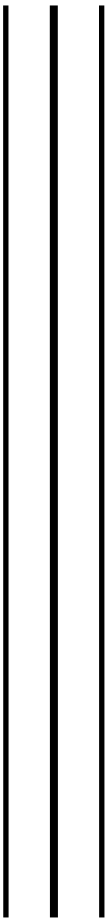
UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO



FACULTAD DE INGENIERÍA

NAVEGACIÓN DE UN ROBOT MÓVIL APLICANDO CAMPOS POTENCIALES Y RECONOCIMIENTO DE OBJETOS USANDO OS ANDROID.

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO ELÉCTRICO Y ELÉCTRONICO

P R E S E N T A N

BUSTOS DE LA CRUZ NERY HERIBERTO

GODÍNEZ GARCÍA JULIO CÉSAR

DIRECTOR DE TESIS:

ING. ROMÁN VICTORIANO OSORIO COMPARÁN

MÉXICO D.F. **FEBRERO, 2016**

Dedicatoria.

A mi Sobrina Sinaí que me inspira para seguir adelante.

Nery Heriberto Bustos De La Cruz.

A José y Bernarda, quienes con amor, paciencia y sabiduría han guiado mis pasos desde el día en que llegué a este mundo.

A José y Juan, quienes han sido mi ejemplo, mis cómplices, mis compañeros y a veces hasta mis verdugos.

¡Los amo!

Julio César Godínez García

Agradecimientos

Estoy seguro que esta sección será la primera en donde el lector se detendrá, irónicamente estos párrafos los escribí en el último momento, esto se debe a que no encontraba las letras adecuadas para plasmar las emociones hacia las personas que me apoyaron a lo largo de mi carrera. Trataré de hacer mi mejor esfuerzo, así que, a lo que vamos:

Deseo agradecer a mi familia: a mi padre Adrian por el apoyo en todos los aspectos posibles. A mi madre Nolberta por estar siempre a mi lado y por todo el cariño que me brinda, a mis hermanas Adriana y Samara por su compañía y comprensión, pero principalmente agradezco a mi sobrina Sinaí ya que su llegada cambió positivamente la vida de todos los integrantes de mi familia y ella representa una fuerza motriz en todos los aspectos de mi vida.

También quiero agradecer a todos los amigos, amigas, compañeros y compañeras que conocí a lo largo de mi formación profesional. Si cito a todos y cada uno de ellos la lista sería un poco larga, puesto que estas personas las he ido conocido a lo largo de varios recintos de estudios y centros de convivencia, así que esta parte es cuasi general: agradezco a todas y cada una de estas personas, ya que me han brindado apoyo moral y momentos inolvidables a lo largo de esta parte de mi vida, pero agradezco especialmente, a mis amigos Oswaldo y Julio , este último por haberme apoyado en varios aspectos tanto académicos, morales y personales, además de haber aportado ideas y técnicas de gran relevancia para este trabajo.

Agradezco a la Universidad Nacional Autónoma de México y a la Facultad de Ingeniería puesto que considero que la formación social y profesional que he recibido en esta institución ha sido excelente. También agradezco al Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas, principalmente al Ingeniero Román, al Doctor Mario y al Maestro Humberto por su apoyo académico y por brindarme la oportunidad de desarrollarme profesionalmente.

Nery Heriberto Bustos De La Cruz.

Agradecimientos

Quiero agradecer en primer lugar a mis padres y hermanos, quienes han sido pieza clave en mi realización como ingeniero, porque sin ellos no sería ni un atisbo de lo que hoy sí.

En segundo lugar, a mi querida UNAM, pero sin olvidar a mi preciada FI; que cobijó mi ignorancia obsequiándome conocimiento.

A todo el equipo en el DISCA del IIMAS, en particular al Ing. Román Osorio por la oportunidad, el apoyo y la asesoría.

A Gaby, por el apoyo permanente e incondicional.

A Joel, por su compañía, amistad y por la disposición para perfeccionarme en lo académico.

A Nery, por su colaboración en la realización de este trabajo y por todas las aventuras que implicaron.

Pero en especial a mí, por haber continuado hasta llegar a esta meta.

Julio César Godínez García

ÍNDICE GENERAL.

CAPÍTULO 1

INTRODUCCIÓN.

1.1 Historia de la robótica.

Para el ser humano la idea de sobrepasar sus límites, mejorar y superarse, siempre ha sido primordial. Desde el momento en el que comenzó a ser sedentario, dejó el comportamiento salvaje y comenzó con el empleo de herramientas que le ayudaran, y así mismo, facilitaran sus tareas cotidianas. De este modo el ingenio pasó a primer plano para en primera instancia lograr sobrevivir y posteriormente para hacer la vida más placentera. El hombre desarrolló herramientas y artefactos para llevar a cabo deberes que por sí mismo no hubiera sido capaz de realizar. Con el paso del tiempo había cada vez más exigencias y era imprescindible innovar, por esta razón, estos artefactos tenían que ser cada vez más sofisticados, tal complejidad dio pauta al ser humano para imaginar artefactos a su imagen y semejanza; capaces de tener cierto grado de razonamiento.

Debido a lo anterior, nace la robótica. Algunos historiadores opinan que esta nació con los antiguos griegos, alrededor del año 270 a. C., cuando Ctesibio[1] construyó órganos y relojes de agua con piezas móviles. Otros historiadores piensan que la robótica comenzó con los muñecos mecánicos en la década de 1770, en donde Pierre Jacquet-Droz[2], creó tres peculiares muñecos, cada uno puede[3] desempeñar una función: uno escribe, otro toca piezas musicales en un órgano, y el tercero puede dibujar. Ya más recientemente, en el año de 1898, Nikola Tesla[4] construyó un bote sumergible controlado por radio, que más adelante planeaba hacerlo autónomo, sin embargo la falta de fondos le impidió seguir investigando.

A pesar de que como se mencionó en párrafos anteriores, el desarrollo de la robótica se sitúa miles de años atrás, no es sino hasta el año de 1921 cuando el checo Karel Capek usa por primera vez la palabra "robot", en una obra teatral[5]. *Robot* en checo significa trabajo forzado. En la obra *RUR*, [1] un fabricante de criaturas mecánicas diseñaba robots para reemplazar a trabajadores humanos, eran eficientes pero sin emoción alguna, al principio se creyó que estos robots eran mejores que las personas, puesto que hacían lo que se les ordenaba sin preguntar, pero finalmente, los robots se rebelaron contra sus amos. Casi exterminaron la raza humana, únicamente dejaron vivo a un solo hombre para que pudiera seguir produciendo más robots.

En consecuencia a lo anterior, existe hasta hoy el miedo a que los robots se apoderen del trabajo de los humanos, y quizá por esto mismo ha dado como resultado el retraso en el desarrollo de esta área. No obstante, algunas otras personas tienen ideas positivas acerca de los robots. Estas

[1] Ingeniero griego de Alejandría que vivió en el siglo III a. C.
[2] Relojero suizo que también inventó el reloj de muñeca
[3] Aún funcionan los muñecos, en la actualidad se exhiben en Museo de Arte e Historia de Neuchâtel, y el primer domingo de cada mes se ponen en movimiento para deleite del público.
[4] Ingeniero croata, nacionalizado estadounidense. Nacido en el año de 1856 y fallecido en Nueva York en enero de 1943
[5] *Rossums Universal Robots (RUR)*, obra que tuvo mucho éxito en Broadway

personas imaginan al robot como un ayudante para la humanidad, una de ellas fue Isaac Asimov, quien postuló en sus historias de ciencia ficción en la década de 1940, tres leyes para la robótica.

1.1.1 Leyes de la robótica

1. Un robot no debe dañar a un ser humano ni, por su inacción, dejar que un ser humano sufra daño.
2. Un robot debe obedecer las órdenes que le son dadas por un ser humano, excepto si éstas entran en conflicto con la primera ley.
3. Un robot debe proteger su propia existencia, a menos que ésta entre en conflicto con las dos primeras leyes.

Posteriormente, en la década de los 50, George C. Devol y Joseph H. Engelberger fundaron *UNIMATION Robotics Company*[6]. En 1960 Devol vendió su patente a la corporación *Condec.* En 1962 General Motors instaló el primer UNIMATION en su planta de fundición por troquel en New Jersey.

A partir de entonces, la robótica ha evolucionado en numerosas aplicaciones, desde su uso en soldadura, pintura, ensamble, carga y descarga de herramentales de máquinas, inspección, agricultura, enfermería, cirugía médica, usos militares y seguridad hasta las exploraciones subacuáticas y del espacio.

1.1 Estado del arte.

A lo largo de la historia, la humanidad ha visto inventos que han transformado radicalmente el mundo: desde la rueda que le permitió transportar cargas pesadas a lo largo de largas distancias así como elaborar un sinfín de máquinas, el telégrafo que le permitió comunicarse de una manera mucho más rápida comparada con utilizar el correo, o el transistor[7] que fue la clave para poder desarrollar computadoras más compactas y más rápidas comparadas con las que usaban tubos de vacío. Desde entonces, el avance de la tecnología ha logrado que en la actualidad contemos con dispositivos electrónicos para un sinfín de aplicaciones, estos dispositivos son cada vez más eficientes y compactos. Del mismo modo se han desarrollado equipos de cómputo que poseen cada vez más capacidad de procesamiento, incluso contamos con dispositivos móviles que poseen características y funcionalidades muy semejantes a las de una computadora de escritorio ordinaria. Estos logros han impulsado diferentes áreas de la tecnología y una de ellas es la robótica.

Desde la primera generación de robots (Tabla 1), se han desarrollado diferentes tipos de autómatas, que, dependiendo de las tareas para las cuales fueron diseñados poseen capacidades y características diferentes. En la actualidad un gran número de centros de estudios, universidades y empresas han invertido en el diseño y construcción de robots.

[6] La palabra *UNIMATION* es la contracción de los términos universal y automation, debido a la creencia de que el robot es una herramienta universal que puede usarse para muchos tipos de tareas.
[7] Inventado en los laboratorios Bell en 1947.

1° generación (1960-1980): Se construyen robots industriales que realizan tareas repetitivas. Se desarrollan las funciones de manipulación fundamentalmente.
2° generación (1980-1985): Se construyen robots dotados de sensores capaces de interactuar con el entorno. En este caso el robot es capaz de modificar su actuación en tiempo real en función de lo que ocurra en su entorno. Se desarrollan considerablemente las funciones de percepción y algo de planificación.
3° generación (1985-actual): Se construyen robots móviles. Esto implica un gran desarrollo de la tarea de la inteligencia artificial aplicada a la robótica así como de la tarea de planificación. Además se mejoran significativamente las tareas de percepción.

Tabla 1 Generaciones de la robótica.

A continuación se describen algunos robots que han sido construidos por diferentes organizaciones, empresas o centros de estudios, estos robots destacan por su innovación dentro del campo de la robótica.

1.2.1 Robot ASIMO.

En 1986 la empresa japonesa Honda comenzó con el desarrollo de un robot andante, las primeras versiones de este Robot (E1, E2 y E3) solo constaban de piernas robóticas que simulaban el caminar humano. En las versiones posteriores (E4, E5 y E6) los ingenieros de Honda trabajaron para lograr equilibrio y estabilidad en el autómata, permitiéndole subir escaleras. Posteriormente se le agregaron cabeza al cuerpo y los brazos para mejorar el equilibrio, dando pie a su nueva serie de modelos (P1 P2 y P3). En el año 2000 se bautizó a este robot como ASIMO (Figura 1.1).

Figura 1.1 *Robot ASIMO* [2].

Esta versión tiene un aspecto más amigable, puede moverse con una velocidad de $9\ [Km/h]$, sube y baja escaleras, trata de alcanzar y agarrar objetos, responde a comandos por voz y reconoce a un determinado grupo de personas a través de un sistema de visión que está basado en dos cámaras instaladas en la cabeza que además le permite navegar en entornos ordinarios. El sistema de

inteligencia de este robot consiste en un PC para el procesamiento de imágenes, un PC para la síntesis y reconocimiento de voz, un procesador para el control y la planificación y una tarjeta de DSP para detectar fuentes de sonido. Esta versión de ASIMO ya está siendo arrendado a empresas para desarrollar trabajo de recepcionista.

1.2.2 Robot Junior.

En el 2007 la Universidad de Standford participó en "Desafío Urbano" de la DARPA[8]con un vehículo robótico con el nombre de Junior que se diseñó con la colaboración de estudiantes, profesores y empleados de la Universidad (responsables del software, y liderato principal del proyecto) y varias organizaciones afiliadas como el laboratorio de investigación electrónica de Volkswagen de América (responsable del desarrollo del vehículo), Google e Intel. Este vehículo se trata de una modificación de un Volkswagen Passat Wagon (Figura 1.2) equipado con un motor de inyección de diésel a cuatro cilindros. El motor de este vehículo es el encargado de proporcionarle la energía eléctrica a cada uno de sus elementos a través de un prototipo de alternador de alta corriente. Para la navegación, este robot cuenta con un sistema que le proporciona integración en tiempo real de múltiples sensores GPS de doble frecuencia, para la medición de distancia cuenta con un sistema de odometría en las llantas, el error de posición y orientación en tiempo real de este sistema es de $100\ [cm]$ y 0.1 grados respectivamente. Para la detección de la estructura de la carretera se utilizan dos sensores láser montados en los laterales y un sensor láser más montado en la parte delantera del vehículo robotizado, apoyados de un sistema de visión conformado por seis cámaras de video las cuales están conectadas a los ordenadores de Junior a través de una interfaz *FireWire*, este sistema proporciona una vista omni-direccional de la zona que rodea al vehículo que también incluye a la superficie de la carretera. Para lograr la correcta detección de obstáculos el robot cuenta con un módulo Velodyne HDL-64E montado en el techo, este módulo incorpora 64 diodos láser, los cuales giran a una frecuencia de hasta $15\ [Hz]$ proporcionando un campo de visión horizontal de 360 grados y un campo de 30 grados vertical. El módulo Velodyne es complementado con cinco radares de largo alcance BOSCH montados alrededor del vehículo. El sistema de cómputo de Junior está montado en el maletero y consta de dos ordenadores con procesadores multi-núcleo de Intel, así mismo el robot cuenta con una interfaz de módulos de control para todos los sensores y actuadores. Todas las computadoras cuentan con sistema operativo Linux. La universidad de Stanford ganó el segundo lugar del desafío urbano con este robot.

Figura 1.2 *Robot Junior* [3]

[8] Carrera de vehículos autónomos que deben llegar desde un punto de los E.U.A a otro sin intervención humana.

1.2.3 Robot Cheetah.

En el año de 2009 el laboratorio de *Biomimetic Robotics* del Instituto Tecnológico de Massachusetts (MIT) anunció su proyecto Cheetah, el cual se trata de un robot cuadrúpedo, que, como su nombre lo indica está basado en la morfología del mamífero Chita (Figura 1.3). Este robot posee una espalda articulada que se flexiona ida y vuelta en cada paso, esto hace que aumente su zancada y por consiguiente su velocidad de carrera de la misma forma que lo hace el animal. Este robot pesa 32 Kilogramos, mide $70\ [cm]$ de alto y cuenta con motores eléctricos ligeros colocados en los hombros del cuadrúpedo que producen un alto par con muy poco calor desperdiciado. En Septiembre del 2014 el equipo del MIT encargado del desarrollo de este robot hizo una demostración en la cual se observa que este autómata es capaz de correr sin ataduras y saltar obstáculos de una altura de hasta $33\ [cm]$ lo cual corresponde al 40 % de su altura con total autonomía. Para logar el salto el robot estima la altura del objeto y la distancia, en base a las lecturas de un sistema visual que utiliza los reflejos de un láser para mapear el terreno. En base de la altura del obstáculo el robot aplica una cierta cantidad de fuerza para aterrizar con seguridad. Actualmente se considera a Cheetah como el robot cuadrúpedo más rápido del mundo alcanzando velocidades de hasta $48\ [km/h]$.

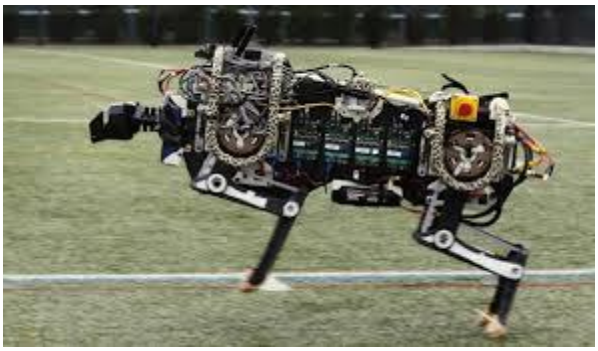


Figura 1.3 Robot Cheetah [4].

De la misma manera que el área de la robótica va teniendo un mayor impulso, se están desarrollando nuevas tecnologías dentro del campo de la inteligencia artificial. Uno de los campos de la inteligencia artificial que ha tenido grandes avances es la visión por computadora. En los siguientes párrafos se mencionan algunos proyectos innovadores dentro del área de la visión por computadora.

1.2.3 Ayuda visual basada en el reconocimiento de objetos.

En el artículo [5] se describen una serie de módulos basados en el procesamiento de imágenes. Este software corre sobre un dispositivo con sistema operativo Android y su finalidad es ayudar a personas ciegas o con debilidad visual. Este proyecto está constituido básicamente por dos módulos, un módulo es el encargado del reconocimiento de objetos a través de técnicas de procesamiento de imágenes como detección de bordes, segmentación por umbral, transformación al espacio de colores HSV y análisis de histogramas. Por otra parte el segundo módulo realiza la detección de movimiento, llevando a cabo la eliminación del fondo de la imagen que contiene el objeto reconocido por el respectivo módulo, posteriormente el movimiento es estimado a través de cambios drásticos entre valores de pixeles de una imagen cargada previamente y los valores actuales. Las notificaciones que recibe el usuario son mediante mensajes verbales.

