener más de un emisor infrarrojo y un solo receptor.

Los sensores infrarrojos son clasificados dependiendo de su modo de operación:

- ***Sensor infrarrojo de barrera.*** El emisor está en línea directa con el receptor, debido a que el modo de operación de esta clase de sensores se basa en la interrupción del haz de luz, la detección no se ve afectada por el color, la textura o el brillo del objeto a detectar. Estos sensores operan de una manera precisa cuando el emisor y el receptor se encuentran alineados. Esto se debe a que la luz emitida siempre tiende a alejarse del centro de la trayectoria.
- ***Sensor auto réflex.*** En este caso el emisor recibe la luz infrarroja reflejada al chocar con un objeto, es decir el emisor envía un rayo de luz infrarroja la cual viaja en línea recta, en el momento en que un objeto se interpone y el haz de luz rebota contra este y cambia de dirección permitiendo que la luz sea enviada al receptor y el elemento sea sensado, un objeto de color negro no es detectado ya que este color absorbe la luz y el sensor no experimenta cambios.
- ***Sensor réflex***. El emisor y receptor están en el mismo encapsulado, el haz de luz se establece mediante la utilización de un reflector catadióptrico. El objeto es detectado cuando el haz formado entre el componente emisor y el componente receptor es interrumpido. Debido a esto, la detección

no es afectada por el color del mismo. La ventaja de las barreras réflex es que el cableado es en un solo lado, a diferencia de las barreras emisor-receptor que es en ambos lados.

Compás o Brújula

El compas magnético o también llamados Brújula, es un magnetómetro que mide el campo magnético de la tierra y es capaz de expresarlo en una señal eléctrica. De forma general suelen ofrecer hasta una precisión de 0.5 grados y su tasa de muestreo puede ser incluso menor a un microsegundo. Son usados en entornos interiores y exteriores controlados, ya que debe considerarse que la señal de salida del compás puede ser fácilmente contaminada por fuentes electromagnéticas (*HardIron Distortion*) por ejemplo cables eléctricos o por grandes estructuras ferromagnéticas (*SoftIron Distortion*) como estanterías metálicas. Debido a lo anterior, el compás frecuentemente es usado como un sensor de apoyo a otros sistemas sensoriales, por ejemplo cuando se usa para compensar el efecto de los derrapes en las ruedas de odometría.

Actualmente la tecnología ha avanzado y hoy en día existen en el mercado brújulas que son capaces de corregir los errores que provienen del robot, además detectan posibles medidas erróneas cuando existe un campo magnético externo o un elemento metálico que pueda influir en la medición.

Los compas magnéticos suelen trabajar de forma conjunta con los inclinómetros para poder estimar la orientación del robot ya que son sensores de posicionamiento de medida absoluta. Un inclinómetro es un dispositivo muy simple y barato que es capaz de medir la orientación del vector gravitacional [61].

Quizás una de las aplicaciones más importantes de este sensor, es cuando se usa en el problema de acumulación de errores odométricos corrigiendo el ángulo de orientación [9], es ahí donde el compás proporciona una forma de corregir dicho error y es la razón por la que este sensor sigue siendo de interés en la comunidad científica.

Básicamente la brújula mide los componentes del campo magnético de la tierra con el propósito de determinar la orientación actual del robot. Uno de los compas electrónicos más usuales en la robótica móvil son los que funcionan con transductores-magneto resistivos, cuya resistencia eléctrica varia con los cambios del campo magnético aplicado, su sensibilidad esta por debajo de 0.1 miliGauss con tiempos de respuesta menor a 1 microsegundo, esta tasa de muestreo permite su uso confiable en robots móviles que se desplazan a altas velocidades.

Algunos trabajos relacionados con el uso de la brújula o compass sensor en inglés, para la localización de robots móviles son [15, 62, 63].

Giroscopios

Existen dos sensores típicamente utilizados para medir la orientación del robot móvil entre las que destacan los compases magnéticos y los giróscopos. Como se vio en la sección anterior (véase en el apartado B), el compas magnético es un magnetómetro, es decir, son sensores que miden orientación con respecto a el campo magnético terrestre. Por otro lado los giróscopos miden la velocidad angular de rotación, en otras palabras miden que tan rápido gira un objeto sobre sí mismo, gracias a esto se puede calcular la velocidad de rotación de un robot móvil en relación a los ejes x, y ó z.

Los giróscopos a menudo son más usados que los compases magnéticos, ya que pueden trabajar sin problemas en entornos con interferencias electromagnéticas y ferromagnéticas, condiciones bajo las que si se ven afectados los compases magnéticos.

Algunas desventajas es que su costo es proporcional a la precisión del sensor por lo que su precio puede llegar a ser elevado.

Dependiendo de sus principios físicos, existen diferentes tipos de giroscopios, a continuación serán descritos a grandes rasgos los giroscopios mecánicos, ópticos y electrónicos.

Giroscopios Mecánicos. Están compuestos por un volante o masa que rota suficientemente rápido alrededor de un eje estando la masa distribuida en la periferia con el objeto de que el momento de inercia del eje de rotación sea alto (véase la figura B.3).

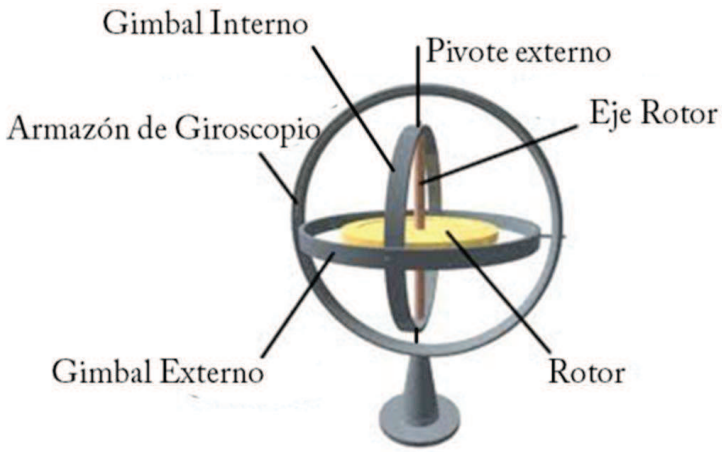


Figura B.3: Componentes del giróscopo mecánico.