1.2.4 Un compacto sistema de seguimiento de objetos.

En el artículo [6] se explica el desarrollo de un sistema compacto de seguimiento de objetos que utiliza un microcontrolador y dos servomotores, así como la cámara y el procesador de un dispositivo con sistema operativo Android para capturar y procesar cuadros de video, este sistema se muestra en el diagrama de la figura 1.4. El algoritmo de seguimiento es una novedosa combinación de tres algoritmos conocidos en el área de procesamiento de imágenes, los cuales son SURF (*Speeded-Up Robust Features*), este algoritmo es utilizado para obtener información del objeto a seguir y poder reconocerlo satisfactoriamente, CAMSHIFT (*Continuously Adaptive Mean Shift*) y Lucas-Kanade. Los últimos dos algoritmos buscan estimar tanto la posición como el movimiento del objeto fijando puntos de interés dentro de este, por lo tanto, el seguimiento no depende del objeto mismo sino de la existencia de los puntos mencionados. Estos algoritmos fueron desarrollados utilizando la librería OpenCv en un entorno de Android. Con este proyecto se busca implementar un sistema de seguimiento compacto y económico comparado con los sistemas de seguimiento que existen en el mercado.

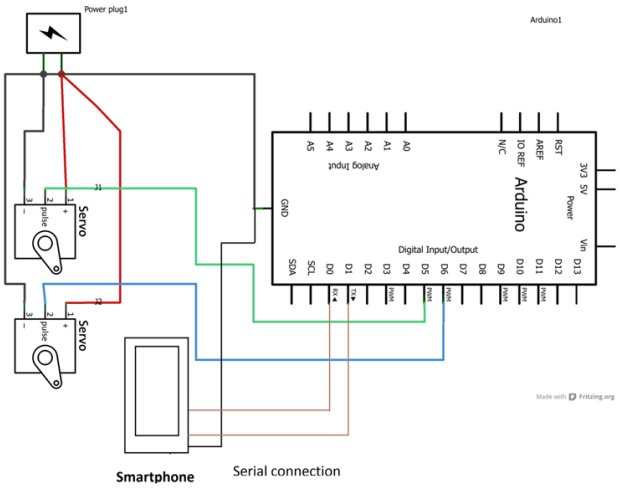


Figura 1.4 Circuito electrónico del sistema de seguimiento de objetos [6].

1.2 Tipos de robots móviles.

Podemos definir a un robot móvil como un manipulador diseñado para trasladarse libremente de un punto a otro mediante movimientos programados para llevar a cabo la realización de varias tareas. Todo esto sin la intervención de un ser humano que lo opere.

El campo de acción de los robots móviles comprende diferentes ramas que van desde la investigación científica, la actividad productiva, la domótica e incluso la agricultura. En la investigación científica un robot móvil puede ser empleado para la exploración de campos que al ser humano se le hace imposible o riesgoso ir, como por ejemplo lugares afectados por radiación nuclear, medios submarinos e incluso terrenos fuera de nuestro planeta.

Por otra parte, en la actividad productiva, un robot móvil es empleado para hacer más segura y eficiente la producción, realizando tareas que serían de alto riesgo, difíciles e incluso imposibles para una persona, por ejemplo transportando materiales que debido a su naturaleza o dimensiones significarían un riesgo físico para un humano.

En la domótica los robots móviles pueden ser empleados como medios de servicios, utilizando al robot como un asistente encargado de varias tareas como puede ser la limpieza. En este ámbito también se encuentran los robots como medio de transporte, el cual es un campo que en la actualidad ha tenido un gran impacto y un avance considerable.

En el campo de la agricultura los robots móviles son empleados para la siembra o recolección de diferentes cultivos, también son utilizados para la fumigación e inspección[9].

Como se puede observar los robots móviles son empleados dentro de una gran gama de actividades, es por eso que se han diseñado un gran número de modelos y tipos de robots móviles.

Para clasificar a los robots móviles los dividiremos en tres grandes grupos caracterizados por el medio que emplean para su locomoción:

1.3.1 Robots con ruedas.

Este tipo de manipuladores son los más populares que existen y esto se debe a que son fáciles de construir y pueden soportar una carga relativamente mayor a la que podría soportar cualquier otro tipo de robot, otra ventaja que poseen es que pueden alcanzar velocidades considerablemente altas. Estos robots generalmente son construidos para maniobrar en terrenos suficientemente duros, dándole una desventaja en terrenos blandos. Dentro de este grupo de robots existen diferentes tipos, dependientes de la configuración y funcionamiento de sus ruedas, estas configuraciones les otorgan diferentes propiedades respecto a su eficiencia, maniobrabilidad y carga útil. A continuación se definen brevemente estas configuraciones:

1.3.1.1 Ackerman.

Son vehículos de cuatro ruedas, tal cual un coche tradicional como se observa en la figura 1.5. Existen robots móviles que han sido el resultado de modificaciones en automóviles convencionales. Las ruedas delanteras son las encargadas de darle dirección a estos vehículos. Para eliminar el deslizamiento en este tipo de robots la rueda interior gira ligeramente en un ángulo mayor que al de la exterior ($\theta_1 > \theta_0$).

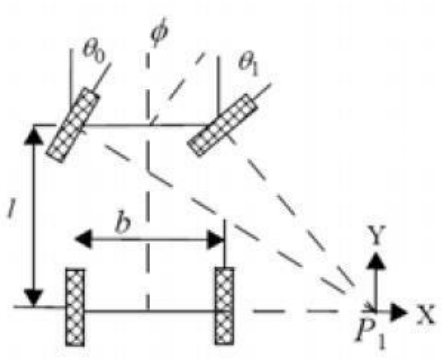


Figura 1.5 Configuración Ackerman [7].

[9] Ejemplos de robots en la agricultura http://www.ieee-ras.org/agricultural-robotics

1.3.1.2 Triciclo clásico.

Esta configuración es más simple que la anterior, debido a que la rueda delantera es la encargada de darle dirección y tracción al robot. El eje trasero está conformado por dos ruedas laterales que se mueven libremente (Figura 1.6). Este robot presenta una mayor maniobrabilidad comparado con la configuración Ackerman, pero puede presentar problemas de estabilidad en terrenos difíciles.

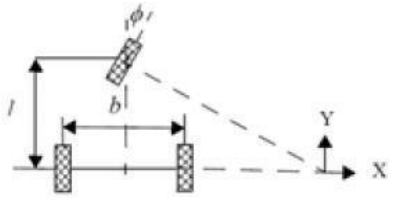


Figura 1.6 Robot Móvil configuración triciclo clásico [7].

1.3.1.3 Diferencial.

Las ruedas laterales encargadas de darle tracción a un robot con este tipo de configuración, son las mismas que le dan dirección, esta se obtiene al variar la velocidad de las ruedas, la diferencia de velocidades de las ruedas resulta en un cambio de dirección de estos vehículos. Suelen tener una o dos ruedas de apoyo, este tipo de robots se ilustran en la figura 1.7.

Figura 1.7 Configuración diferencial [7].

1.3.1.4. Pistas de deslizamiento

La dirección y tracción se obtiene de variar las velocidades de sus ruedas, las cuales tienen acopladas pistas de deslizamiento como se aprecia en la figura 1.8, estos vehículos destacan por su gran potencia y por su buen desempeño en terrenos suaves e irregulares.

Figura 1.8 Robot móvil con pistas de deslizamiento [8].

1.3.1.5 Omnidireccionales.

En los robots con este tipo de configuración (Figura 1.9) se emplean un tipo de ruedas especiales llamadas “ruedas suecas” las cuales permiten conseguir el movimiento hacia cualquier dirección. Es por esto que este tipo de configuración permite una mayor libertad de movimiento respecto a las configuraciones anteriores.

Figura 1.9 Robot móvil Omnidireccional [9].

1.3.2 Robots con patas.

Este tipo de robots presentan una mayor complejidad en cuanto a mecanismos necesarios para su funcionamiento y respecto a su control, por esta razón la construcción de este tipo de robots representa un costo mayor respecto a los robots con ruedas. La principal ventaja que tienen es que son muy versátiles en terrenos muy irregulares y tienen una gran facilidad de evadir obstáculos, estos tipos de robots incluso pueden subir escaleras.

La configuración más común que utilizan es la de seis patas como el ejemplo de la figura 1.10, aunque también existen de ocho patas, cuadrúpedos y bípedos que mantienen el equilibrio de forma dinámica.

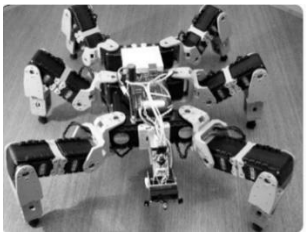


Figura 1.10 Robot móvil con configuración de seis patas [10].

1.3.3 Robots con configuraciones articuladas.

Este tipo de robots son recientes y requieren todavía de un avance significativo para lograr su funcionamiento en campos fuera de los laboratorios (Figura 1.11). Estos robots son principalmente empleados en terrenos difíciles a los que tiene que adaptarse el cuerpo del robot como son caminos estrechos o tuberías con un diámetro pequeño.

Figura 1.11 Robot móvil articulado [7].

1.3 Sonda Robot Móvil.

La sonda Robot Móvil (SRM) es un robot con configuración de triciclo clásico como se observa en la figura 1.12. Fue desarrollado en la Universidad Nacional Autónoma de México (UNAM), específicamente en el DISCA (Electrónica y Automatización) del Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS), que fue pensado de manera general para asistir y llevar a cabo tareas arriesgadas, como lo son el transporte de material peligroso, limpieza industrial, excavaciones, etc. Así mismo podría ser empleado en tareas de tele operación, en donde existe un retraso sensible en las comunicaciones, y es por eso que resulta interesante el uso de vehículos con cierto grado de autonomía. Otro grupo de aplicaciones en donde cabe este tipo de robots, lo componen las labores de vigilancia, inspección o asistencia a personas discapacitadas.

La SRM, con respecto a su estructura física, está constituido por una base de forma hexagonal, de $43\ [cm]\ x\ 50\ [cm]$ con una altura de $27\ [cm]$. Originalmente contaba con un manipulador de 6 grados de libertad, pero en la actualización anterior se decidió omitirlo. La placa hexagonal está construida en aluminio de $1\ [cm]$ de espesor, que funge como base para los elementos mecánicos, eléctricos y electrónicos.

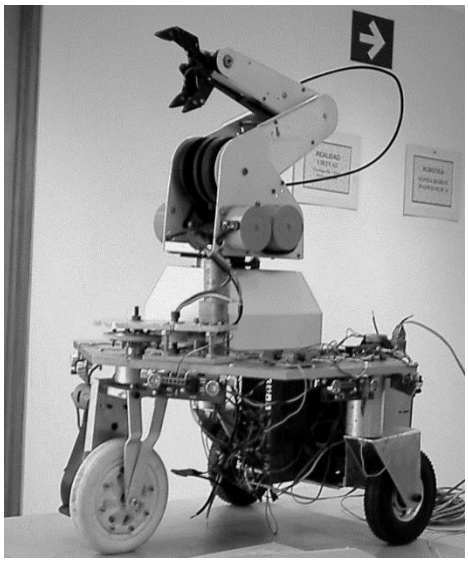


Figura 1.12 Sonda Robot Móvil (SRM) original [11].

Los elementos que constituyen la electrónica de la SRM están alojados en el compartimiento trasero:

- Contaba con tres tarjetas electrónicas (circuitos impresos). Una de ellas contenía el microcontrolador y todos sus periféricos (convertidor A/D, puertos extra, compuertas OR y AND, oscilador a $6\ [MHz]$, capacitores y terminales de entrada/salida). Las otras dos tarjetas contenían la etapa de potencia, que alimenta tanto a los motores (tracción, dirección y los del manipulador), cuya configuración está basada en transistores. Actualmente la SRM solo cuenta con la tarjeta que contiene la etapa de potencia para los motores de tracción y dirección.
- La tracción está compuesta por un motor de CD de $12\ [V]\ /\ 2\ [A]$. Situado en la llanta delantera.
- La dirección, colocada por encima del chasis es movida por un motor de CD $12\ [V]\ /\ 500\ [mA]$.
- La etapa de control se llevaba a cabo por medio de un microcontrolador 8751H de Intel, el cual se conectaba a una interface periférica 82C55 de la misma marca (puertos extra), para encender la etapa de potencia de los motores.
- La SRM contaba con una etapa de adquisición de datos (presión, temperatura, humedad relativa, etc.) que se llevaba a cabo por medio de un convertidor analógico/digital de 8 bits y 16 canales multiplexados, el cual utiliza una técnica de aproximación sucesiva en la conversión.

- Los datos provenientes del convertidor A/D eran almacenados en una EEPROM 2864 (U3 de la figura 5.6) de 8 KB X 8 (8191 localidades de 8 bits), para su posterior envío a una PC, si así se requería.
- La etapa de comunicación era realizada por medio de una interfaz RS-232 (MUX232) serial, que se encontraba por un lado conectada al microcontrolador y por otro podía ser conectada a una PC.
- El robot también contenía un transmisor inalámbrico para video.
- Todo el sistema utiliza una batería de $12\ [V]\ /\ 7\ [Ah]$ que se coloca por debajo del chasis y por detrás de la llanta delantera.

1.4.1 Sensores Ultrasónicos

Para hacer posible la evasión de obstáculos se incluyeron 6 sensores ultrasónicos en la periferia de la SRM, que eran multiplexados con ayuda de un par de circuitos integrados y el microcontrolador. Básicamente se puede describir en 7 pasos el proceso de detección de obstáculos (Figura 1.13) empleado con anterioridad:

1) Un transductor emisor envía periódicamente ultrasonidos de breve duración sobre el obstáculo a detectar.
2) La salida del circuito del sensor ultrasónico, seleccionado por un microcontrolador, y a través de un circuito multiplexor, permanece en nivel alto de voltaje hasta que el cambiador de nivel le indique lo contrario (lo que sucederá si se recibe un eco de retorno, es decir, si existe un obstáculo cerca del transductor receptor seleccionado).
3) Inmediatamente después del tren de ondas, un dispositivo de cronometrado (cuya base de tiempo es regulable) toma la salida.
4) La salida cambia de estado cuando el transductor receptor recibe el eco de retorno.
5) Si no se encuentra un objeto cerca del transductor receptor la salida del circuito seguirá en su estado normal (nivel alto de voltaje).
6) El microcontrolador recibe la salida y la maneja de acuerdo a sus necesidades, además, dicho dispositivo electrónico tiene la capacidad de conocer el estado de otro sensor ultrasónico, utilizando el multiplexor.
7) El ciclo anterior se repite mientras el dispositivo se encuentre encendido.

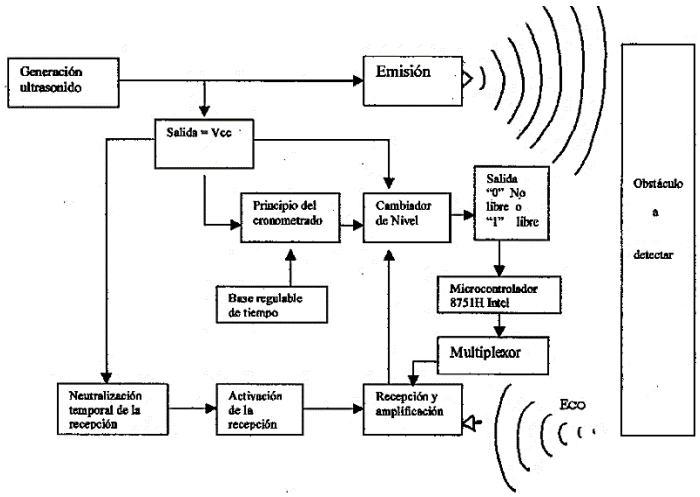


Figura 1.13 Esquematización de la detección de obstáculos en la SRM [11]

También es conveniente explicar en qué consistía el funcionamiento de cada sensor. En la figura 1.14 se aprecia el funcionamiento de los sensores ultrasónicos, dividiendo éste en dos etapas:

I. Seis circuitos transmisores. Cada uno de ellos, individualmente contienen los transductores emisor y receptor. Éstos son los encargados, cada uno por su parte, de emitir la señal ultrasónica al aire, de filtrar y amplificar la señal recibida como eco, si esta se presenta.
II. El circuito maestro, se encarga de generar la señal ultrasónica para todos los circuitos transmisores, de recibir la señal amplificada (de un solo circuito transmisor) y de enviarla al cambiador de nivel (flip flop RS).

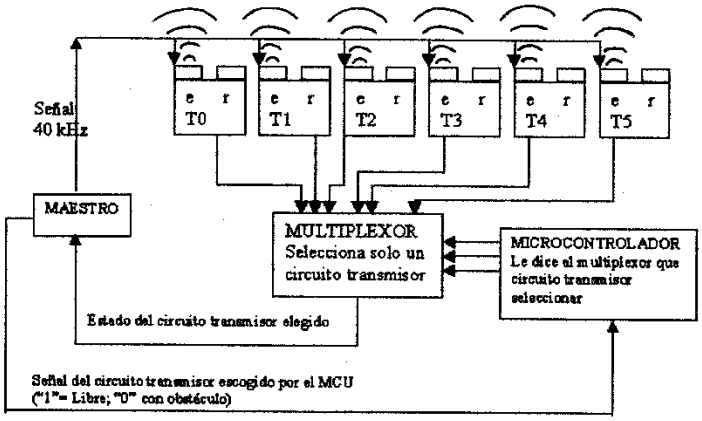


Figura 1.14 Funcionamiento de los sensores ultrasónicos [11].

Donde E: Transductor emisor, T: Transductor transmisor, T1, T2, T3,…: Se refiere al número de circuito transmisor.