La propiedad más interesante de los giróscopos es que si se le aplica un par de fuerzas perpendicular al eje se puede apreciar que la rueda gira, este fenómeno parecería que desafía a las leyes de la gravedad, ya que en lugar de que la rueda tenga un movimiento en su eje como se esperaría, esta adquiere un movimiento lento de rotación, pero alrededor del otro eje perpendicular a él y al eje de giro de la rueda. A este movimiento se le conoce con el nombre de movimiento de precesión y se mantiene mientras existe la inercia giroscópica. Esta propiedad también se puede usar para determinar el ángulo de giro de un objeto en movimiento.

Giroscopios Ópticos. El principio por el que se basa un giróscopo laser es el llamado efecto Sagnac, este tipo de giroscopios presentan algunas ventajas competitivas con respecto a los giróscopos mecánicos, tales como:

- ≻ No es sensible a la gravedad.
- ≻ Posee mayor rango dinámico.
- ≻ Su lectura puede digitalizarse fácilmente.

- ≻ Costo relativamente bajo.
- ≻ Compacto.

Giroscopios Electrónicos. Constituidos por sensores de velocidad angular los cuales se basan en el principio del efecto de Coriolis. Un sensor típico puede tener dimensiones entre 2 y 3 milímetros.

El efecto Coriolis explicado en 1835 por el científico Gaspard-Gustave Coriolis. El efecto básicamente consiste en la existencia de una aceleración relativa del cuerpo en dicho sistema en rotación dicha aceleración es siempre perpendicular al eje de rotación del sistema y a la velocidad del cuerpo. En otras palabras el efecto Coriolis hace que un cuerpo que se mueve alrededor del radio de un disco en rotación, tiende a acelerarse con respecto a ese disco según si el movimiento es hacia el eje de giro o si este se aleja del eje.

Algunos trabajos relacionados con el uso de los giroscopios para la localización de robots móviles son [64, 65, 66].

Acelerómetros

Son dispositivos que detectan los cambios en la velocidad. Existe una gran cantidad de ellos con costos medianamente accesibles, no obstante los más comunes no están preparados para medir velocidad constante, pero proporcionan las mediciones de aceleración o desaceleración.

Los acelerómetros son capaces de detectar desplazamientos del robot, incluso cuando las ruedas del robot están detenidas, ese fenómeno ocurre, por ejemplo en derrapes, en donde las llantas no giran pero existe una variación en el movimiento del robot.

De forma convencional los acelerómetros suelen utilizarse para corregir o minimizar los errores que se ocasionan con el uso de la técnica de estimación de rotación, ya que se complementan con el uso de encoders para incrementar la fiabilidad al calcular el desplazamiento de un robot.

Como ya se mencionó existe una amplia gama de acelerómetros en el mercado, estos dependiendo de la tecnología empleada, varían en características como la precisión, rango de temperatura, sensibilidad al ruido, tamaño, entre otras.

Algunos de los acelerómetros más comunes son los pizo-electrico, pizo-resistivos, térmicos, ópticos, capacitivos. Es importante realizar una cuidadosa selección tomando en cuenta las características de su uso Borenstein [9]presenta un análisis de cada uno, con el fin de realizar la mejor selección posible.

Algunos trabajos de localización de robot móviles que hacen uso de los acelerómetros son [67, 68, 14].

Apéndice C

Tecnologías de comunicación

Sistemas Globales de Posicionamiento por Satélite (GNSS)

Son sistemas que utilizan las coordenadas geográficas terrestres de latitud, longitud y altitud como referencia para la localización dentro del globo terráqueo del robot móvil, se emplean señales que son retransmitidas por satélites geoestacionarios y que mediante técnicas de triangulación se puede localizar un receptor. El receptor montado en el robot tiene la función de captar las señales y de medir el tiempo de vuelo de cada una de ellas y finalmente mediante cálculos de triangulación estimar su posición actual.

Aunque el sistema más conocido de GNSS es el GPS por sus siglas en ingles *"Global Positioning System"* operado por los EEUU, se debe mencionar que existen otros países que también cuentan con sistemas de este tipo, tal es el caso de GLONASS de Rusia, y algunos que todavía están en desarrollo entre los cuales se tiene a GALILEO en Europa y BEIDOU de China.

El GPS (Sistema de Posicionamiento Global en español), fue diseñado por el Departamento de Defensa de los Estados Unidos y era en sus inicios de uso exclusivo del ejercito [9].

El sistema básicamente está constituido por 6 orbitas semicirculares alrededor de la tierra a una latitud de 12,000 millas (19,312.128 km) aprox. con 4 satélites por orbita, haciendo un total de 24 satélites los cuales forman seis planos inclinados a 55° con respecto al plano del ecuador terrestre. Cada satélite transmite de forma continua dos señales de radio en alta frecuencia, que están moduladas con un pseudo-aleatorio binario en las que se codifican de forma compleja información sobre el instante en que la señal fue transmitida, a la cual se le denomina información orbital [69]. El grupo de 24 satélites giran alrededor de la tierra dando una vuelta completa cada 12 horas [70, 71].

La posición del robot móvil se obtiene utilizando técnicas simples de trilateralización basada en el tiempo de vuelo (TOF) de la señal de radio.

El GPS ambicionaba remplazar a los sistemas de localización anteriores, sin embargo al igual que otros sistemas presenta inconvenientes en donde quizás una de sus mayores desventajas son los altos costos de operación, ya que hay que pagar por la utilización de los satélites. Otro problema importante es el hecho de la mala recepción en espacios cerrados de señales GPS con fines de localización, además de la escasa precisión que los datos GPS proporcionan cuando se trata de discernir entre distancias del orden de centímetros.

➢ Las Cadenas de Código (Pozo-Ruz, Ribeiro, García-Alegre, & García, 2000), están formadas por tres tipos de cadenas:

- Código «*Coarse/Acquisition(C/A)*». Con un rango de frecuencia alrededor de los 1.023 MHz y es de uso civil.
- Código *«Precision Code (P)*». Con un rango de frecuencias 10 veces superior al C/A, utilizado por los EEUU para uso militar.
- Código *Y*. Esta cadena es un encriptado del código P cuando está activo el modo de operación antiengaños[1].

Niveles de Servicio El GPS facilita dos niveles de servicios para orden civil (recordar que el uso militar es exclusivo de EEUU)[72].

– Servicio de Posicionamiento Estándar (*SPS, Standard Positioning Service*). Proporciona una precisión normal obtenida con el código C/A.
– Servicio de Posicionamiento Preciso (*PPS, Precise Positioning Service*). Proporciona una mayor precisión extraído del código P de frecuencia dual y su uso requiere autorización.

Actualmente existen sistemas diferenciales de posicionamiento global por satélite (DGPS) que permiten estimar la posición con errores de centímetros.

Los robots móviles y el empleo de la tecnología GPS

Hoy en día existen esfuerzos en el uso del GPS en la localización de robots móviles algunos trabajos relacionados con el uso de esta tecnología son: [73, 74, 75].

Radio de banda ultraancha (UWB-Ultrawideband)
Ultra-wide-band (UWB) [76] es una tecnología de radio que se encuentra en un ancho de banda mayor a los 500 MHz o del 25 % de la frecuencia central, de acuerdo con la FCC (Federal Communications Commission)[2]. Originalmente la tecnología era solo para uso militar (radar/comunicaciones seguras), a partir de 1990 se libera para uso civil.

UWB difiere sustancialmente de las estrechas frecuencias de banda de radio (RF) y tecnologías "spread spectrum" (SS), como el Bluetooth. UWB usa un gran ancho de banda del espectro de RF para transmitir información. Por lo tanto, UWB es capaz de transmitir más información en menos tiempo que las de otro tipo de tecnologías basadas en RF.

➢ Ventajas:

- Mejor capacidad de detección, propagación por interiores, resistencia al fading y penetración en materiales presentes en edificios.
- El emisor de ondas de radio tiene bajo consumo de energía.
- Menor interferencia a la de otros sistemas de RF (baja densidad espectral).

[1] El modo anti-engaños, operativo desde 1994, impide que fuerzas hostiles generen y transmitan una señal igual a la de los satélites GPS.

[2] La FCC por sus siglas en ingles Federal Communications Commission es una agencia estatal independiente de Estados Unidos. La FCC es creada en 1934 junto con la Ley de Comunicaciones, la agencia tiene el propósito de la regulación y censura de telecomunicaciones interestatales e internacionales por radio, televisión, redes inalámbricas, satélite y cable.

- Buena eficiencia para identificar y aproximar la distancia en línea directa entre el robot móvil y la baliza montada en el entorno.
- Puede monitorear más de un robot móvil.
- Velocidad de transmisión de datos.

➢ Desventajas:

- Requiere una sincronización precisa entre las balizas montadas en el entorno y el robot móvil.
- Su buen funcionamiento está limitado para espacios reducidos.

Robots Móviles y la tecnología UWB.
Actualmente existen esfuerzos en el uso de el UWB para la localización de robots móviles [77, 78, 79][77, 78, 79].

Basados en Tecnología RFID

RFID (*Radio Frequency Identification*, en español *Identificación por radiofrecuencia*), su fecha de origen es dudosa, ya que se ha sugerido que está íntimamente relacionada con la Segunda Guerra Mundial, sin embargo recientemente ha surgido como una tecnología usada para la localización de objetos, entre otros usos.

Básicamente es un sistema de almacenamiento y recuperación de datos inalámbricos, para ello emplea dispositivos llamados etiquetas o tags RFID en las que reside la información, estas etiquetas RFID tienen montadas antenas, con la finalidad de permitirles recibir y responder peticiones en un sistema emisor-receptor RFID (también conocido como lector RFID), las etiquetas usan un chip con capacidad de almacenamiento de datos, los cuales son guardados con un identificador único. El principio de funcionamiento es parecido a los Sistemas de Código de Barras que se pueden encontrar en los productos de un centro comercial, la diferencia reside en que el código de barras es leído por medios ópticos mientras que la tecnología de RFID emplea señales de radiofrecuencia para la trasferencia de los datos.

El RFID es una tecnología que emplea señales de radiofrecuencia en diferentes bandas dependiendo del tipo de sistema (véase la tabla C.1), típicamente los rango de frecuencia son 125 KHz, 13,56 MHz, 433-860-960 MHz y 2,45 GHz. Las clasificaciones más comunes de esta tecnología son:

Según el modo de alimentación:

➢ Pasivos. si las etiquetas no necesitan batería.

- Sin batería propia (la energía la proporciona la señal del lector).
- Menor rango de acción (centímetros).
- Mayor vida útil (años).

Tabla C.1: Según el rango de frecuencia de trabajo [80].

País/Región	***Low-Frequency*** (LF)	***High-Frequency*** (HF)	***Ultra-High-Frequency*** (UHF)	Microondas
USA	125-134 KHz	13.56 MHz	902-928 MHz	2400-2483.5 MHz 5725-5850 MHz
Europa	125-134 KHz	13.56 MHz	865-868 MHz	2.45 GHz
Japón	125-134 KHz	13.56 MHz	No permitida	2.45 GHz
China	125-134 KHz	13.56 MHz	No permitida	2446-2454 MHz

- Menor tamaño.

≻ Activos. Si las etiquetas requieren de una batería para transmitir la información.