Para finalizar este apartado, se describe a continuación la etapa del multiplexor, la cual es muy importante, pues es la encargada de seleccionar uno de los 6 circuitos transmisores y verificar su estado, es decir, si ese circuito tiene o no obstáculo dentro de su rango. Los sensores fueron dispuestos tal como lo indica la figura 1.15, de este modo los 6 circuitos emiten la señal simultáneamente, pero por medio del multiplexor es posible conocer el estado de sólo uno, y de esta manera conocer el estado de todos en forma secuencial.

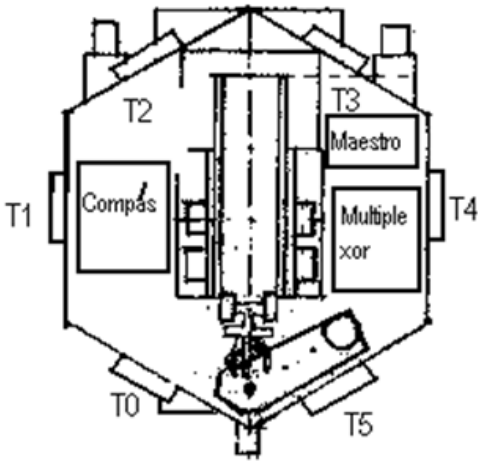


Figura 1.15 Disposición de los sensores y del multiplexor [11].

1.4.2 Compás

Este sensor es una brújula digital, que mide las componentes X e Y del campo magnético terrestre, haciendo permisible conocer la posición angular de la SRM con respecto al norte magnético o geográfico si así se desea.

El dispositivo es fabricado por Precision Navigation, Inc. (PNI). Se trataba de un sensor magneto-inductivo de dos ejes mostrado en la figura 1.16. Este sensor posee las siguientes características:

- Exactitud = 2 °, resolución = 1°.
- Tamaño de 3.75 X 3.4 X $0.8\ [cm]$; peso de $8.6\ [g]$
- Voltaje de alimentación de $VDD\ =\ 5\ [V]$ de CD
- Doble resolución de la salida, 5 y $2.5\ [Hz]$
- Voltaje de entrada y salida de todos los pines de: $-0.3\ <\ VDD\ <\ +0.3\ [V]$
- Corriente máxima de los pines de salida de $5\ [mA]$.
- Tres formatos de salida serial. (BCD, binario y columna).
- Calibración contra campos magnéticos constantes.
- Modo de trabajo continuo o poleo.
- Capacidad para generar el reloj de datos en modo maestro o esclavo.
- Temperatura de operación de $-20\ \ a\ 70\ [°C]$
- Temperatura de almacenamiento de $-30\ a\ 90\ [°C]$

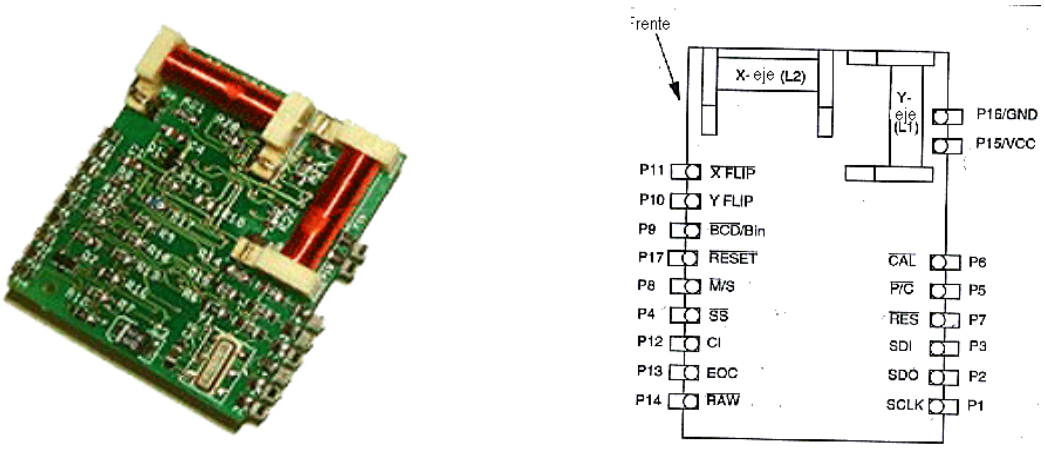


Figura 1.16 Compás empleado en la SRM [11].

Este circuito cuenta con pines programables, permitiendo al usuario utilizarlo de acuerdo a sus necesidades. El compás utiliza un puerto serial síncrono para comunicarse con el dispositivo controlador. Los datos de salida son seleccionables entre los modos binario y BCD (binary code decimal), y la salida de datos serial (SDO, serial data output) permanece en estado de alta impedancia cuando la unidad no se encuentra en el estado de esclavo.

El compás puede proporcionar los datos de salida como ángulo (también llamado Azimut) del compás o como salidas en columna de las dos bobinas sensoras. Estas salidas son relativas a la cantidad de campo magnético que recibe cada una.

- El ángulo o cabeceo es una salida en grados tanto en código binario decimal (BCD) como en formato binario.
- Existen dos salidas columna relacionadas al campo magnético. Se llamará a una de estas salidas la salida columna del eje X y a la otra la salida columna del eje Y del campo magnético. Las lecturas del eje X y del eje Y leídas por los dos sensores del compás son adimensionales.

1.4.3 Odómetro

Los codificadores ópticos (odómetros) se basan en el uso de un disco o rotor ranurado (cratícula), acoplado a un eje que se mueve entre una fuente de luz permanentemente habilitada y un detector. A medida que gira el rotor, el paso de luz hacia el detector es habilitado e inhabilitado por un patrón de áreas obscuras y transparentes impreso en el disco (en nuestro caso es un patrón de material agujerado), produciendo una señal digital que puede ser fácilmente interpretada por el microcontrolador de movimiento del sistema. Típicamente la fuente de luz es un LED infrarrojo o láser y el detector un fototransistor o un fotodiodo. Para mejorar la resolución del conjunto, se utiliza una fuente de luz colimada y se coloca una máscara estacionaria entre la cratícula rotatoria y el detector, todo lo citado se aprecia en la figura 1.17. Esta disposición produce un efecto de persiana, el cual garantiza el paso de luz hacia el detector únicamente cuando las secciones transparentes (o los agujeros) del disco y la máscara estén alineadas.

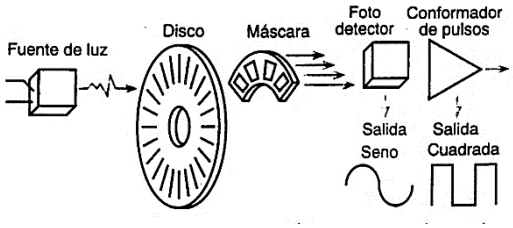


Figura 1.17 Estructura de un codificador óptico incremental rotatorio [11].

La SRM posee dos odómetros, dispuesto uno en cada rueda, así como lo muestra la figura 1.18. En la misma figura se muestran el tipo de interruptor óptico empleado.

Para la construcción del odómetro además de lo anterior, se montó un disco ranurado de $7\ [cm]$ de diámetro, que posee 10 perforaciones, es decir una cada $36°$. Lo que implica una serie de 10 pulsos digitales por vuelta.

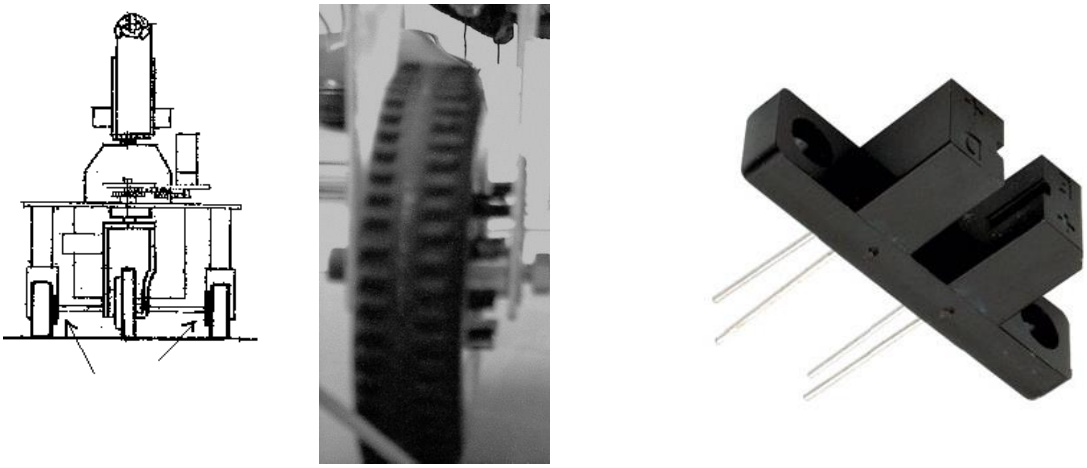


Figura 1.18 Odómetros en las ruedas del SRM (izquierda). Opto interruptor (derecha) [11].

1.4.4 Microcontrolador

El microcontrolador que originalmente fue instalado es el 8751H de *Intel (Figura 1.19),* que se utilizaba para realizar las tareas de control, recepción y transmisión de datos. El circuito integrado 8751H de Intel es un microcontrolador de 8 bits, el cual está basado en la arquitectura de la familia MCS-51 y es construido con tecnología HMOS.

- Unidad Central de proceso (CPU) de 8 bits.
- Oscilador y circuito de reloj internos.
- 32 líneas de entrada / salida.
- 64 Kbyte de espacio de direcciones para memoria de datos externos.
- 64 Kbyte de espacio de direcciones para memoria de programación externa.
- Dos Relojes/Contadores de 16 bits.
- Cinco fuentes de interrupción con dos niveles de prioridad.
- Puerto serial full dúplex.
- Procesador Booleano

El 8751 tiene separados los espacios de dirección para memoria de programa y para memoria de datos. La memoria de programa puede ser de hasta 64 Kbyte de tamaño. Los 4 k más bajos se encuentran internamente en el circuito. La memoria de datos puede consistir de 64 Kbyte de memoria externa, además de incluir 128 bytes de RAM internos, más los SFR (Registros de funciones especiales).

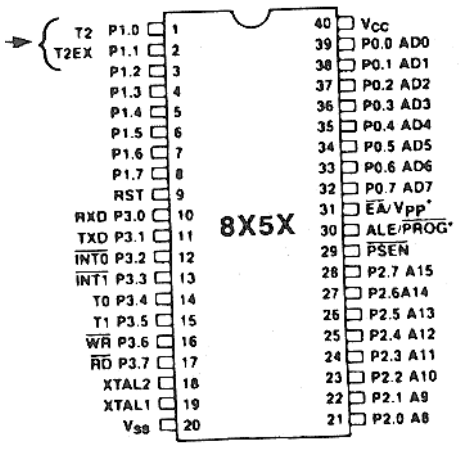


Figura 1.19 Pinout del microcontrolador 8751 [11].

1.4.5 Alcances del proyecto.

Este trabajo de tesis tiene la finalidad de crear tecnología propia en el campo de la robótica, escasamente desarrollada en México, específicamente retomando el proyecto de la SRM y realizando el análisis, diseño y construcción de un sistema de navegación para este robot móvil,

con capacidad de evadir obstáculos que se presenten durante su recorrido en tiempo real, apoyándose de un sistema de visión artificial para reconocer y llegar a su objetivo final con éxito, para ello utilizando como base la plataforma Android y algoritmos de navegación que permitirán dicho objetivo.

En el capítulo 2 se describirá la teoría de los componentes electrónicos implementados en la actualización de la SRM así como las modificaciones que se realizaron sobre la misma. En los capítulos 3, 4 y 5 se presentará el marco teórico sobre el sistema operativo Android, visión por computadora y el algoritmo de navegación implementado para lograr el objetivo de este proyecto respectivamente, en el capítulo 6 se describirá el desarrollo del sistema y finalmente en el capítulo 7 se mencionan las pruebas y los resultados obtenidos de este trabajo.

CAPÍTULO 2

SENSORES.

2.1 Conceptos Importantes.

A lo largo de éste capítulo se presenta información idónea para comprender los componentes que ostenta la SRM. En general, la plataforma está conformada por una suma de circuitos electrónicos que unidos dan por resultado entidades aún más complejas, las cuales tienen un uso específico. Un ejemplo tangible de esto son los sensores.

Pero, ¿Qué es un sensor?

Un sensor es un dispositivo que capta magnitudes físicas u otras alteraciones de su entorno.

Por otro lado, un transductor es un dispositivo que tiene la misión de recibir energía de una naturaleza: eléctrica, mecánica, acústica, etc., y suministrar otra energía de diferente naturaleza, pero de características dependientes de la que recibió.

Usualmente sensor y transductor se emplean como sinónimos, sin embargo, un sensor se refiere a la ampliación de los sentidos para adquirir conocimiento de magnitudes físicas, que debido a su naturaleza o tamaño, los sentidos no son capaces de percibirles.

2.2 Sensores de Distancia.

Existen diversas formas de medir la distancia a la que se encuentran dos objetos entre sí, de modo que, también existe un amplio catálogo de sensores que son capaces de realizar ésta tarea, y la llevan a cabo de distintas maneras.

En este trabajo se toman en cuenta los sensores de distancia por ultrasonido, cuyo principio de funcionamiento resulta sencillo de entender. Básicamente se trata de una onda con una frecuencia superior a los $20\ [KHz]$ que es transmitida por un dispositivo y que al rebotar por la causa de un objeto que se interpone en su trayectoria, esa onda reflejada, es captada por otro dispositivo receptor, como se ilustra en la figura 2.1.

La manera de calcular la distancia reside en tomar en cuenta el tiempo en el que tarda en regresar la onda reflejada al dispositivo receptor (eco). Contando con éste último dato (t), y conociendo la velocidad de propagación del sonido en el aire ($V = 330\ \left[\frac{m}{s}\right]\ a\ 20\ [°C]$), se puede calcular la distancia a la que se encuentra el otro objeto con la siguiente expresión: $d = \frac{1}{2}Vt$.

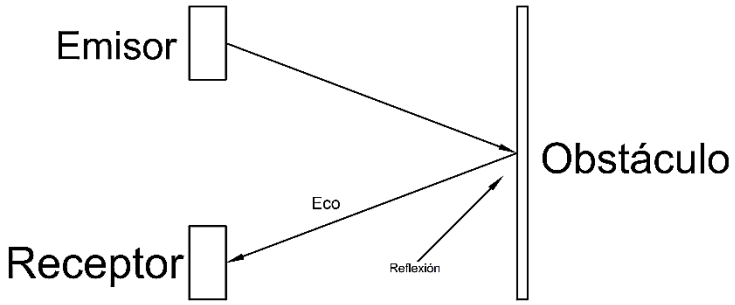


Figura 2.1 Principio de funcionamiento de los sensores ultrasónicos.

2.3 Encoders.

Resulta de vital importancia conocer la distancia del desplazamiento que ha realizado un robot, de este modo se puede verificar que el punto al que llegue sea el requerido. Una manera de lograr esto, es por medio de un odómetro.

Existen diferentes tipos de odómetros usados en robótica, pero los más comunes, funcionales y económicos, resultan ser los codificadores ópticos acoplados a las armaduras de los motores o a sus ejes.

Un encoder óptico es un tipo de sensor que permite detectar el movimiento de rotación de un eje. Consta de un disco o rotor ranurado (cratícula), que acoplado al eje, se mueve entre una fuente de luz habilitada permanentemente y un detector. Conforme gira el rotor, el paso de la luz hacia el detector, es habilitado o inhabilitado por el patrón de ranuras del disco, originando una señal digital que puede ser interpretada con facilidad por un microcontrolador. La fuente de luz puede ser un LED infrarrojo, y el detector puede ser un fototransistor o un fotodiodo. Algunas veces para mejorar se agrega una máscara estacionaria entre la cratícula y el detector, produciendo un efecto de persiana que de algún modo se encarga de garantizar el paso de la luz hacia el detector sólo cuando las ranuras y la máscara estén alineadas, este tipo de sensor es ilustrado en la figura 2.2.

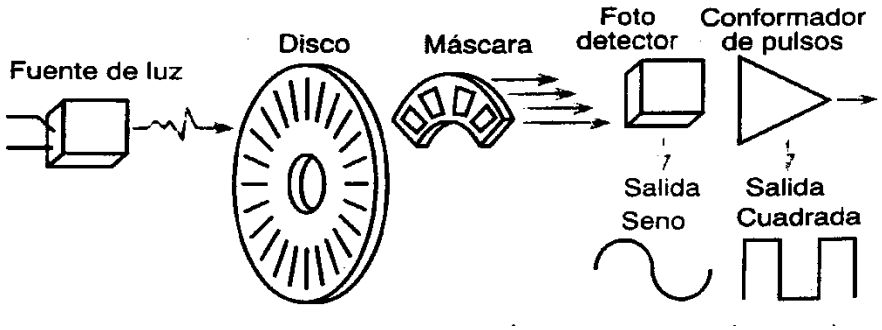


Figura 2.2 Esquema de un codificador óptico incremental rotatorio [11].

2.4 Brújulas Electrónicas.

En la antigüedad era de suma importancia contar con un sistema de navegación, es decir contar con instrumentos que contribuyeran para la correcta orientación de la humanidad. Hoy no es la excepción, sin embargo con el paso de los años, y el avance de la tecnología, se cuenta con dispositivos sofisticados y pequeños para lograr este cometido. El instrumento adecuado para esto es la brújula o también conocido como compás.

La brújula es un instrumento que ya existe desde el siglo IX, consistía en una aguja imantada que flotaba en un recipiente con agua, haciendo posible que la aguja apuntara hacia el norte magnético (sur geográfico). Actualmente existen brújulas electrónicas, que si bien no han cambiado mucho su principio de funcionamiento, difieren entre sí por la manera en la que funcionan y están construidas.

Hablando de brújulas electrónicas, podemos encontrar principalmente 4 diferentes tecnologías:

- Fluxgate
- Efecto Hall
- Magnetorresistividad
- Magnetoinducción

Para este trabajo resulta importante hablar de los sensores magnetorresistivos. Los sensores magnetorresistivos (MR), se refieren a la variación de la resistencia eléctrica de un material en presencia de un campo magnético, y se fabrican depositando una película delgada de una aleación Ni-Fe (Permalloy) sobre una oblea de Si, formando así una lámina resistiva. Durante el proceso de deposición, el material es sometido a un campo magnético externo para alinear en una cierta dirección su vector de magnetización M.