- Batería incluida.
- Mayor rango de acción (kilómetros).
- Mayor confiabilidad (menos errores en la transmisión).
- Vida útil limitada (meses).

≻ Semi-pasivas

- Batería incluida para el microprocesador (la energía para la transmisión la proporciona la señal del lector).
- Mayor sensibilidad a la señal que viene del lector pero el rango de transmisión es limitada como en la etiqueta pasiva.
- La batería dura más que en la etiqueta activa.
- Actividad autónoma del microprocesador para realizar otras funciones.

Robots Móviles y la tecnología RFID.
Actualmente existen esfuerzos en el uso del RFID para la localización de robots móviles [81, 82, 83, 84, 85].

Apéndice D

Utilizando MRDS

A groso modo la idea fundamental de MRDS, es ver a cada dispositivo que integra al robot como un servicio distribuido y asíncrono, para ejemplificar pensemos en un sensor el cual en MRDS será visto como un servicio que proporciona al robot información y un motor entonces es un servicio de respuesta del robot, de forma que un servicio de control es el encargado de interpretar la información del servicio de entrada del sensor y determinar las acciones pertinentes enviadas al servicio de respuesta que representa el motor.

Mientras que la coordinación se ve reflejada en un tipo de comunicación asíncrona entre todos estos servicios. De manera que en un programa anti-choques es un servicio de entrada (sensor), detecta un obstáculo enviando un mensaje al servicio del controlador de dicho evento, el cual a su vez envía otro mensaje al servicio de respuesta para que las ruedas realicen las maniobras oportunas. Todo este escenario incrementa su nivel de complejidad cuando se tienen un conjunto de sensores que pueden enviar de forma simultánea información al servicio de control, y el servicio de control tendrá que emitir de forma simultánea las indicaciones necesarias a los servicios de respuesta. Es justo ahí donde los servicios de coordinación son los encargados de manejar todo este tráfico de información en tiempo real, siendo estos lo encargados de aplicar políticas complejas de control, para la correcta administración de los diferentes servicios.

El tiempo de ejecución de MRDS consiste en dos tiempos de ejecución de menor nivel que se basan en CLR 2.0. Estos dos tiempos de ejecución son los servicios de software descentralizado (DSS) y el módulo de tiempo de ejecución de simultaneidad y coordinación (CCR). DSS es un módulo de tiempo de ejecución ligero y orientado al servicio, basado en los principios de Transferencia de estado de representación (REST) que se usan para aumentar la eficacia de la Web. CCR es una biblioteca de Microsoft .NET Framework compatible con el procesamiento asincrónico. Esto tiene una importancia vital para las aplicaciones de robótica, ya que hay numerosos sensores y actuadores que están continuamente enviando y recibiendo datos.

La concurrencia y coordinación de la ejecución CCR

Cuando se programa un robot es necesario controlar una serie de dispositivos que pueden entrar en funcionamiento simultáneamente, por lo cual el programador está obligado a implementar las políticas de coordinación y sincronización entre los distintos hilos de cada servicio en ejecución. Sin embargo en MRDS ha diseñado mecanismos que le permiten al programador abstraerse de los procesos internos inmiscuidos en la coordinación y sincronización de servicios, por lo tanto MRDS provee un mecanismo para el control de los distintos hilos de ejecución de una forma más eficiente que los tradicionales threads

(hilos) de Windows y mucho más transparente para el programador. Para ello MRDS ha integrado herramientas que nos permiten multitud de servicios ejecutados al mismo tiempo. La concurrencia y coordinación de la ejecución o en inglés "Concurrency and Coordination Runtim (CCR)", es la encargada de la administración del tráfico de mensajes, de manera que si no existiera CCR en esta plataforma se tendrá que hacer de forma manual la administración de recursos con bloqueos, semáforos, prioridades, entre otras estrategias. El CCR tiene el objetivo de controlar el paso de mensajes de forma asíncrona y proveer una directriz a los tiempos de ejecución de los diferentes servicios activos.

El CCR permite la coordinación concurrente y asíncrona del flujo de ejecución abstrayendo al programador del uso de hilos, semáforos y otras técnicas de más bajo nivel para el aseguramiento de la exclusión mutua o la prevención del interbloqueo. Además plantea un modelo de programación asíncrona que facilita y optimiza la explotación de un entorno de ejecución paralelo o multihilo. Cabe destacar que el CCR es un componente DLL que se ejecuta en el entorno .NET y accesible desde cualquiera de los lenguajes de programación disponibles en .NET.

Entonces la biblioteca de CCR es una DLL que ofrece una serie de clases que permiten a los desarrolladores un modelo de objetos simples que se encargan de las operaciones de E/S, además presenta un alto rendimiento en la administración de hilos dedicados a servicios de respuesta I/O, brindando alta escalabilidad y optimizando la concurrencia en la aplicación.

Básicamente el CCR está compuesto de tres componentes que son claves para comprender como es que funciona:

- ≻ Los puertos (clases Port y PortSet).
- ≻ Las primitivas de coordinación o bien árbitros y receptores (clase Arbiter).
- ≻ Las tareas, colas de ejecución y dispensadores (clases Dispatcher, DispatcherQueue y Task).

Para comenzar con la descripción de las clases dedicadas a los puertos, es necesario precisar que cuando se habla de «elementos» se refiere un servicio que hace uso de un puerto, mientras que los mensajes son peticiones que llegan al puerto encapsulados dentro de determinadas clases, que provienen de otros servicios asociados, y las respuestas generadas por el servicio de respuesta a dichas peticiones.

Los mecanismos de administración de los puertos están regidos por la clase genérica Port, el cual es en esencia una cola FIFO (*First-in, First-out*) de mensajes, de manera que una instancia de esta clase solo es capaz de recibir mensajes de un determinado tipo, el cual es previamente especificado durante la instalación.

Ahora bien para realizar envíos que provienen de distintos tipos de mensajes a través de un mismo puerto principal es necesario crear una clase PortSet, la cual implementa un agregado de colas de distinto tipo que son tratadas como un tipo de entidad única. Entonces podemos decir que la clase PortSet, tiene la misma funcionalidad de Port, pero con la diferencia de que acepta varias clases de elementos, es un puerto que representa diferentes tipos de datos y puede recibir cualquiera de ellos. Habitualmente se crea un PortSet que soporta n tipos diferentes y es capaz de coordinar los diferentes mensajes independientemente.

De manera general existen dos maneras de crear una instancia de tipo PortSet:

1. Usando argumentos genéricos que son definidos durante el tiempo de compilación, tales como el número y el tipo de puerto (port<>). Los CCR soportan durante la utilización de PortSet un

total de veinte argumentos de tipo genérico en el CLR[1] de escritorio, así como ocho argumentos genéricos en .NET Compact Framework.

Ejemplo

```
//Lenguaje de programación C#
// Creación de un PortSet con argumentos de tipo vargenericPortSet = new PortSet<int, string, double>();
genericPortSet.Post(10);
genericPortSet.Post("Hola mundo");
genericPortSet.Post(3.14159);
```

1. La segunda forma es inicializando un constructor el cual se integra de una lista de parámetros de tipo argumento. El usuario puede implementar un árbitro de tipos y la instancia port< > será creado en la ejecución.

2. Debido a que la plataforma de MRDS realiza de forma asíncrona el envió y recepción de mensajes que envían los diferentes servicios, para ello CCR nos brinda la posibilidad del uso de árbitros (Arbiters en inglés), los cuales ejecutan otras partes de código mientras se espera que llegue algún mensaje de algún puerto.

Ejemplo

```
//Lenguaje de programación C#

// Crear un PortSet entiempo de ejecución, usando la inicialización de un

// Constructor para proporcionar una matriz de tipos PortSetruntimePortSet = new PortSet(

typeof(int),

typeof(string),

typeof(double) );

runtimePortSet.PostUnknownType(10); runtimePortSet.PostUnknownType("Holamundo");
runtimePortSet.PostUnknownType(3.14159);
```

Las primitivas de coordinación o bien árbitros y receptores

Los árbitros son los encargados de coordinar un conjunto de receptores los cuales se encuentran alerta de un determinado puerto, con el propósito de esperar un determinado tipo de mensaje. Una vez que llega algún mensaje el receptor encargado de su guardia lo captura y después el árbitro correspondiente indica que acciones se deben llevar a cabo.

Un árbitro es capaz de realizar complejas operaciones producto de la combinación de mensajes capturados por los distintos receptores que coordinan.

[1] ThreadPool (CLR) Son servicios que hacen uso del paralelismo de la CPU, por lo que aprovechan las ventajas de las arquitecturas de multinúcleo, su objetivo es distribuir de forma rápida y óptima el trabajo, haciendo uso de colas de libre de bloqueo evitando la contención y robo de trabajo para el equilibrio de carga (para obtener más información véase msdn.microsoft.com/magazine/cc163340).

Para trabajar con las primitivas de coordinación se usa la clase Arbiter que proporciona diversos métodos estáticos como los siguientes:

Arbiter.FromTask -->Crea instancia de trabajo.
Arbiter.Choice --> Crea instancia de elección.
Arbiter.Receive --> Crea instancia de receptor.
Arbiter.Interleave --> Crea instancia de Interleave.
Arbiter.JoinedReceive -->Crea instancia de Join-Receiver.
Arbiter.MultipleItemReceive --> Crea un puerto simple de recepción.

Tareas, colas de ejecución y dispensadores

Las tareas son un mecanismo empleado para equilibrar la distribución de los recursos en uso, es decir las tareas se generan cuando los mensajes llegan a los puertos. Después son recibidos por los receptores y piden hacer uso de los recursos de los que dispone el robot para ser ejecutadas. Ahí es donde se necesita la creación de tareas que sean capaces de regularizar y equilibrar el uso de tales recursos. Estas tareas pueden ser de tres tipos diferentes.

La clase tarea. Incluye la implementación de la interface ITask, Task e IterativeTask. La clase tarea permite encapsular una serie de funciones para posteriormente asignarle un lugar en una cola de ejecución mientras espera para ser ejecutada o atendida.

La clase dispensadores. O también llamados Dispatchers en inglés, los cuales están encargados de asignar los hilos de ejecución cuando es necesario que los mensajes sean atendidos de forma simultánea, en otras palabras un Dispensador es un distribuidor que administra los subprocesos del sistema operativo y tiene como objetivo equilibrar la carga de las tareas de cola de uno o más casos DispatcherQueue.

Ejemplo de Dispatchers

```
//Lenguaje de programación C#
public sealed class Dispatcher : IDisposable {
public Dispatcher();
public Dispatcher(Int32 threadCount, String threadPoolName);
public Dispatcher(Int32 threadCount, ThreadPriority priority, String threadPoolName); publicICollection<DispatcherQueue>DispatcherQueues { get; }
...
// Fin de la creación de dispatchers
```

La clase DispatcherQueue. DispatcherQueue implementa una cola FIFO de tareas (Task), para ello hace uso de delegados que identifican los métodos que se pueden ejecutar según la secuencia de la cola de tareas tipo FIFO. Entonces el DispatcherQueue es el encargado de seguir las políticas de FIFO para enviar al despachador el servicio que sigue para ser ejecutado y para que posteriormente el despachador le asigne los hilos correspondientes para su correcta ejecución.