Si se aplica un campo magnético H paralelo a la lámina de permalloy y perpendicular a la corriente, el vector M gira acercándose a la dirección de la corriente (Figura 2.3). El ángulo relativo α entre M e i viene dado por la expresión:

$$sen^2\alpha = \frac{H_y^2}{H_0^2} \quad (1)$$

La resistencia de la lámina cambia en función del ángulo α según:

$$R = R_0 + \Delta R_0 \cos^2 \alpha \quad (2)$$

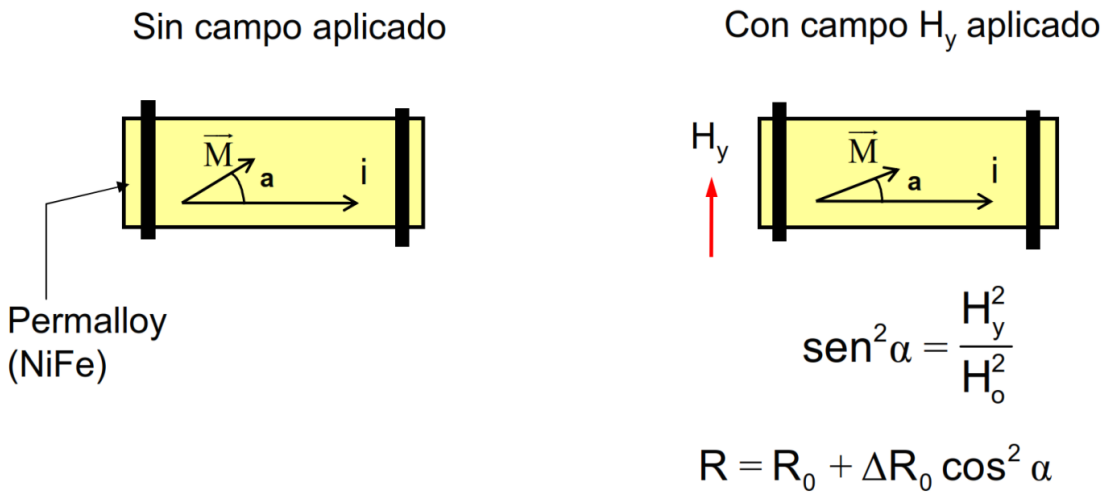


Figura 2.3 Principio de funcionamiento del sensor magnetorresistivo [32].

2.5 Fotorresistencias.

Estos dispositivos varían la resistencia eléctrica de un semiconductor al incidir en éste radiación óptica (radiación electromagnética con longitud de onda entre $1\ [mm]$ y $10\ [nm]$).

Esto puede entenderse mejor si recordamos que la conductividad eléctrica en un material depende del número de portadores en la banda de conducción. Para el caso particular de los semiconductores, a una temperatura baja la mayor parte de sus electrones están en la banda de valencia, de modo que se comporta como aislante. Caso contrario, cuando aumenta la temperatura y consecuentemente los electrones se agitan, ocasionando que los electrones pasen de la banda de valencia hacia la de conducción, y así, logrando que aumente la conductividad.

La energía necesaria para el salto entre bandas, puede venir de fuentes externas, es decir no sólo del calor; una de ellas puede ser la radiación óptica, cuya energía (E) y frecuencia (f), están relacionadas mediante la expresión:

$$E = hf \qquad (3)$$

donde $h = 6.62\ x\ 10^{-34}\ [Ws^2]$ es la constante de Planck. Esto significa que si la radiación tiene la suficiente energía para permitir el salto de los electrones de una banda a otra, pero sin exceder el umbral necesario para que se desprendan del material, obtendríamos el efecto fotoeléctrico interno, dicho en otras palabras, un fotoconductor que a mayor iluminación, presentará una conductividad mayor.

2.6 Sonda Robot Móvil (SRM).

La SRM se retomó en el año 2015, haciendo cambios que conllevaron a mejorar el desempeño de la plataforma. Se reemplazó la electrónica del siglo pasado por módulos y circuitos integrados más específicos. Por ejemplo, se reemplazaron los módulos para medir distancia por unos módulos más pequeños, funcionales y económicos mostrados en la figura 2.5.

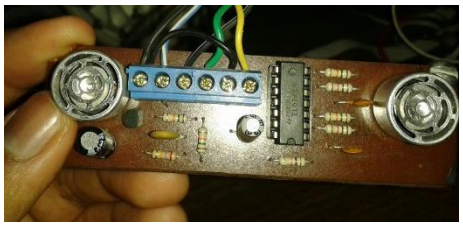


Figura 2.4 Sensor ultrasónico anterior.

Figura 2.5 HC- SR04 Sensor ultrasónico actual.

También se sustituyó el circuito maestro de los sensores de distancia, por otro en donde se genera la señal de Trigger que ocupan los mismos, en la misma placa se integró la parte de multiplexaje para ahorrar espacio y mantener la funcionalidad (Figura 2.6). El circuito empleado fue un multiplexor analógico[10] de 8 canales (74HC4051) que es un dispositivo semiconductor complementario de óxido metálico (CMOS por sus siglas en inglés), empleado para seleccionar cada una de las salidas de los 6 sensores dispuestos en la periferia de la plataforma móvil, y así ahorrar pines de entrada en el microcontrolador.

Figura 2.6 Circuito maestro original (izquierda), y el nuevo circuito.

En la Figura 2.7 se distinguen 4 áreas importantes del circuito diseñado: Oscilador, Inversión de la señal, Multiplexaje y Entradas y salidas.

Para la parte del oscilador se empleó un temporizador LM555, el cual fue configurado para obtener a su salida una señal de reloj con periodo de $10\ [ms]$, con un tiempo en bajo de $1\ [ms]$, consecuentemente esa señal pasa a la parte de inversión de la señal, en donde por medio de un transistor NPN de señal pequeña BC547, configurado en emisor común, se obtiene la inversión de la señal, obteniendo así en el colector la señal con el mismo periodo de $10\ [ms]$, pero con un tiempo en alto de $1\ [ms]$, señal que es suficiente para que el sensor HC-SR04 funcione

[10] Circuito capaz de recibir varias entradas, (en este caso 8, pero sólo se ocupan 6) y con sólo una salida de datos.

adecuadamente. Esta señal se observa en la parte de entradas y salidas, y es denominada en la figura 2.7 como Trigger.

Para la parte del multiplexor, se aprecia en los conectores de selección el bit más significativo (MSB) en la parte inferior y el bit menos significativo (LSB) en la parte superior. En las entradas del multiplexor se concentran las 6 señales de los sensores y se enumeran desde 0 hasta el 5, estos mismos pines están conectados en la parte de entradas y salidas. Finalmente el pin etiquetado como *Out_Mux* está destinado para ser conectado al microcontrolador y este procese a su vez los datos obtenidos de los sensores ultrasónicos, obteniendo así las distancias.

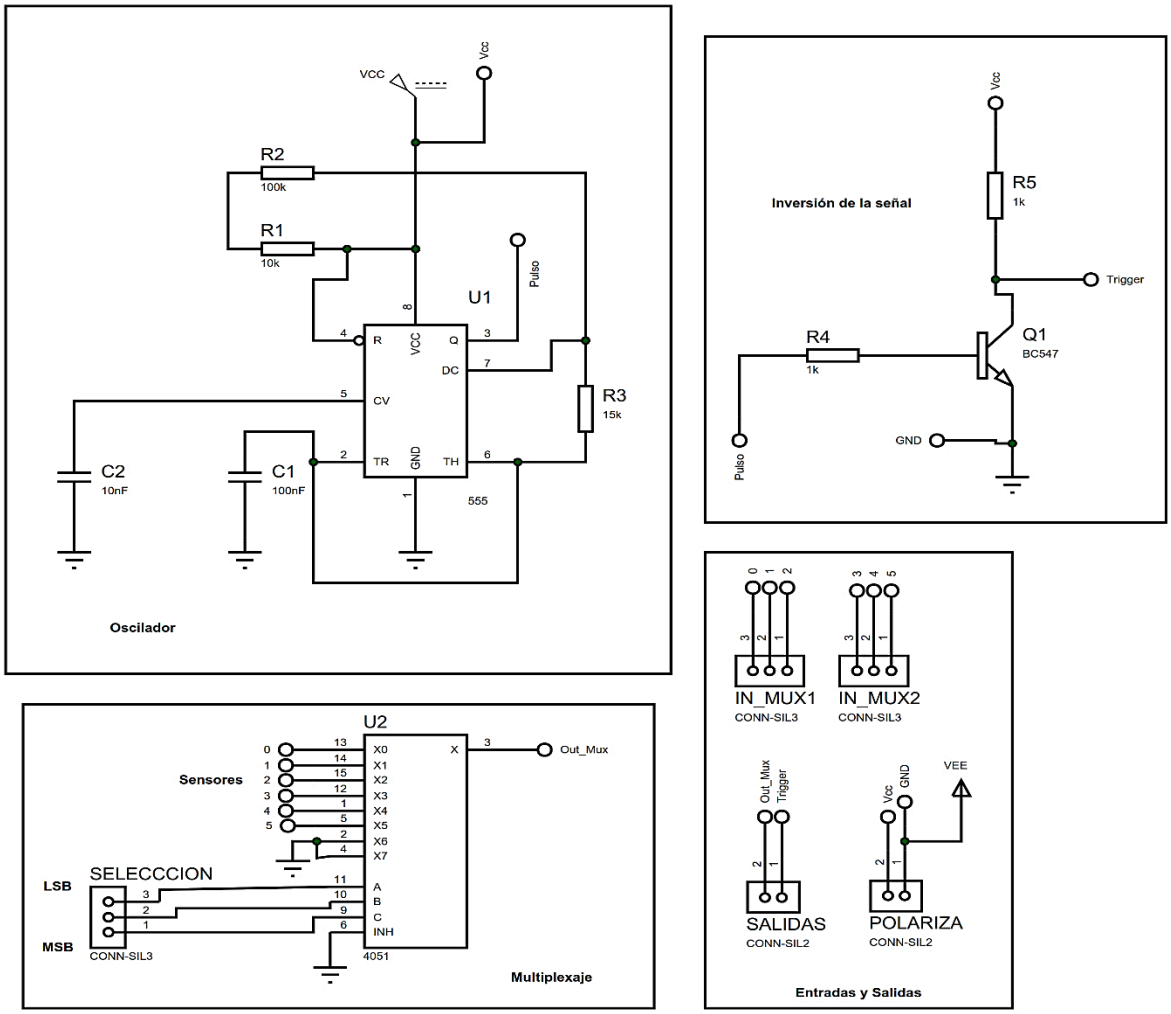


Figura 2.7 Esquemático del nuevo circuito maestro para los sensores ultrasónicos.

Por otro lado, el sistema de navegación estaba conformado por un compás magneto-inductivo fabricado por *PNI Sensor Corporation*, éste se sustituyó por otro compás magneto-resistivo fabricado por *Honeywell, ambos* mostrados en la figura 2.8. El nuevo sensor presenta la ventaja de ser más preciso, la velocidad de actualización de las lecturas se puede seleccionar, funciona con voltajes de $2.7\ a\ 5.2\ [V]$, y si está operando a $3\ [V]$ únicamente consume $1\ [mA]$. Otra de las

ventajas es que opera por medio del protocolo I2C, por tanto sólo se emplean 2 pines en el microcontrolador para comunicarse.

Figura 2.8 Compás magneto-inductivo (izquierda), y compás magneto-resistivo que lo reemplaza [12].

Cabe mencionar que se agregó un circuito para limitar el giro de la rueda delantera para así evitar posibles daños en el cableado de la SRM. El circuito es sencillo, consta de 2 partes: Emisor y receptor. El emisor consiste en 2 láser head, y el receptor está compuesto por un par de fotorresistencias (Figura 2.9) que están colocadas de manera tal que cuando la rueda está alineada hacia el frente, el haz de ambos láser head incide en ambas fotorresistencias, el microcontrolador se encarga de la lectura de voltajes de ambas fotorresistencias.

Figura 2.9 Fotorresistencia del lado izquierdo y láser head.

Además, se sustituyó la batería que tenía por una nueva de $12\ [V]$ @ $12\ Ah$ como la mostrada en la figura 2.10.

Figura 2.10 Batería empleada para la actualización y correcto funcionamiento de la SRM en 2015 [33].

Uno de los agregados más significativos de este proyecto es el uso de un dispositivo móvil con la plataforma Android, para que por medio de la cámara de éste, sea posible el reconocimiento de algún objeto, que en este caso, se propone el reconocimiento de una pelota, para su posterior seguimiento.

El dispositivo empleado es un terminal desarrollado por Motorola. Se trata del XT1032 (Figura 2.11). Este móvil cuenta con un procesador Qualcomm MSM8x26 quad-core A7 1.2GHz (Adreno 305 450 $[MHz]$ GPU), una pantalla de 4.5" 1280 X 720 HD, 329 ppi, cuenta con dos cámaras; la trasera tiene una resolución de 5 Megapíxeles (4:3) / 3.8 Megapíxeles (16:9), con capacidad para captar video a 30 cuadros por segundo, no obstante la cámara frontal tiene una resolución de 1.3 Megapíxeles. También cuenta con conectividad Wi-Fi y Bluetooth. La versión de Android que ostenta es la 4.4.4 (Kit-Kat).

Figura 2.11 Motorola XT1032 [34].

El microcontrolador que utilizaba la SRM, hoy en día resulta prácticamente obsoleto, es así que se optó por algo más sencillo, económico, y funcional. Finalmente, el microcontrolador elegido fue el ATmega328 ilustrado en la figura 2.12, este dispositivo es desarrollado por Atmel y pertenece a la serie megaAVR. Se trata de un microcontrolador de 8 bits con un conjunto reducido de instrucciones (RISC por sus siglas en inglés), y con 32 KB de memoria para programación. Posee una memoria EEPROM de 1 KB, y una memoria estática de acceso aleatorio (SRAM) de 2 KB, 32 registros de proceso general, tres temporizadores flexibles/contadores con modo de comparación, interrupciones internas y externas, programador de modo USART, y convertidor analógico – digital de aproximaciones sucesivas de 10 bits. Este microcontrolador es utilizado con la plataforma Arduino.

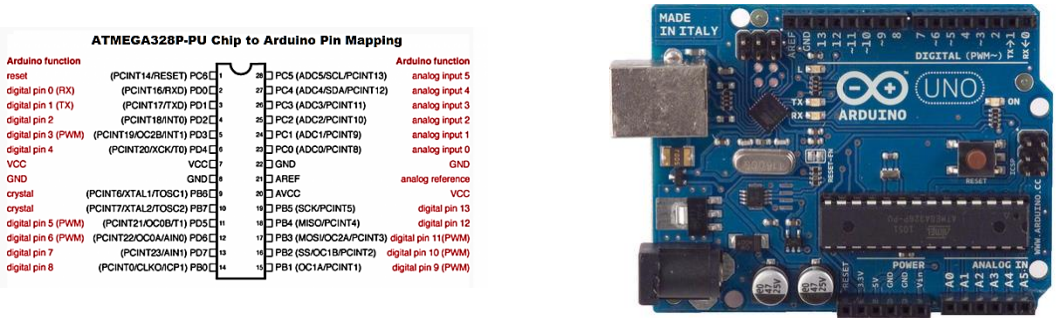


Figura 2.12 Atmega328 (izquierda) y placa Arduino UNO (derecha) [13].

Para hacer posible la comunicación entre el teléfono con Android y el microcontrolador, se agregó un módulo Bluetooth económico, sencillo de usar y de adquirir. Se trata del HC – 05 (Figura 2.12) y utiliza el protocolo UART RS 232 serial. Cuenta con 4 pines (VCC, GND, TX, RX). Es compatible con Bluetooth V2.0, su voltaje de alimentación puede ser entre $3.3[V]\ DC - 6[V]\ DC$, con un Baud

rate ajustable: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200. El consumo en operación es inferior a los $40\ [mA]$ y en modo sleep es menor a $1\ [mA]$.

Figura 2.13 Módulo Bluetooth HC- 05 [14].

Gracias al avance de las técnicas de fabricación de circuitos integrados, hoy en día es posible construir circuitos cada vez más complejos, empleando cada vez menos espacio, y a menor costo.

Por esto y por los cambios anteriormente mencionados, fue posible la rehabilitación de la SRM, proporcionando así, mejoras sustanciales que dotan a este proyecto de lo más actual y funcional en el mercado.

CAPÍTULO 3

SISTEMA OPERATIVO ANDROID.

3.1 ¿Qué es Android?

¿Cuántas veces se ha pensado que el sistema operativo (S.O.) que constituye a Android es similar al que se usa en una computadora? Sin embargo son completamente diferentes, ya que si bien el sistema operativo de ambos se basa en un conjunto de órdenes y programas que controlan los procesos básicos de operación, permitiendo así el funcionamiento de otros programas; en Android éste está diseñado para ser funcional específicamente en teléfonos móviles bajo el núcleo Linux.

Desde el año 2003, este sistema operativo comenzó con la compañía Android Inc., formada por el ingeniero Andy Rubin en Palo Alto, California. Para Julio de 2005 Google adquirió Android Inc., y así Rubin pasó a formar parte del equipo de Google, supervisando el desarrollo de Android y colaborando para la creación de la máquina virtual Java optimizada para móviles, de este modo en Noviembre de 2007, simultáneamente con la fundación de la Open Handset Alliance, finalmente se hiciera oficial el anuncio de Android.

No fue sino hasta Octubre de 2008 cuando HTC hace posible la aparición del primer teléfono con Android, con el HTC Dream.

3.2 Desarrollo a través del tiempo

Como en la mayoría de las veces, el desarrollo de Android comenzó de menos a más. La plataforma comenzó con la idea de tener un artefacto que cupiera en la palma de la mano y que fuera capaz de funcionar como un teléfono y tener toda la disponibilidad para poder navegar en internet, posteriormente Google, ambiciosa y acertadamente ideó tener un sistema operativo especial para dispositivos móviles.

Actualmente se han desarrollado 12 versiones de este S.O. A continuación se muestra en la tabla 2 una comparativa de dichas versiones:

Versión		Fecha de lanzamiento	Características
Android Beta		5/11/2007	• El Kit de desarrollo de Software (SDK) fue lanzado hasta el 12/11/2007.
Android 1.0	**Apple Pie**	23/09/2008	• Incorpora Android Market. • Navegador Web que es capaz de visualizar HTML y XHTML • Soporte para cámara. • Soporte POP3, IMAP4 y SMTP. • Sincronización de los servicios de Google (Maps,Gmail,Calendar, etc.). • Mensajería instantánea (SMS y MMS). • Notificaciones configurables. • Marcación por voz. • Fondo de escritorio configurable. • Soporte para Wi-Fi y Bluetooth.
Android 1.1	**Banana Bread**	9/02/2009	• Reseñas cuando se buscan negocios en los mapas. • Pantalla más larga cuando está en uso el manos libres. • Posibilidad para guardar archivos adjuntos en los mensajes.

Android 1.5	**Cupcake**	30/04/2009	• Soporte para teclados virtuales de terceros con predicción de texto y diccionario de usuarios para palabras personalizadas. • Soporte para widgets. • Grabación y reproducción en formatos 3gp y mpeg-4. • Auto-sincronización y soporte para bluetooth estéreo. • Posibilidad de copiar y pegar en el navegador web. • Posibilidad de subir videos a Youtube. • Posibilidad de subir fotos a Picasa.
Android 1.6	**Donut**	15/09/2009	• Permite capacidades de búsqueda avanzada en todo el dispositivo. • Incorpora gestures y multi-touch. • Permite la síntesis de texto a voz. • Soporte para resolución de pantallas WVGA. • Aparece el atributo XML, onClick. • Mejoras en la aplicación de la cámara.
Android 2.0/2.1	**Eclair**	25/10/2009	• Mejora gestión de contactos. • Ofrece más ajustes en la cámara. • Se optimiza la velocidad de hardware. • Se aumenta el tamaño de ventana y resoluciones soportadas. • Agrega Google Maps. • Soporte para HTML5. • Mejoras en el calendario. • Soporte para Microsoft Exchange. • La clase Motion Events soporta eventos en pantallas multitáctil.
Android 2.2.x	**Froyo**	20/05/2010	• Ejecución del código de la CPU de 2 a 5 veces más rápida que la versión anterior de acuerdo a varios benchmarks. • Soporte para Adobe Flash 10.1 • Incorporación del motor Java Script V8 utilizado en Chrome. • Soporte para instalación de aplicaciones en un medio de almacenamiento externo. • Opción para deshabilitar acceso de datos sobre red móvil. • Es posible dar acceso a internet a otros dispositivos (tethering), tanto por USB como por Wi-Fi. • Soporte para redes Wi-Fi 802.11n. • Se pueden realizar fotos o videos en cualquier orientación.
Android 2.3.x	**GingerBread**	6/12/2010	• Soporta mayores tamaños y resoluciones de pantalla. • Se mejora el diseño de interfaz de usuario con incrementos en velocidad y simpleza. • Incluye soporte nativo para cámara frontal. • Introducción de un nuevo recolector de basura en la máquina virtual Dalvik que minimiza las pausas en las aplicaciones. • Soporte para tecnología NFC. • Soporte nativo para giroscopios y barómetros. • Incorporación de un gestor de descargas para archivos “pesados”. • Soporte para la reproducción de video WebM/VP8 y codificación de audio AAC.
Android 3.x	**Honeycomb**	22/02/2011	• Resolución por defecto WXGA. • Se mejora la reproducción de animaciones 2D/3D gracias al renderizador OpenGL acelerado por hardware. • Soporte optimizado para tabletas. • Primera versión de la plataforma que soporta procesadores multinúcleo. • Optimización de la máquina virtual Dalvik para permitir multiprocesado. • Soporte para la transferencia de archivos multimedia a través de USB con los protocolos MTP y PTP. • Nuevas APIS de Bluetooth A2DP y HSP con streaming de audio. • Se incorpora la conexión de teclados por USB o Bluetooth.
Android 4.0.x	**Ice Cream**	19/10/2011	• Compatibilidad con cualquier tipo de dispositivo, gracias a la unificación

Sandwich			de las dos versiones anteriores (2.x para teléfonos y 3.x para tabletas). • Desbloqueo facial. • Mejoras en el reconocimiento de voz. • Gestor de tráfico de datos de red que permite ver el consumo de forma gráfica, haciendo posible la delimitación del consumo. • Se mejora el API para comunicaciones por NFC y la integración con redes sociales.
Android 4.1	**Jelly Bean**	13/07/2012	• Principal objetivo mejorar un punto débil de Android: la fluidez de la interfaz de usuario. • Los widgets de escritorio pueden ajustar su tamaño. • La función Google Now permite utilizar información de posición, agenda y hora en las búsquedas. • El dictado por voz puede realizarse sin conexión a Internet (de momento en inglés). • Es posible crear varias cuentas de usuario en el mismo dispositivo (sólo en tabletas). • Los widgets de escritorio pueden aparecer en la pantalla de bloqueo. • Se incorpora un nuevo teclado predictivo deslizante al estilo Swipe. • Posibilidad de conectar dispositivos y TVHD mediante Wi-Fi (Miracast).
Android 4.4	**KitKat**	31/10/2013	• Disponibilidad en una gama más amplia de dispositivos, incluyendo aquellos de memoria RAM de 512 MB. • Incorporación de pantalla completa. • Corrige los fallos con la sincronización de cuentas de correos Exchange. • Web Views basadas en el motor de Chromium. • Posibilidad de impresión utilizando Wi-Fi. • Innovación en las transiciones y efectos visuales. • Se añade un content provider para la gestión de SMS. • Incorporación de una nueva máquina virtual ART (En etapa experimental)

Tabla 2 Versiones de Android

Nota Fuente: Tomas, Jesús. (2013). Cap. 1. Visión general y entorno de desarrollo. *El gran libro de Android* (pp. 38-43). México: Alfaomega Grupo Editor.

3.3 Android en el futuro.

El sistema operativo Android que nace en 2007, hoy es uno de los más populares a nivel mundial. Después de implementarse en teléfonos móviles migró hacia los nuevos dispositivos en el 2011, las tabletas. Siendo su siguiente objetivo las Smart TV (televisiones inteligentes). Ahora Android se ha unido a la era wearable, es decir a los dispositivos que son portables en el cuerpo, como: el Moto 360, Samsung Gear S, LG G watch, etc.

Si bien el futuro es totalmente incierto, se puede construir una idea de lo que probablemente pueda pasar si se realiza una retrospectiva en el impacto que ha tenido el desarrollo de la plataforma Android en el mercado.

Por ejemplo, Nielsen confirmó en el 2011 que en E.U. un 40% de los consumidores móviles, mayores a 18 años, ya contaban con un teléfono inteligente (smartphone); de los cuales el 40% contaba con Android y el 28% con iOS. En ese mismo año, la tercera parte de las personas que declararon se comprarían un smartphone, mencionaron que sería uno con Android como sistema operativo.

En el segundo trimestre del 2014, Android fue el sistema operativo más usado, con más de la mitad de los smartphones en E.U., es decir el 52% de los smartphones ejecutaban este sistema operativo. No obstante, Apple sigue siendo el mayor fabricante de smartphones en aquel país, acaparando así el 43 % de los usuarios con su sistema operativo, iOS.

Esta información se puede ver reflejada en los siguientes gráficos:

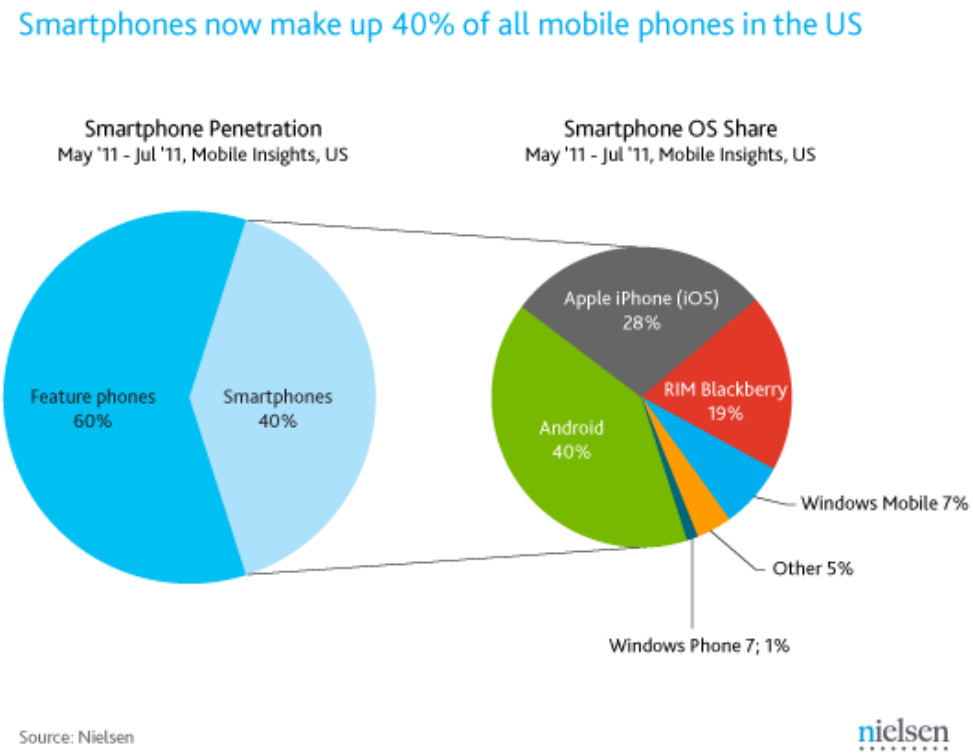


Figura 3.1 Consumidores de teléfonos inteligentes en el 2011 y distribución de los sistemas operativos para el mismo año en los E.U

Nota Fuente: The Nielsen Company (2011) [40 Percent of U.S. Mobile users own smartphones; 40 percent are android], Smartphones now make up 40% of all mobile phones in the US, Fecha de consulta: 20 de Julio de 2015. URL: http://www.nielsen.com/us/en/insights/news/2011/40-percent-of-u-s-mobile-users-own-smartphones-40-percent-are-android.html

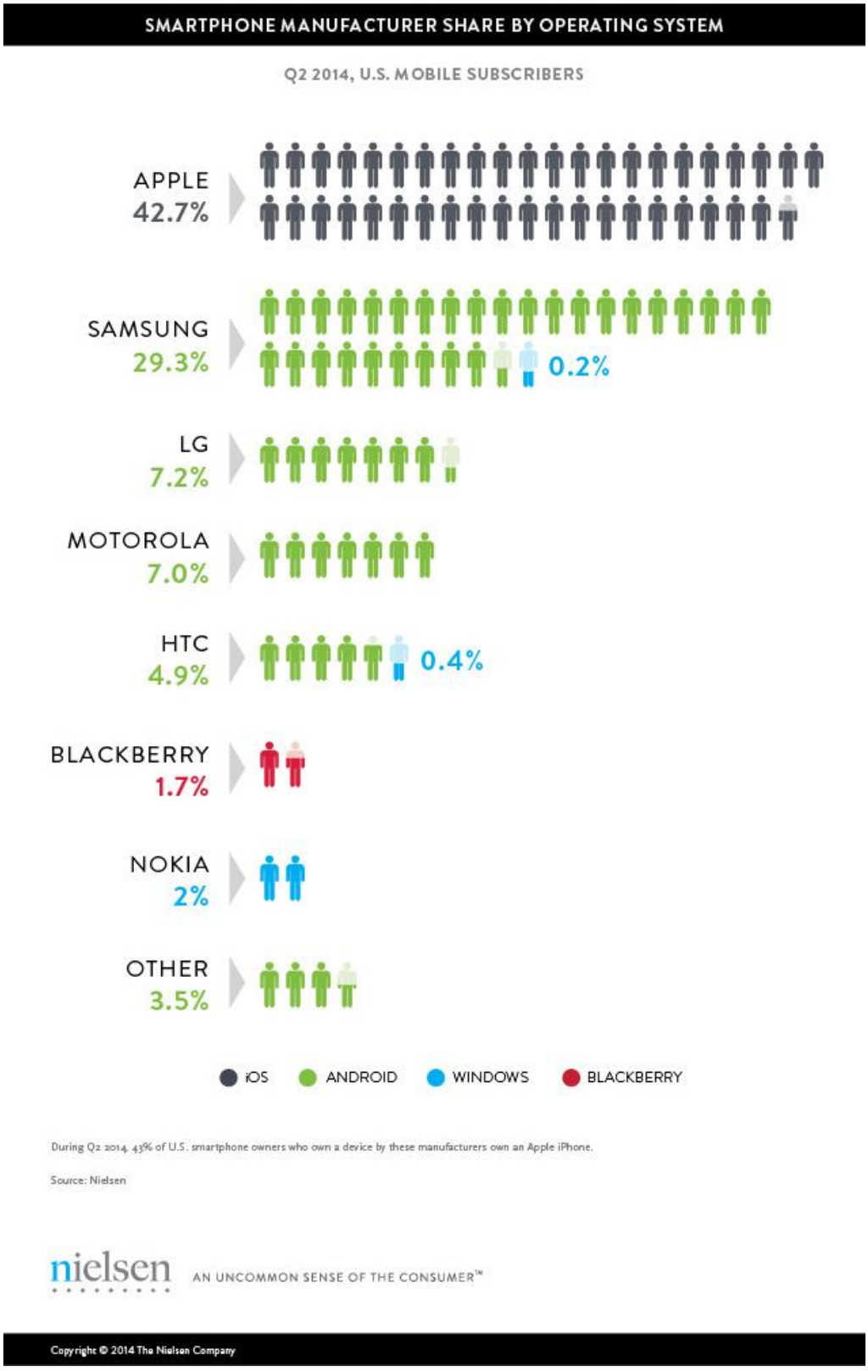


Figura 3.2 Fabricantes de smartphones con diferentes plataformas.

Nota Fuente: The Nielsen Company (2011) [Mobile millennials: over 85% of generation y owns smartphones], Smartphone manufacturer share by operating system, Fecha de consulta: 20 de Julio de 2015. URL: http://www.nielsen.com/us/en/insights/news/2014/mobile-millennials-over-85-percent-of-generation-y-owns-smartphones.html

3.4 ¿Por qué Android?

Android es una plataforma de código abierto, y por esto mismo ha presentado una acelerada evolución. Las mejoras constantes en el software han permitido la compatibilidad con diversos dispositivos de diferentes fabricantes, esto ha propiciado el desarrollo de terminales cada vez más avanzadas. Su diseño permite a los usuarios crear distintas aplicaciones en las que se pueden aprovechar diferentes características de los teléfonos móviles, tales como la memoria, el hardware, etc.

Una de las ventajas más significativas de Android es su plataforma que puede ser desarrollada en varios lenguajes. Sin embargo, Java se ha convertido en el más utilizado. Debido a que comparte con Android una característica primordial, su adaptabilidad. Lo anterior permite hacer de las clases creadas, clases reciclables, otorgando la oportunidad a cualquier ingeniero de mejorar o adaptar el código. Además, debido a su lenguaje multiplataforma, el ingeniero puede trabajar en Android o en cualquier otro sistema operativo.

Java abre la puerta al desarrollo de aplicaciones específicas, ya que permite agregar librerías, tales como OpenCV.

OpenCV se constituye por un código abierto (open source). Fue creada para el procesamiento de imágenes y visión por computadora escrita en C++, haciendo posible un trabajo directo con el procesador.Actualmente cuenta con interfaces para C++, C, Python, MATLAB y Java; funcionando en Windows, Linux, Mac OS y Android.

La innovación en los nuevos procesadores para los dispositivos móviles ha hecho posible desarrollar terminales más pequeñas pero con mayor potencia de procesamiento, representando la posibilidad de prescindir de un ordenador en aplicaciones específicas.

Ejemplo de ello es la capacidad de procesamiento del dispositivo móvil Motorola XT1032, la cual es idónea para la adquisición de imágenes y su procesamiento, necesarios para este proyecto.

Entonces, ¿por qué Android? Porque es un sistema operativo que no sólo puede emplearse para hacer llamadas, enviar mensajes o navegar en la red; sino que también se puede emplear como una potente herramienta que apoye a la ciencia y tecnología en el desarrollo de diversos e interesantes proyectos, tal como lo es la SRM.

CAPÍTULO 4

RECONOCIMIENTO DE IMÁGENES.

4.1 Visión por computadora.

Se puede decir que la visión es el principal sentido del ser humano y de la mayoría de los animales, ya que a través de este sentido se obtiene información acerca del entorno por medio de los ojos, esta información es representada por impulsos eléctricos que posteriormente son procesados por diferentes partes del cerebro para poder clasificar formas, colores, patrones y así podemos desenvolvernos en la tarea de navegación o en la realización de diversas actividades.

Al proceso de simular el sentido de la visión a través de computadoras se le denomina visión por computadora o visión artificial. Si bien se busca emular este sentido utilizando computadoras, aún falta mucho para lograrlo puesto que ojo humano es capaz de obtener una gran cantidad de información, procesarla e interpretarla en tan solo décimas de segundos y en condiciones de luz cambiantes, en contraste con lo anterior se puede decir que en la actualidad se ha logrado que a través del procesamiento de imágenes obtenidas por medio de una o varias cámaras, se reconozcan formas, colores, patrones y movimientos, pero el tiempo de procesamiento de estos procedimientos visuales elementales es elevado en comparación con el tiempo que le toma realizar estas tareas a la visión humana. No obstante, esta disciplina es ampliamente utilizada en varias ramas, ya que a diferencia de la visión natural, con la visión artificial se pueden obtener magnitudes físicas que para el humano suelen ser imprecisas, como áreas o longitudes. También se pueden obtener resultados satisfactorios en tareas rutinarias o que requieren mucha concentración, como lo son procesos de control de calidad.