Ejemplo de DispatcherQueue

```
public sealed class DispatcherQueue : IDisposable {
// Haciendo uso de CLR y no de un Dispatcher

publicDispatcherQueue(string name);
publicDispatcherQueue(String name, Dispatcher dispatcher);

public virtual void Enqueue(ITask task);
public virtual void EnqueueTimer(
TimeSpantimeSpan, Port<DateTime>timerPort);
publicvoidDispose();

// Fin de la creación de DispatcherQueue
```

Servicios de Software Descentralizados DSS

Al realizar la programación necesaria para el control de los diferentes dispositivos que integran un robot, se debe considerar la monitorización y manipulación de múltiples procesos o servicios, donde en la mayoría de las veces suelen requerir su ejecución de forma concurrente. Para ello MRDS hace uso de los Servicios de Software Descentralizados DSS (en inglés Decentralized Software Services) los cuales son un componente clave, ya que gracias a los servicios es posible el control de los diferentes sensores y actuadores que constituyen a un robot. Para comenzar a entender la importancia de DSS en el manejo de componentes primero se define lo que es un servicio para después describir cómo es que se hace uso de estos servicios durante el control y manejo de su ejecución.

Servicio

Un servicio en MRDS es el elemento más importante, ya que un servicio puede verse como una representación abstracta de diferentes elementos de hardware (sensores, motores) y dichos servicios se ejecutan dentro de un contexto de nodos DSS.

Un nodo DSS es un lugar en donde se hospedan los servicios con el fin de proporcionar soporte a los servicios, ya que en un nodo DSS se pueden crear, manejar y eliminar servicios además los servicios se pueden comunicar independientemente de si son ejecutados dentro del mismo nodo o a través de la red. De forma que una vez que un servicio es creado dentro de un nodo DSS, se le asigna de forma automática un número identificador, el cual permite ser identificado por otros nodos provenientes creados por el mismo nodo DSS y así poder comunicarse entre ellos.

Como ya se ha mencionado los servicios tienen un número de identificación, pero también se le asigna un tipo de estado es decir, los parámetros que genera un servicio. Por ejemplo considérese que el estado de un servicio para un motor puede estar representado por las rotaciones por minuto (RPM), la temperatura, la potencia utilizada, etc. Mientras que en un servicio de un teclado, su estado puede estar representado por las pulsaciones por minuto.

Ahora bien para aprovechar el conjunto de información proveniente del estado de diversos servicios, es necesario que estos puedan ser capaces de identificarse y después comunicarse, para ello se hace uso de servicios asociados o partner. Si se considera la utilización de los servicios asociados, en realidad nos referimos a las relaciones existentes entre cada uno de los elementos que forman parte de la asociación, dicho en otras palabras podemos dar la instrucción de que si un servicio no encuentra a otro servicio

registrado bajo un determinado número de identificación en un parámetro de tiempo prestablecido, entonces no se ejecutara el servicio buscador; o bien se puede dar la instrucción de que el servicio buscador se ejecuto no importando si encuentra al otro servicio que está buscando.

La plataforma de MRDS nos proporciona un conjunto de servicios que cuentan con programación predefinida, lo cual pretende facilitar el uso de los servicios. Sin embargo debe tenerse en consideración que al ser generalizados también presentan algunas limitaciones, por lo que en muchas ocasiones es mejor diseñar servicios propios.

En la última versión del programa, la 1.5, una de las novedades, es que se incluye una serie de servicios que proporcionan un acceso completo a todas las funcionalidades de los LEGO MINDSTORMS NXT. Esto demuestra que la plantilla de servicios predefinidos, irá creciendo en futuras versiones en función de los robots del mercado y la expansión de la robótica doméstica.

El modelo de servicio DSS ha sido diseñado para facilitar la reutilización de los servicios. Todos los servicios de DSS consisten en un conjunto común de componentes tal como se describe en esta sección.

La arquitectura de los servicios DSS pretende facilitar la reutilización de los servicios, proporcionando una programación más eficiente. Los principales componentes de un servicio pueden verse en la siguiente lista.

- ***ServiceIdentifier (Identificador de Servicio).*** Proporciona un número que permite identificarse y comunicarse con otros servicios que provienen del mismo nodo DSS e incluso que permite a un navegador Web acceder a él.
- ***ContractIdentifier (Identificador de Contrato).*** Se trata de una descripción única del comportamiento del servicio. En este comportamiento se predefine el tipo de operaciones que el servicio es capaz de realizar, así como la definición del tipo de datos que necesita dicho servicio para ser ejecutado de forma correcta.
- ***ServiceState (Estado del Servicio).*** Contiene el estado del servicio en determinado momento. Está integrado por la información que caracteriza al servicio, la información que proviene de dicho servicio puede ser accesible, modificable y/o monitorizable durante su ejecución.
- ***ServicePartners (Servicios Asociados).*** Ya que es muy raro que un servicio sea ejecutado como único servicio, es necesario un servicio que pueda realizar la interacción entre los diferentes servicios en ejecución. Entonces cada uno de los servicios en un inicio busca una conexión con cada uno de los servicios socios y pueden definirse distintos comportamientos dependiendo de si es posible realizar dicha conexión o no.
- ***Main Port (Puerto Principal).*** Es un puerto CCR donde llegan mensajes de otros servicios. Este puerto es un miembro privado de la clase Service y se identifica con el atributo ServicePort.
- ***ServiceHandlers (Manejadores de Servicio).*** Se encarga de registrar cada una de las operaciones que es capaz de soportar un servicio, para ello debe haber un manejador, el cual se encarga de ejecutar una tarea que proviene del puerto principal.
- ***Notificaciones (Notifications).*** Se trata de mensajes de aviso emitidos por el servicio, proporcionando información con respecto a los cambios en su estado.

Contratos

Los contratos son elementos que permiten al programador manejar fácilmente los procesos que tienen directa relación con el hardware. Mediante el uso de un contrato genérico, el servicio no necesitará saber de entrada a qué hardware estará conectado.

Los servicios envían mensajes el uno al otro mediante puertos. Los servicios saben cómo comunicarse el uno con el otro porque tienen contratos. El contrato contiene toda la información sobre el formato de éstos mensajes y qué soporta el servicio. Todo esto se especifica en un esquema definido en un fichero XML. El nodo DSS se encarga de coordinar toda esta actividad.