Por esta razón se concluye que la visión por computadora complementa la visión natural del ser humano.

Algunas de las ramas en donde la visión por computadora es implementada son:

La medicina: en esta área la visión artificial es utilizada ampliamente para la detección de enfermedades mediante el procesamiento de imágenes médicas. La implementación más común dentro de la medicina la podemos observar en radiografías, tomografías y resonancias magnéticas.

La industria: la visión por computadora puede ser empleada en la industria dentro del área de inspección o control de calidad en la manufactura de un producto, ya que se puede verificar la presencia de características esperadas o dimensiones especificas por ejemplo radio, longitud , área, etcétera.

La robótica: su implementación va desde los robots que emplean la visión artificial para reconstruir entornos desconocidos con el fin de detectar la presencia de objetos y poder evadirlos o seguirlos, hasta los robots de servicio que logran reconocer rostros , figuras y objetos para manipularlos y realizar diversas tareas.

Un sistema de visión por computadora se divide en varios subsistemas o etapas, de las cuales se puede decir lo siguiente:

- Adquisición de imágenes: Esta parte es la encargada de adquirir la información del mundo real mediante dispositivos electrónicos que normalmente son una o varias cámaras que digitalizan la imagen, la cual es llevada hasta la memoria de un computador para ser procesada.
- Procesamiento: en esta etapa los datos adquiridos son manipulados de tal forma que se reduzca el ruido en la imagen o se realcen características de interés dentro de la misma.

- Reconocimiento: como su nombre lo indica, en esta etapa, los objetos o características de interés son distinguidas.
- Interpretación: en esta etapa se asigna un significado a los datos obtenidos en la etapa anterior.

4.2 Técnicas de reconocimiento de imágenes.

El termino reconocimiento de imágenes abarca una gran cantidad de técnicas dentro del campo de visión por computadora, puesto que en una imagen se pueden reconocer diferentes características como son color, forma, texturas o patrones, incluso se pueden combinar estas características para logar reconocer algo más complejo como lo es un rostro humano. Sin embargo, este trabajo está enfocado al reconocimiento de color y de forma, por lo tanto, se tratarán las técnicas destinadas a dicho reconocimiento.

Por otra parte, se puede decir que la materia principal del reconocimiento de imágenes es la imagen misma, ya que dentro de ella se encuentran todos los datos que se necesitan procesar para obtener un resultado satisfactorio en cuanto a las características buscadas en el procesamiento. Se puede definir a una imagen como una representación visual de objetos, personas o cosas. No obstante, el procesamiento llevado a cabo para nuestro fin es por medio de un procesador computacional, por lo tanto es necesario que se trabaje con imágenes digitales las cuales son adquiridas por medio de muestreo generalmente por medios electrónicos. Se puede definir a una imagen digital como "una función bidimensional que cuantifica la intensidad de la luz" [15]. Normalmente se representa como $I(x,y)$ en donde el valor de intensidad se obtiene haciendo referencia a los valores de x e y. Lo anterior se puede representar de forma matricial como se muestra en la siguiente ecuación:

$$I(x,y) = \begin{bmatrix} I(1,1) & I(2,1) & \cdots & I(N,1) \\ I(1,2) & I(2,2) & \cdots & I(N,2) \\ \vdots & \vdots & \ddots & \vdots \\ I(1,M) & I(2,M) & \ldots & I(N,M) \end{bmatrix} \quad (4)$$

En donde:

$I(x,y)$: Imagen.

$I(i,j)$: Valor indexado de un pixel que pertenece a la imagen I.

N, M: Dimensiones de la imagen.

Habitualmente se trabaja con imágenes en donde cada uno de los elementos de la matriz que la representa (píxeles[11]) toman valores que van desde 0 a 255, siendo el valor de 0 correspondiente al color negro, es decir cantidad de iluminación nula, el valor de 255 corresponde al color blanco o máxima iluminación. Sin embargo existen diferentes modelos que representan una imagen digital, a continuación se mencionan algunos.

4.2.1 Imágenes en escala de grises:

Se puede representar una imagen en escala de grises si a cada uno de los elementos de su matriz asociada se le asigna un valor que representa una escala de grises, normalmente estos valores van de 0 a 255 en donde el 0 corresponde al color negro y 255 corresponde al color blanco.

[11] Un pixel es la parte más pequeña que compone una imagen.

4.2.2 Imágenes binarias:

Este tipo de imágenes son las más fáciles de manipular debido a que los elementos que conforman a su matriz asociada solo pueden asumir dos valores: blanco o negro. Comúnmente se suelen asignar los valores de 0 para el color negro y 255 para el color blanco.

4.2.3 Imágenes RGB:

Este modelo se suele emplear para representar imágenes a color, en donde cada pixel es representado por las combinaciones de los colores rojo (*RED*), verde (*GREEN*) y azul (*BLUE*). Para poder representar este tipo de imágenes se utiliza una matriz tridimensional, MxNx3 en donde M y N representa el alto y el ancho de la imagen y cada dimensión de esta matriz representa la intensidad de cada uno de sus canales R, G, B.

4.2.4 Reconocimiento de color.

El color es una característica predominante en una imagen, y una de las características que más se busca reconocer en el ámbito de la robótica, ya sea con la finalidad de discriminar objetos y clasificarlos o con el fin de reconocer objetos que tengan cierta propiedad de color para poder dirigirse hacia ellos, como es el caso de la SRM. Las técnicas más utilizadas para lograr reconocer un color en una imagen generalmente consisten en: obtener la imagen mediante una o varias cámaras, después, realizar una discriminación para poder eliminar todos los colores excepto el de interés y finalmente, obtener una imagen binaria en donde los pixeles que la conforman (valores diferentes de cero) representan a los pixeles con el valor de color buscado. Como se puede apreciar la técnica de reconocimiento de color es algo sencilla, sin embargo, la técnica seleccionada para implementar en este proyecto tiene que ser robusta ya que el color en una imagen es una característica que se deforma fácilmente con las variaciones de luz en el ambiente.

A continuación se mencionan algunas técnicas para el reconocimiento de color en imágenes digitales.

4.2.4.1 Filtro RGB.

Este filtro utiliza los valores de RGB para enfocarse en los colores primarios de este modelo, es decir rojo, verde y azul, dependiendo del color seleccionado, todos los colores restantes serán suprimidos. Si por ejemplo, se selecciona el color verde, entonces:

$$R = 0 \tag{5}$$

$$G = (G - B) + (G - R) \tag{6}$$

$$B = 0 \tag{7}$$

El valor de G es normalizado referido al valor de verde máximo.

$$G = \left(\frac{G}{G_{max}}\right) \text{x } 255 \tag{8}$$

Al aplicar estas ecuaciones a cada uno de los valores de los pixeles de la imagen, todos los colores serán removidos a excepción del color seleccionado. La razón por la cual se descartó la posibilidad de utilizar este algoritmo en el proyecto fue que solo permite filtrar los colores rojo, verde o azul, limitando el reconocimiento del objeto de interés.

4.2.4.2 Segmentación por umbral en el espacio de colores RGB.

Esta técnica consiste en seleccionar un par de umbrales (umbral superior y umbral inferior) de acuerdo al color y la tonalidad que se busca reconocer. Estos umbrales corresponden a una tercia de valores, en donde cada valor representa a un canal en el modelo RGB. Con estos umbrales se aplica una técnica de procesamiento de imágenes llamada segmentación por umbral que consiste en que todos los pixeles de la imagen asuman dos valores diferentes p_0 o p_1 , dependiendo de la relación que guarden con el umbral, como se describe en la siguiente ecuación:

$$f_{th}(p) = \begin{cases} p_0 & si \quad p > p_{th_{inferior}} \,\&\&\, p < p_{th_{superior}} \\ p_1 & Cualquier\ otro\ caso. \end{cases} \quad (9)$$

En donde:

$p_0 = 255$
$p_1 = 0$

$p_{th_{inferior}}$ = Umbral inferior
$p_{th_{superior}}$ = Umbral superior

De esta manera se eliminan todos los pixeles de la imagen que queden fuera de los umbrales, obteniendo como resultado una imagen binaria en donde cada pixel con valor 255 (color blanco) representa a un pixel con el color seleccionado.

Con esta técnica se logran reconocer una amplia gama de colores con diferentes tonalidades. Como se menciona, el color que se busca reconocer está definido por los umbrales seleccionados, de tal modo que es primordial la selección de los umbrales adecuados, ya que si no se hace la sincronización de los umbrales correctamente no se logrará la discriminación del color seleccionado. Por esta razón este método fue descartado.

4.2.4.3 Segmentación por umbral en el espacio de colores HSV.

Esta técnica consiste en transformar la imagen al espacio de colores HSV en el cual la información de la imagen es representada en tres componentes Tonalidad (*Hue*), Saturación (*Saturation*) y Valor (*Value*). El modelo HSV es tradicionalmente representado por una pirámide invertida (Figura 4.1) en donde el eje vertical representa el valor (V), la distancia horizontal tomando como referencia el eje V, corresponde a la saturación (S), mientras que el ángulo alrededor de V es la tonalidad (H). Al igual que en el modelo RGB, para representar una imagen en el espacio de colores HSV se utiliza una matriz de MxNx3 en donde M y N representan las dimensiones de la imagen, y cada dimensión de la matriz hace referencia a cada uno de los componentes que comprenden este modelo.

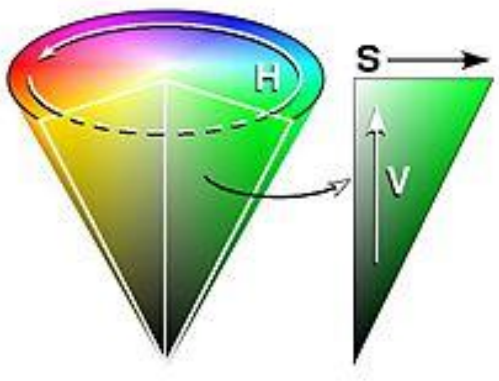


Figura 4.1 Espacio de colores HSV [16].

Después de haber obtenido la imagen en el espacio de colores HSV, se aplica la técnica de segmentación por umbral, en donde cada umbral es una tercia de valores que corresponden a las componentes H, S y V. A diferencia de realizar la segmentación por umbral con una imagen en el modelo RGB, en este caso la selección de los umbrales es mucho más sencilla, debido a que en el espacio HSV, el valor de tonalidad (H) contiene información del color, la saturación (S) corresponde a la cantidad con que este color se mezcla con el blanco, y el valor (V) corresponde a la cantidad con que este color se mezcla con el negro, de tal modo que, para discriminar correctamente un color, solo se tiene que elegir adecuadamente el valor de la tonalidad (H), ya que las componentes de saturación y valor pueden variar de acuerdo a las condiciones de iluminación del entorno.

Se decidió implementar este método en el proyecto, puesto que se logra un reconocimiento de color robusto gracias a que el color buscado queda perfectamente definido con los valores de los umbrales de tonalidad (H), los valores de saturación y valor se eligen de acuerdo a las condiciones de iluminación, a diferencia de la técnica de segmentación en el espacio RGB, en donde si cambian las condiciones de iluminación, sería necesario cambiar todos los valores de los umbrales hasta lograr una sincronización adecuada para lograr reconocer el color buscado.

4.2.5 Reconocimiento de forma.

Para hacer más robusto el reconocimiento del objeto de interés en el proyecto, el cual es una pelota de un color específico, se agrega la capacidad de reconocer la forma de la pelota, es decir una circunferencia.

La detección de bordes juega un papel importante dentro del reconocimiento de formas en las imágenes digitales, puesto que es más eficiente buscar figuras en solo los bordes que en la imagen completa. Un borde se define como los puntos que pertenecen a una imagen en donde la intensidad de estos puntos cambia drásticamente respecto a los puntos cercanos. Con el transcurso del tiempo y el desarrollo del campo de visión por computadora, han surgido una gran cantidad de técnicas para lograr el reconocimiento de bordes dentro de una imagen.

Debido a que el cambio drástico de intensidad es lo que define el borde en una imagen, los métodos utilizados para determinar esta propiedad están basados en aproximar la primera o segunda derivada de los pixeles que forman una imagen. Existen métodos en donde se implementa una matriz de 3x3 como máscara para aproximar la primera derivada en imágenes y poder detectar los bordes como el llamado operador de Prewitt o el operador de Sobet, también existen filtros basados en el momento como el Laplaciano o el Gaussiano. Para este proyecto se decidió utilizar el algoritmo de Canny para detectar los bordes, debido a que este algoritmo, a diferencia de otras técnicas, busca minimizar el número de bordes falsos, cierra los contornos evitando errores en su ejecución, así como una mejor localización de bordes en la imagen y entregar una imagen con el borde de

un pixel. El algoritmo de Canny es un método basado en el gradiente y se aplica en direcciones y resoluciones diferentes.

Por otra parte, las técnicas para lograr el reconocimiento de formas de un objeto, van desde complejos algoritmos genéticos hasta técnicas que se centran en el estudio de los bordes obtenidos en una imagen como lo es el método de contornos activos el cual consiste en obtener a partir de los bordes, una serie de ecuaciones llamadas funciones de energía, las cuales definen características que posee el contorno como son la distancia entre puntos y la curvatura. Este método es novedoso, debido a que logra obtener información de formas contenidas en una imagen de una manera diferente comparada con los métodos más utilizados. La principal desventaja de este método es que requiere un tiempo de procesamiento elevado, por lo tanto no resulta una buena opción utilizarlo en aplicaciones de procesamiento en línea como lo es este proyecto. La técnica seleccionada para implementar en este trabajo se trata de la transformada de Hough, esta técnica es la más utilizada en el área de visión artificial para el reconocimiento de líneas y circunferencias, debido a su sencillez y eficiencia, por lo tanto existe mucha información acerca de este método, a continuación se explica brevemente.

4.2.5.1 Transformada de Hough.

Este método nos permite encontrar formas paramétricas como líneas, círculos y elipses a partir de la distribución de píxeles en una imagen. Se ejemplificara primeramente la transformada de Hough para la detección de líneas.

Una línea se puede representar mediante la utilización de dos parámetros reales como se indica en la siguiente ecuación:

$$y = kx + d \tag{10}$$

En donde k es la pendiente y d es el punto del eje y en donde dicha línea lo intercepta. Se sabe que una línea que pasa por dos diferentes puntos $p_1 y\ p_2$ debe satisfacer las siguientes ecuaciones:

$$y_1 = kx_1 + d \text{ y } y_2 = kx_2 + d \tag{11}$$

El objetivo es determinar los parámetros k y d de una línea la cual pasa por diferentes puntos de los bordes de una imagen. La transformada de Hough resuelve este problema de una manera distinta al producir todas las líneas que pasan sobre un pixel el cual corresponde a un borde en la imagen. Cada línea L_p la cual pasa por el punto $p_0 = (x_0, y_0)$ posee la siguiente ecuación:

$$L_p: y_0 = kx_0 + d \tag{12}$$

En donde los valores de k y d son variados para encontrar a todas las rectas que pasen por el punto p_0. Para un determinado valor de k se produce la correspondiente solución en función de d:

$$d = y_0 - kx_0 \tag{13}$$

La cual es una función lineal, donde k y d son las variables que definen el espacio de parámetros también llamado el espacio de Hough y x_0 e y_0 son las constantes en la función. El conjunto de valores $\{(k, d)\}$ describe los parámetros de todas las posibles rectas que contiene al punto:

$$p_0 = (x_0, y_0) \tag{14}$$

En otras palabras, cada punto P en el espacio de parámetros de la imagen corresponde a una recta en el espacio de parámetros de Hough, por lo tanto se está interesado en aquellos puntos en donde las rectas establecidas en el espacio de Hough se interceptan, que corresponden a los valores de k y d que representan a la recta en el

espacio de parámetros de la imagen que pasa por los puntos que formularon las rectas en el espacio de parámetros de Hough. Mientras más rectas en el espacio de parámetros de Hough se corten significa que la línea en el espacio de la imagen está formada por ese número de puntos.

Las circunferencias representadas en el espacio bidimensional necesitan tres parámetros para quedar completamente definidas los cuales son el centro (u, v) y el radio ρ , un punto $p = (x, y)$ que forma parte de la circunferencia tiene que cumplir la siguiente ecuación:

$$\rho^2 = (x - u)^2 + (y - v)^2 \quad (15)$$

Por consiguiente el espacio de parámetros de Hough queda definido por tres parámetros u, v, ρ. De forma análoga que para cada punto P en el espacio de parámetros de la imagen, corresponde a una recta en el espacio de parámetros de Hough, para el proceso de buscar circunferencias, cada punto en el espacio de parámetros de la imagen corresponde a un cono en el espacio de parámetros de Hough (Figura 4.2) , después de transformar todos los puntos que conforman el contorno de la figura que contiene la imagen, los puntos pertenecientes a un circulo darán como resultado una intersección de estas superficies cónicas formando un área circular, esta área está caracterizada por un centro (u, v) y un radio ρ, que corresponden a los parámetros de la circunferencia buscada.

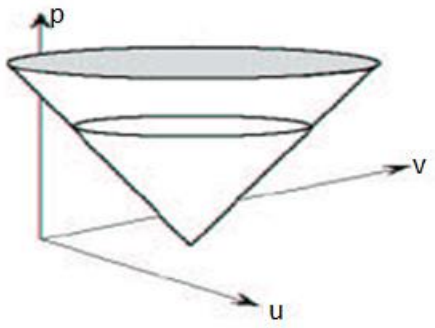


Figura 4.2 Espacio de parámetros de Hough [17].

Una forma más eficiente de reconocer circunferencias empleando el algoritmo de Hough, consiste en representar una circunferencia como la describen las siguientes ecuaciones:

$$x = u + \rho \cos\theta \quad (16)$$

$$y = v + \rho \sin\theta \quad (17)$$

Si consideramos $\rho = R$ como una constante, lo cual implica que buscamos circunferencias de un radio específico R , el espacio de parámetros de Hough, estará conformado por las siguientes ecuaciones:

$$u = x - R \cos\theta \quad (18)$$

$$v = y - R \sin\theta \quad (19)$$

De esta forma el espacio de parámetros de Hough se reduce en uno y queda definido por los dos parámetros u y v, esto hace que para cada punto P en el espacio de parámetros de la imagen corresponda a una circunferencia de radio R y centro (u, v) en el espacio de parámetros de Hough. El punto (u, v) en donde estas circunferencias

se interceptan corresponde al centro (x, y) de la circunferencia buscada, lo anterior está reflejado en la figura 4.3.

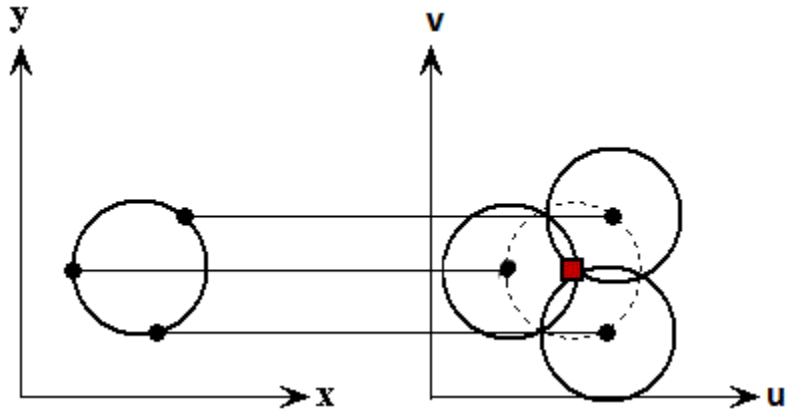


Figura 4.3 Relación del espacio de parámetros de la imagen con el espacio de parámetros de Hough [17] .

De esta forma se concluye que para una de las finalidades de este proyecto la cual es reconocer una pelota de un color específico para que después sea seguida por la SRM, se implementará primeramente, una técnica de reconocimiento de color utilizando segmentación por umbral en el espacio de colores HSV, la cual nos permitirá suprimir todos los colores en la imagen captada a excepción del color de la pelota, obteniendo una imagen binaria. Después se aplicará el algoritmo de Canny para obtener los bordes en la imagen y finalmente se aplicará la transformada de Hough para buscar circunferencias dentro de la imagen y obtener como datos de salida el centroide de la pelota, el cual nos servirá para estimar la dirección hacia donde se tiene que dirigir la SRM, así como el radio de la pelota el cual usaremos para calcular el área y estimar la distancia virtual del objetivo.

4.3 Innovaciones a través del tiempo.

Gracias a los avances en el área de la electrónica y la computación, hoy en día contamos con procesadores cada vez más potentes, gracias a esto el campo de la visión por computadora ha logrado avances muy interesantes, específicamente hablando en el ámbito del reconocimiento de imágenes, contamos con sistemas que son capaces de reconocer rostros comparando los resultados con bases de datos para lograr identificar a la persona en cuestión, con el fin de crear sistemas de seguridad controlando el acceso a áreas restringidas. El reconocimiento de rostros también es usado con fines de ley y vigilancia identificando sospechosos, esta técnica también es utilizada para la validación de usuarios en tarjetas inteligentes o incluso para el entretenimiento, un ejemplo de esto es la conocida red social Facebook que es capaz de reconocer el rostro de personas en fotografías. Los avances dentro de este ámbito son numerosos, por ejemplo en el artículo [18] se menciona un sistema propuesto para el reconocimiento de rostros que consiste en combinar técnicas en el dominio del espacio y en el dominio de la frecuencia, para ello este sistema selecciona el área de interés mediante la transformada de Ripplet la cual es una generalización de la transformada de Curvelet con dos parámetros extras, la transformada de Curvelet permite representar a la imagen en escalas y ángulos diferentes para poder aproximar curvas utilizando pocos coeficientes y así obtener descriptores del rostro, estos descriptores se utilizan para entrenar una red neuronal artificial enfocada al reconocimiento de rostros.

Otra novedosa implementación del reconocimiento de imágenes se puede apreciar en el buscador de Google el cual ha implementado un sistema que permite hacer búsquedas en la red a través de imágenes, este sistema funciona reconociendo puntos de interés en la imagen, posteriormente estos puntos son procesados para

identificar como se relacionan entre ellos y por último el sistema busca estos datos en una enorme base de datos para encontrar el objeto buscado.

Existen innumerables proyectos que aportan algo nuevo al campo de visión por computadora, estos avances se han logrado gracias a la combinación de varias técnicas y herramientas de la ingeniería, esto lo podemos observar en los ejemplos anteriores en donde el reconocimiento de imágenes se apoya de redes neuronales artificiales y bases de datos para lograr un sistema de reconocimiento robusto y preciso.

CAPÍTULO 5

ALGORITMOS DE NAVEGACIÓN.

Al referirse a plataformas móviles, es imposible evitar pensar en una pregunta: ¿Cómo lograr la evasión de obstáculos? Para lograrlo existen diversas técnicas, sin embargo dos métodos pueden ser idóneos.

5.1 Campos Potenciales

Este método comenzó a plantearse desde finales de la década de los años 70. Sin embargo, fue a partir de la publicación del trabajo realizado por Oussama Khatib en 1985 en donde se hace uso de los campos potenciales para conseguir la evasión de obstáculos.

Khatib parte de la necesidad de lograr que un robot opere en ambientes no estructurados, a la vez que lo provee de la capacidad de modificar y ajustar sus movimientos en tiempo de ejecución para evadir obstáculos.

El método reactivo, utilizado por Khatib, busca generar fuerzas de atracción y repulsión dentro del ambiente de trabajo del robot para guiarlo a la meta, en donde ésta última representa una influencia atractiva para el robot, mientras que tiende a alejarse de cada obstáculo, con la finalidad de evitar colisiones.

En este método se modela el espacio como un campo escalar en el que a cada punto del espacio se le asigna un potencial. El espacio se modela de modo que el punto al que se quiere llegar sea el que tenga el potencial más bajo, y los obstáculos a evadir tengan un potencial muy alto; de esta manera el robot debe moverse hacia el punto de menor potencial, como se observa en la Figura 5.1.

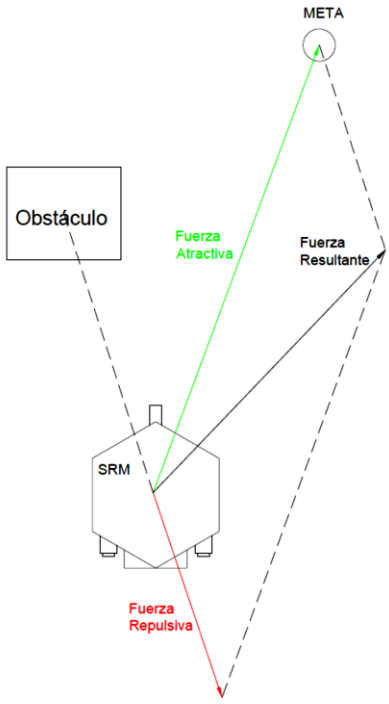


Figura 5.1 Fuerzas involucradas en los Campos Potenciales.

Para esto, el robot es representado como una partícula bajo la influencia de un campo potencial escalar U, definido como

$$U = U_{att} + U_{rep} \tag{20}$$

En donde U_{att} y U_{rep} son los campos atractivo y repulsivo, respectivamente.

Entonces el campo representa una energía potencial artificial construida de modo tal que el robot se mueva en la trayectoria deseada. El campo vectorial de las fuerzas artificiales $\boldsymbol{F}(\boldsymbol{q})$ está dado por el gradiente de U

$$\boldsymbol{F}(\boldsymbol{q}) = -\boldsymbol{\nabla} U_{att} + \boldsymbol{\nabla} U_{rep} \tag{21}$$

Donde ∇U es el vector gradiente de U para la posición del robot $\boldsymbol{q}(x, y)$ en un mapa bidimensional. De este modo, $\boldsymbol{F}$ es definido como la suma de dos vectores $\boldsymbol{F}_{att}(\boldsymbol{q}) = -\boldsymbol{\nabla} U_{att}$ y $\boldsymbol{F}_{rep}(\boldsymbol{q}) = \boldsymbol{\nabla} U_{rep}$, tal como se muestra en la siguiente ecuación.

$$\boldsymbol{F}(\boldsymbol{q}) = \boldsymbol{F}_{att}(\boldsymbol{q}) + \boldsymbol{F}_{rep}(\boldsymbol{q}) \tag{22}$$

La forma general de los campos potenciales fue propuesta por Kathib de la siguiente manera:

- El campo potencial atractivo se expresa de la siguiente manera.

$$U_{att} = \frac{1}{2} \xi d^2 \tag{23}$$

Donde

$d = |q - q_a|$ q es la posición del robot

q_a es la posición del punto de atracción ξ *es una constante ajustable* (*constante de atracción*)

Por consiguiente la fuerza atractiva se expresaría de la siguiente manera.

$$\boldsymbol{F_{att}}(\boldsymbol{q}) = -\boldsymbol{\nabla} U_{att} = -\xi(\boldsymbol{q} - \boldsymbol{q_a}) \tag{24}$$

Donde

q *es la posición del robot* ξ *es una constante ajustable* (*constante de atracción*)

q_a *es la posición del punto de atracción*

- Así mismo el campo potencial repulsivo está expresado como

$$U_{rep} = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{d} - \frac{1}{d_0}\right)^2, & si\ d \le d_0 \\ 0, & si\ d > d_0 \end{cases} \tag{25}$$

Donde

$d = |q - q_o|$ d_0 *es la distancia de influencia*

q_o es la posición del obstáculo η *es una constante ajustable* (*constante de repulsión*)

q es la posición del robot

Entonces la correspondiente fuerza repulsiva es

$$\boldsymbol{F}_{rep}(\boldsymbol{q}) = \boldsymbol{\nabla} U_{rep} = \begin{cases} \eta\left(\frac{1}{d} - \frac{1}{d_0}\right)\left(\frac{\boldsymbol{q} - \boldsymbol{q_o}}{d^3}\right), & si\ d \le d_0 \\ 0, & si\ \ d > d_0 \end{cases} \tag{26}$$

Donde

$d = |q - q_o|$ d_0 *es la distancia de influencia*

q_o es la posición del obstáculo η *es una constante ajustable* (*constante de repulsión*

q es la posición del robot

Con lo anterior es de notar que la fuerza de atracción está en función de las coordenadas del punto a donde se quiere llegar, mientras que la fuerza de repulsión está en función de las posiciones de los objetos que se desea evadir.

En general, podemos resumir que sólo se utiliza una fuerza de atracción, y la fuerza de repulsión es la suma de las fuerzas generadas por cada objeto en el ambiente.

En las ecuaciones (24) y (26), con base experimental y con el propósito de mejorar el desempeño del algoritmo, en [19] se propone modificar las mismas.

Es entonces cuando la ecuación (24) es normalizada para generar una fuerza de atracción independiente de la distancia entre el robot y el punto de meta; obteniendo así lo siguiente

$$\boldsymbol{F_{atr}}(\boldsymbol{q}) = -\boldsymbol{\nabla} U_{atr} = -\xi(\boldsymbol{q} - \boldsymbol{q_a})\frac{1}{\boldsymbol{q} - \boldsymbol{q_{atr}}} \tag{27}$$

Donde

q *es la posición del robot* ξ *es una constante ajustable* (*constante de atracción*)

q_{atr} *es la posición del punto de atracción*

Por otro lado en la ecuación (26), la fuerza de repulsión F_{rep} es definida como

$$\boldsymbol{F}_{rep}(\boldsymbol{q}) = \boldsymbol{\nabla} U_{rep} = \begin{cases} \eta\sqrt{\left(\frac{1}{d} - \frac{1}{d_0}\right)}\left(\frac{\boldsymbol{q} - \boldsymbol{q_{obs}}}{\left|q - q_{obs}\right|^3}\right), & si\ d \le d_0 \\ 0, & si\ \ d > d_0 \end{cases} \tag{28}$$

Donde

q_{obs} es la posición del obstáculo

q es la posición del robot

d_0 *es la distancia de influencia*

η *es una constante ajustable* (*constante de repulsión*)

η *es una constante ajustable* (*constante de repulsión*)

A medida que el robot se acerca a un obstáculo la fuerza de repulsión aumenta en la dirección opuesta a la trayectoria del robot. Si la distancia del robot al obstáculo es mayor a d_0, ese obstáculo no surte efecto alguno en la trayectoria del robot. La raíz cuadrada de $\left(\frac{1}{d} - \frac{1}{d_0}\right)$ en la ecuación (28) hace posible una respuesta más adecuada para nuestros fines, pues provee un ligero incremento de la fuerza de repulsión, permitiendo así una evasión de obstáculos más segura.

5.2 Redes Neuronales Artificiales.

Las redes neuronales artificiales (RNA) son modelos computacionales que intentan reproducir el comportamiento del cerebro, imitando las estructuras de las que se compone el sistema nervioso, basándose en el aprendizaje a través de la experiencia para que consecuentemente se extraiga conocimiento a partir de la misma. Estas redes se caracterizan por dos aspectos importantes:

a) Topología de las Redes Neuronales.

La arquitectura de las redes neuronales consiste en la organización y disposición de las neuronas formando capas más alejadas o cercanas a la entrada y salida de la red. En este sentido, los parámetros fundamentales de la red son: el número de capas, el número de neuronas por capa, el grado de conectividad y el tipo de conexiones entre neuronas.

b) Mecanismo de Aprendizaje.

El aprendizaje es el proceso por el cual una red neuronal modifica sus pesos en respuesta a una información de entrada. Los cambios que se producen durante el proceso de aprendizaje se reducen a la destrucción, modificación y creación de conexiones entre las neuronas. La creación de una nueva conexión implica que el peso de la misma pasa a tener un valor distinto de cero, una conexión se destruye cuando su peso pasa a ser cero. Se puede afirmar que el proceso de aprendizaje ha finalizado, cuando los valores de los pesos permanecen estables.

Al igual que el ser humano aplica el conocimiento ganado a través de la experiencia a nuevos problemas o situaciones, una RNA toma como ejemplos los problemas resueltos para construir un sistema que toma decisiones y realiza clasificaciones. Los problemas adecuados para las RNA son aquellos que no tienen solución computacional precisa o que requieren algoritmos muy extensos.

Por tanto, una RNA puede ser entendida como una interconexión entre neuronas artificiales para formar una red que lleve a cabo un proceso o decisión. Entonces una neurona artificial puede ser considerada como la unidad básica de procesamiento, la cual puede localizarse en alguna de las capas, ya sea en la de entrada, de salida o en una oculta, es decir, en una capa cuyas entradas y salidas se encuentran dentro del sistema, impidiendo una interacción directa con el entorno de entrada o salida.

Una capa o nivel es un conjunto de neuronas cuyas entradas provienen de la misma fuente y cuyas salidas se dirigen al mismo destino. Existen redes de una sola capa y redes de varias capas, mono capa y multicapa, respectivamente.

En una red neuronal ya entrenada, las conexiones entre neuronas tienen un determinado peso (peso sináptico). Un ejemplo de una neurona sobre la que convergen conexiones de diferente peso sináptico (W_i) se observa en la Figura 5.2.

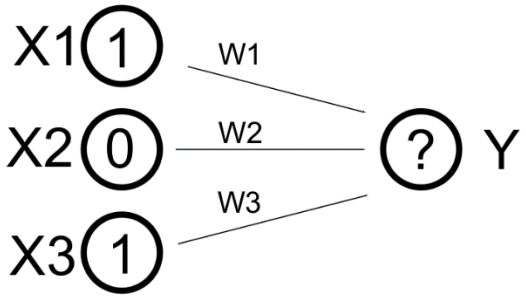
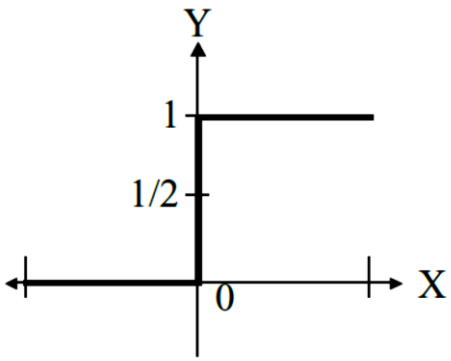


Figura 5.2 Conexiones de diferente peso sináptico ($W_1 > W_2 > W_3$) convergen sobre la misma neurona Y.

De acuerdo a las entradas externas, es decir, a los estímulos de entrada y a los diferentes pesos asignados, es definida una regla de propagación, que básicamente se trata de un cálculo que determina la entrada efectiva que tendrá la siguiente neurona. Generalmente la regla de propagación consiste en una suma ponderada de todas las entradas recibidas, es decir, de las entradas multiplicadas por el peso o valor de las conexiones, aunque existen algunas otras.

Cada neurona posee un estado de activación, delimitado por la función de activación. Dicho de otro modo, la función de activación se encarga de determinar el nivel o estado de activación de la neurona en función de la entrada efectiva.

En un principio se pensó que las neuronas usaban una función de umbral, es decir, que permanecían inactivas y se activaban sólo si la estimulación total superaba cierto valor límite; este comportamiento se puede modelar con una función escalón unitario: la función devuelve 0 por debajo del valor crítico (umbral) y 1 por encima, lo cual se muestra en la Figura 5.3.



$$F_k(x) = \begin{cases} 1 & \text{si } x \geq 0 \\ 0 & \text{si } x < 0 \end{cases}$$

Figura 5.3 Función escalón con valor umbral igual a cero.