Como ejemplos considérese el drive contract implementado por medio del servicio Arcos Drive, este contrato sirve para controlar un objeto, el objeto es un robot basado en la plataforma Mobile Robots Pioneer 3-DX. Este contrato tiene operaciones para seleccionar las velocidades de las ruedas derecha e izquierda del robot independientemente, rotar el objeto un ángulo determinado y desplazarlo una distancia determinada. El servicio de este dispositivo se encarga de mandar una notificación cuando cambia su estado, esto incluye la velocidad de las ruedas y si el control está habilitado.

Introducción al lenguaje de VPL (Visual Programming Language)

Uno de los mayores motivos por el que se creó Robotics Studio era acelerar el desarrollo y la adopción de la robótica, gran parte de este esfuerzo está centrado en la simulación.

El kit de herramientas de MRDS incluye una herramienta VPL que permite la creación de aplicaciones de robótica con tan solo arrastrar los elementos a una superficie de diseño. MRDS también incluye un VSE que le permite experimentar con simulaciones complejas que impliquen a varios robots y obstáculos.

El entorno de simulación está basado en XNA lo cual asegura una calidad, escalabilidad y visualización, que permite a cualquier persona usarlo como si fuera un juego. Un programa en VPL consiste en bloques, que generalmente llamaremos “actividad”. Un flujo de datos consiste en una secuencia de actividades conectadas. Las actividades en VPL pueden representar diversas funciones: un elemento que controla flujo de datos, una función, una sección de código creado por el usuario, etc. Se escriben programas en VPL conectando un conjunto de actividades.

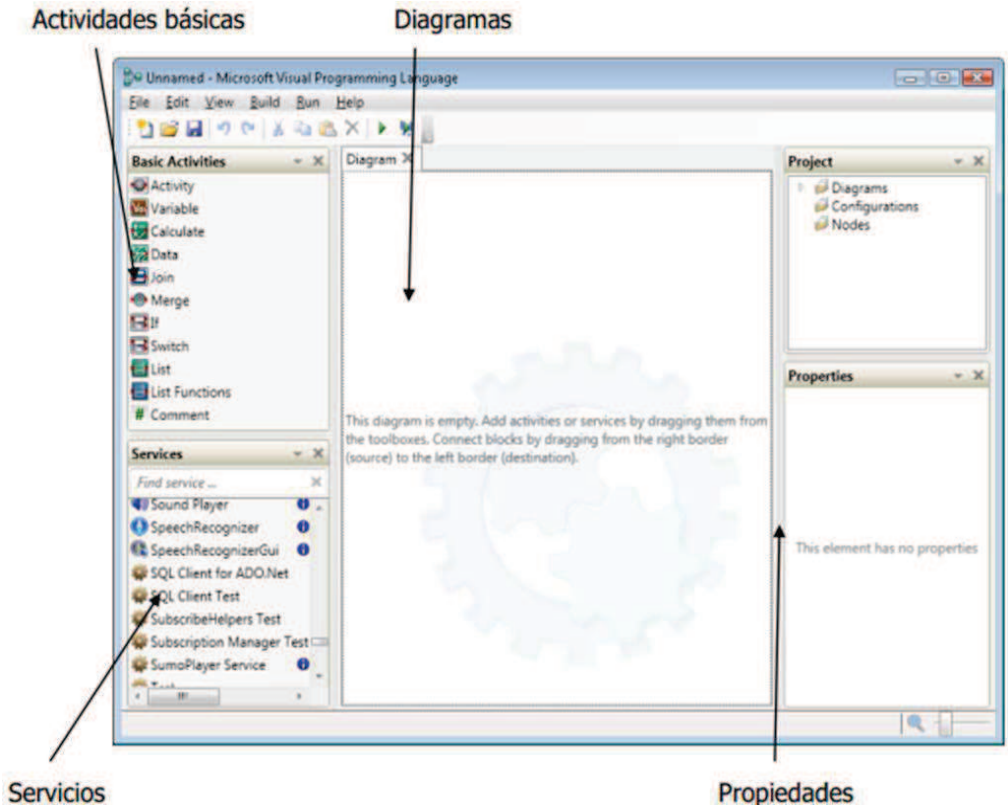


Figura D.1: Ejemplo de entorno de programación a bloques en VPL.

En VPL, las actividades conectadas transfieren datos o información entre sus conexiones, como se muestra a continuación.

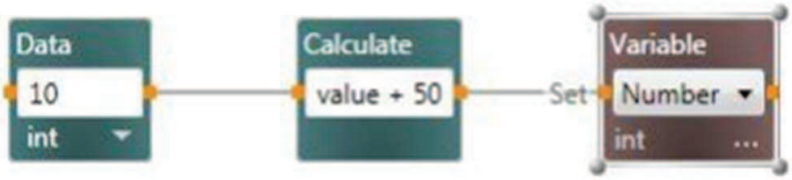


Figura D.2: Ejemplo de bloques de programación en VPL.

Actividades básicas: la ventana de actividades básicas incluye bloques de actividades que realizan las siguientes funciones:

- **Data.** Se aplica para brindar un valor a otra actividad o servicio.
- **If.** Permite seleccionar alternativas de salida a determinada condición aplicada sobre la variable de entrada.

- **Join**. Combina mensajes de dos o más entradas. La diferencia de Merge es que las variables de entrada a combinar deben estar todas activas antes de devolverse el resultado.
- **List.** Se utiliza para crear una lista.
- **ListFunctions.** Se utiliza para modificar una lista existente.
- **Merge.** Combina las variables de entrada sin ninguna condición.
- **Switch.** Se selecciona la salida si la variable de entrada cumple cierta condición.
- **Variable.** Se define una variable.
- **Comment.** Permite colocar bloques de comentario en el diagrama.

Especificaciones del robot y sensores empleados en el experimento 2

En el experimento 2 descrito propiamente en la sección 4.2, es necesario la selección de un robot y de los sensores que lo integran, ya que se pretende realizar una simulación en MRDS, en donde se implementen las memorias asociativas Alfa-Beta. Razón por la que esta sección esta encaminada en la descripción del robot y los sensores utilizados durante la fase de experimentación.

Robot empleado en la fase de experimentación

Es importante considerar que el robot sea compatible con la herramienta de simulación MRDS por lo que se ha seleccionado al robot lego Mindstorm NXT, debido a una serie de ventajas pero sobre todo porque es capaz de procesar datos del sensor tipo brújula que robots de otros tipos no contienen. A continuación se explicara más sobre las cualidades del robot lego Mindstorm NXT.

El kit de Lego Mindstorm ha demostrado ser una poderosa, práctica y potente herramienta para estudiantes profesores e investigadores al momento de desarrollar y crear prototipos, ya que el robot puede armarse bajo diferentes modelos, lo que permite versatilidad en la forma de los robot dependiendo de las necesidades del usuario. Lego Mindstorms es una línea del conjunto de productos para robótica de Lego. Combina dispositivos programables con motores eléctricos, sensores, piezas plásticas. La construcción es por medio de ensamble de piezas en las que integra desde las barras comunes hasta piezas de alta complejidad mecánica.

La primera versión comercial de Lego Mindstorms fue liberada en 1998 y conocida comercialmente como Robotic Invention System (por sus siglas en ingles RIS). La versión actual fue liberada en 2006 como Lego Mindstorms NXT. Además trabaja bajo múltiples lenguajes y entornos de programación, libros, campeonatos mundiales de robótica y ha sido usado en universidades como MIT, UBC, Instituto RWTH, entre otras.

Lego Mindstorms NXT remplaza la primera generación del kit Lego Mindstorms (RIS). Viene originalmente con un software de programación llamado NTX-G. Lego Mindstorms NXT puede ser programado en una gran variedad de lenguajes tales como C, C++, Java, Assembler, NXC, NBC, RobotC, BricxCC, MpLab, MatLab, VPL de MRDS, entre otros. Cabe mencionar que la mayoría son libres y multiplataforma.

Especificaciones técnicas:

- ≻ Microcontrolador de 32-bit AT91SAM7S256.
- ≻ 256 Kbytes FLASH, 64 Kbytes RAM.
- ≻ Microcontrolador AVR de 8-bit.
- ≻ 4 Kbytes FLASH, 512 Byte RAM.
- ≻ Bluetooth.
- ≻ Puerto USB (12 Mbit/s).
- ≻ 4 puertos de entrada, cable 6-vias digital platform.
- ≻ 3 puertos de salida, cable 6-vias digital platform.
- ≻ 100 x 64 pixel LCD graphicaldisplay.

Figura D.3: Lego Mindstorms NXT. La nueva generación del Brick Lego Programable, el NXT LEGO y algunos perifericos externos.

El cerebro de Mindstorm NXT es una pequeña computadora en forma de un pequeño ladrillo llamada comúnmente NXT Brick. Este dispositivo está compuesto por cuatro entradas para sensores y es capaz de controlar hasta tres motores, por medio de cables RJ12. El brick tiene una pantalla LCD monocromática de 100x64 píxeles y cuatro botones que pueden ser usados para navegar en una interface de usuario utilizando menús jerárquicos. También cuenta con bocinas que permiten reproducir archivos de sonido en frecuencias de 8kHz. La energía la obtiene de 6 baterías AA (1.5V cada una).

Sensores empleados en la fase de experimentación

Mindstorm NXT es capaz de soportar diferentes tipos de sensores desde los sencillos como sensores, de tacto, de luz y de color hasta sensores como giroscopios, acelerómetros, barómetros, sensor compas ó brújula digital, entre muchos otros.

NXT Sensor brújula (Compass Sensor NMC1034)

Compass sensor o sensor brújula mide el campo magnético de la tierra y devuelve un ángulo de 0 a 359 grados, siendo cero la orientación exacta al norte. La dirección a la que está apuntando actualmente es calculada al grado discreto más cercano y se actualiza 100 veces por segundo. La Brújula se conecta a través del puerto de sensores que viene equipado con el NXT y utiliza el protocolo de comunicación digital I2C. Es importante mencionar que este sensor no es desarrollado por Lego, sino por HiTechnic; no obstante todos los sensores desarrollados por esta compañía son 100 % compatibles con el lego Mindstorms NXT.

Figura D.4: HiTechnic NXT Sensor brújula (Compass Sensor NMC1034).

NXT Acelerómetro (Acceleration Sensor NAC1040)

El acelerómetro HiTechnic/Sensor permite saber si el robot se ha desplazado en los tres ejes dimensionales (x,y y z), por lo que suele emplearse para determinar si el robot ha sufrido alguna inclinación. Este sensor es capaz de medir la aceleración de su robot bajo un rango de precisión de -2G a +2G. Ideal para experimentar con las fuerzas de aceleración. La medida de aceleración de cada uno de los ejes se refresca unas 100 veces por segundo (Mismo encapsulado que la brújula, véase figura D.4).

NXT Ultrasónico (Ultrasonic Sensor)

Es un sensor que puede detectar objetos que estén entre 3 y 255 cm de distancia. No solo los detecta, sino que también puede determinar la distancia a la que se encuentran con un error máximo de ± 3 cm. Al contrario que el resto de sensores, no devuelve los valores en ninguna escala ni porcentaje, sino en unidades reales, bien centímetros, bien pulgadas. El alcance de medición comprende desde 3 cm hasta 255 cm (tiene una zona muerta en distancias muy cortas), y su funcionamiento por ultrasónicos a 40 KHz lo hace altamente efectivo para escenas relativamente simples y con objetos bien definidos,

mientras que si aparecen elementos pequeños o en grandes cantidades, sus mediciones pueden no ser tan buenas.

Figura D.5: NXT Sensor Ultrasonico (Ultrasonic Sensor).