Descubrimientos posteriores comprobaron que las neuronas emitían impulsos de actividad eléctrica con una frecuencia variable, dependiendo de la intensidad de la estimulación recibida, y que tenían cierta actividad hasta en reposo, con estimulación nula. Estos descubrimientos llevaron al uso de funciones no lineales con esas características, como la función sigmoidal, que es una función continua pero con un perfil parecido al escalón de una función de umbral.

Las RNA pueden aplicarse a diversas disciplinas, tales como:

- Electrónica: Predicción de secuencia de códigos. Distribución de elementos en CI. Control de procesos. Análisis de fallas. Visión artificial. Reconocimiento de voz.
- Finanzas: Tasación real de los bienes. Asesoría de préstamos. Previsión en la evolución de precios. Seguimiento de hipotecas. Análisis de uso de línea de crédito. Evaluación del riesgo en créditos Interpretación y reconocimiento de firmas.
- Manufactura: Control de la producción y del proceso. Análisis y diseño de productos. Diagnóstico de fallas en el proceso y maquinarias. Identificación de partículas en tiempo real. Inspección de calidad mediante sistemas visuales.
- Medicina: Análisis de células portadoras de cáncer mamario. Análisis de electroencefalograma y de electrocardiograma. Reconocimiento de infartos mediante ECG. Optimización en tiempos de trasplante.
- Seguridad: Códigos de seguridad adaptivos. Criptografía. Reconocimiento de huellas digitales.
- Voz: Reconocimiento de voz. Compresión de voz. Clasificación de vocales. Transformación de texto escrito a voz.
- Automóviles: Sistemas de piloto automático. Detección de fallas por reconocimiento externo de vibraciones.
- Robótica: Control dinámico de trayectoria. Robots elevadores. Controladores. Sistemas ópticos.

Entre las ventajas que posee la RNA frente a otros sistemas de procesamiento de información son:

- Pueden sintetizar algoritmos a través de un proceso de aprendizaje.
- Para utilizar la tecnología neuronal no es necesario conocer los detalles matemáticos. Sólo se requiere estar familiarizado con los datos del trabajo.
- Son robustas, pueden fallar algunos elementos de procesamiento pero la red continúa trabajando; esto es contrario a lo que sucede en programación tradicional.

Sin embargo también existen desventajas:

- Las RNA se deben entrenar para cada problema. Además, es necesario realizar múltiples pruebas para determinar la arquitectura adecuada. El entrenamiento es largo y puede consumir varias horas de la computadora (CPU).
- Debido a que las redes se entrenan en lugar de programarlas, éstas necesitan muchos datos.
- Las RNA representan un aspecto complejo para un observador externo que desee realizar cambios.
- Para añadir nuevo conocimiento es necesario cambiar las iteraciones entre muchas unidades para que su efecto unificado sintetice este conocimiento.

CAPÍTULO 6

PLANTEAMIENTO DEL SISTEMA.

6.1 Sistema general.

En el capítulo uno de este trabajo se mencionó el objetivo del mismo el cual es: diseñar un sistema de navegación para la SRM, con capacidad de evadir obstáculos que se presenten durante su recorrido en tiempo real, apoyándose de un sistema de visión artificial para reconocer y llegar a su objetivo final con éxito, utilizando como base un Smartphone con sistema operativo Android y el algoritmo de campos potenciales.

En este capítulo se describe el desarrollo para lograr dicho objetivo. El sistema general que se implementó está dividido en varias etapas o procesos, los cuales se describen a continuación brevemente y es posible observarlo en la Figura 6.1, puesto que a lo largo del capítulo se describirán a fondo:

1. Adquisición de datos: los datos adquiridos son imágenes captadas por la cámara del dispositivo móvil, este dispositivo se encuentra acoplado sobre la base del robot por medio de un soporte para *smartphones*.
2. Procesamiento: el objetivo es reconocer una pelota de color verde, la cual es la meta a la que se debe de dirigir la SRM. Para lograr dicho objetivo se aplican una serie de técnicas de visión por computadora sobre la imagen captada por el dispositivo móvil: transformación al espacio HSV, segmentación por umbral, filtros de erosión y dilatación, algoritmo de Canny y transformada de Hough.
3. Transmisión de datos: los datos obtenidos como resultado del procesamiento (los cuales contienen información de la pelota buscada) son transmitidos por el dispositivo móvil vía Bluetooth, estos mismos datos son recibidos por un módulo Bluetooth, que a su vez, está conectado al puerto serie de un Microcontrolador.
4. Algoritmo de navegación: El microcontrolador ATMEGA 328 está programado con el algoritmo de campos potenciales; este algoritmo usa los datos provenientes del procesamiento de imágenes en conjunto con las lecturas de sensores ultrasónicos acoplados en la periferia del robot para determinar hacia dónde se tiene que dirigir la SRM para llegar a su objetivo evadiendo obstáculos que se presenten en su recorrido.

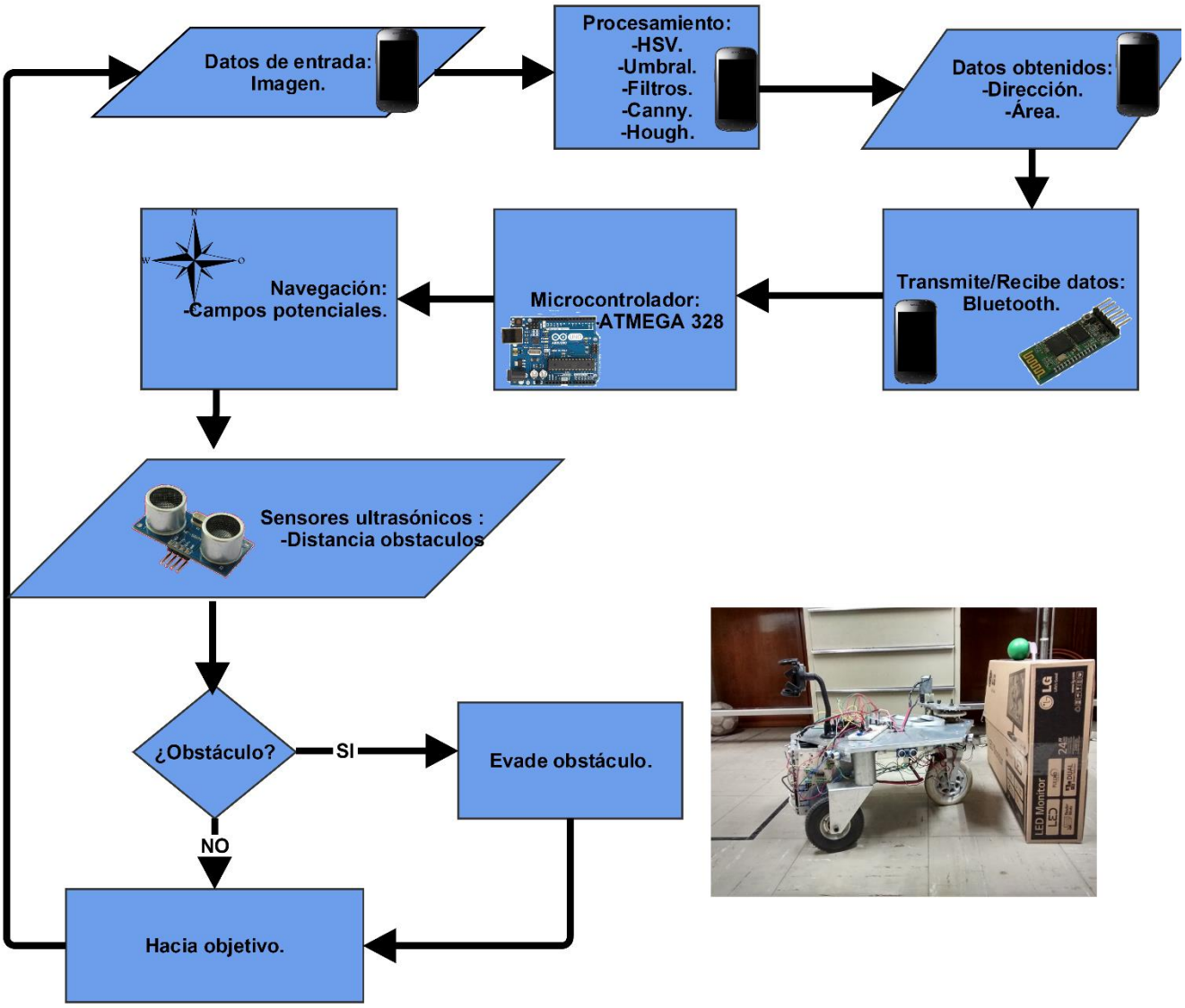


Figura 6.1 Diagrama general del sistema.

En el siguiente apartado se describe el proceso realizado para el reconocimiento de la pelota, posteriormente, se describe el desarrollo de la aplicación móvil implementando el reconocimiento mencionado, después se analiza la implementación del algoritmo de navegación que le permite a la SRM dirigirse hacia su objetivo.

6.2 Implementación del reconocimiento de color y de forma.

El proceso utilizado para el reconocimiento de color y de forma representado en el diagrama de flujo de la Figura 6.2, consiste en utilizar varios algoritmos de procesamiento de imágenes descritos en el capítulo cuatro: transformación al espacio HSV, segmentación por umbral, filtros de erosión y dilatación, algoritmo de Canny y transformada de Hough. Se decidió implementar dicho proceso en un dispositivo móvil con sistema operativo Android, debido a que los recursos y capacidades de procesamiento de estos dispositivos actualmente se

asemejan a un equipo de cómputo ordinario. Así mismo se utilizó la librería de OpenCv[12], que es una librería de código abierto desarrollada por Intel para la visión por computadora, OpenCv es muy utilizado en ámbitos como el entretenimiento y la robótica.

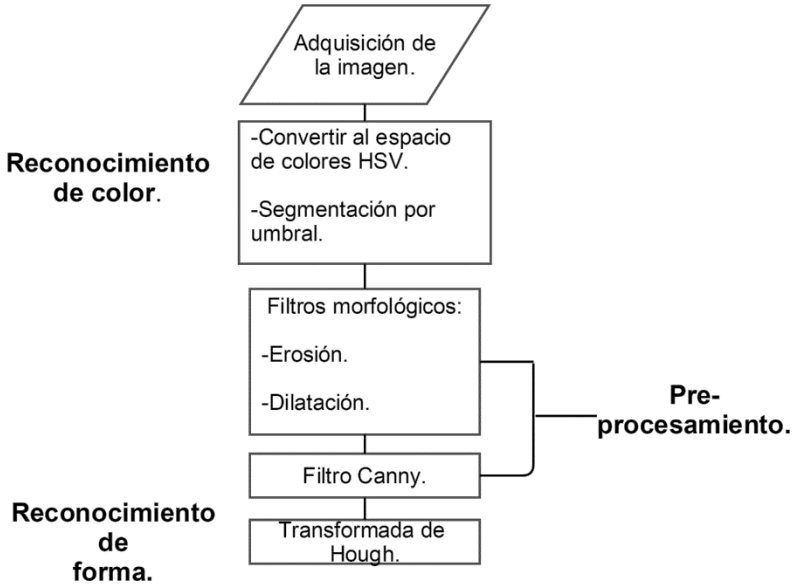


Figura 6.2 Reconocimiento de color y de forma.

6.2.1 Adquisición de la imagen.

El cuadro a procesar es adquirido por la cámara trasera del dispositivo móvil que está acoplado sobre la base del robot, como se menciona en el capítulo dos, este dispositivo es un teléfono móvil Motorola XT1032. La resolución de la cámara de este teléfono es de 5 Mega píxeles, analizar y procesar dicha cantidad de elementos (pixeles) implica un considerable tiempo de procesamiento, por lo que se decidió hacer un escalamiento en la imagen. Por lo tanto las dimensiones de la imagen a procesar son: 649 x 364 pixeles (modo horizontal).

6.2.2 Reconocimiento de color.

Para realizar el reconocimiento de color, el cuadro original obtenido de la cámara del dispositivo móvil es transformado al espacio de colores HSV, como se mencionó en el capítulo cuatro, en el modelo HSV la información de la imagen es representada en tres componentes Tonalidad (Hue), Saturación (Saturation) y Valor (Value). Para logar esta transformación se hace uso del método *cvtColor(scr,dst,code)* de la librería Opencv, este método recibe tres parámetros, *scr* es el objeto de tipo *Mat* origen, este objeto es una representación matricial de la imagen original, *dst* es el objeto *Mat* destino, en este objeto se guardará la imagen resultante y *code* es el código de conversión de espacios para este caso se trata de la constante *CV_RGB2HSV.* Las operaciones que realiza este método para implementar la transformación de espacios son:

$$V = \max(R, G, B) \qquad (29)$$

[12] http://opencv.org/

$$S = \begin{cases} \dfrac{V - \min(R,G,B)}{V} \; si\; V \neq 0 \\ 0 \quad Cualquier\; otro\; caso \end{cases} \qquad (30)$$

$$H = \begin{cases} \dfrac{60(G-B)}{V - \min(R,G,B)} si\; V = R \\ 120 + \dfrac{60(B-R)}{V - \min(R,G,B)} \; si\; V = G \\ 240 + \dfrac{60(R-G)}{V - \min(R,G,B)} \; si\; V = B \end{cases} \qquad (31)$$

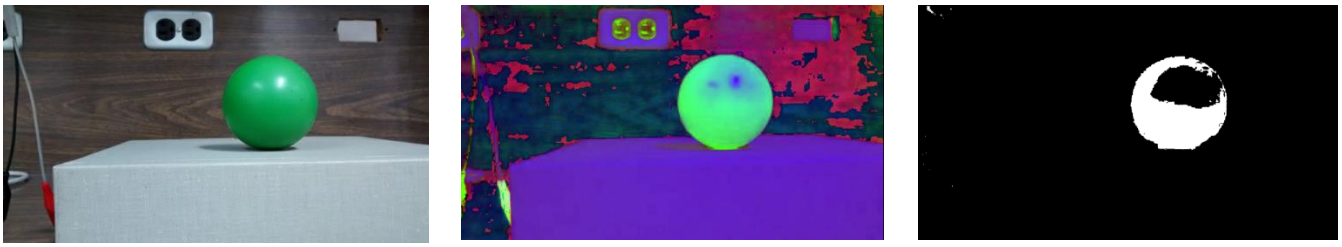
Los valores de salida son: si $H < 0$ entonces $H = H + 360,\ 0 \leq V \leq 1,\ 0 \leq S \leq 1,\ 0 \leq H \leq 360$, estos valores son convertidos de acuerdo al tipo de datos manejados, para este proyecto en donde se trabaja con imágenes de 8 bits los valores que pueden tomar las componentes son: $0 \leq V \leq 255$, $0 \leq S \leq 255$, $H = \frac{H}{2}$

Por lo tanto los valores que pueden tomar las componentes en el espacio HSV son de 0-180 para H, de 0-255 para S y 0-255 para V. El objetivo de aplicar esta transformación es de realizar una discriminación de color más eficiente comparada con la que se obtendría al hacerla sobre la imagen original que se encuentra en el espacio de colores RGB, debido a que en el espacio HSV, el valor de tonalidad (H) contiene información del color, la saturación (S) corresponde a la cantidad con que este color se mezcla con el blanco y el valor (V) corresponde a la cantidad con que este color se mezcla con el negro, de tal modo que para hacer referencia a un color solo se tiene que elegir correctamente el valor de la tonalidad , puesto que los valores de saturación y valor pueden variar de acuerdo a las condiciones de iluminación del entorno.

Posteriormente a la imagen en el espacio de colores HSV se le aplica una técnica de procesamiento de imágenes conocida como segmentación por umbral, la cual consiste en recorrer todos los elementos de la matriz que representa a la imagen y asignarles dos posibles valores 0 o 255, estos valores dependen de la relación que los elementos guarden con un par de constantes denominadas umbrales. Estas constantes contienen información del color buscado, por lo tanto los umbrales son un arreglo de valores compuesto por las tres componentes (tonalidad, saturación y valor) del espacio HSV. El método de la librería de OpenCv utilizado para esta técnica es *inRange(scr,lowth, highth,dst)* en donde *scr* es el objeto *Mat* de origen, *lowth* es el arreglo que representa el umbral inferior, *highth* es el arreglo que representa el umbral superior y *dst* es el objeto de tipo *Mat* en donde se guarda la imagen resultante, este método realiza la siguiente operación sobre cada uno de los elementos de la imagen:

$$P_{(x,y)} = \begin{cases} 255 \; si\; lowerth \leq P_{(x,y)} \leq upperth \\ 0 \qquad\qquad\qquad\qquad Cualquier\; otro\; caso \end{cases} \qquad (32)$$

Al aplicar esta técnica todos los píxeles que queden fuera de los umbrales seleccionados serán suprimidos obteniendo como resultado una imagen binaria en donde cada pixel de color blanco representa un pixel del color buscado. Para el desarrollo de este proyecto, se utilizan los valores de 38, 30,30 para el umbral inferior y de 75, 255,255 para el umbral superior, puesto que se busca reconocer una pelota de color verde y los valores de tonalidad para dicho color en el espacio HSV son de 38-75. En la Figura 6.3 se muestra la imagen obtenida después de realizar la transformación al espacio de colores HSV (b) y la imagen resultante al aplicarle la segmentación por umbral (c).



(a) (b) (c)

Figura 6.3 Reconocimiento de color (a) imagen original, (b) imagen en el espacio HSV, (c) imagen tras aplicar la segmentación por umbral.

6.2.3 Pre-procesamiento.

6.2.3.1 Filtros morfológicos.

Para obtener una imagen mejor definida y eliminar el posible ruido, se aplican un par de filtros morfológicos. Un filtro morfológico es capaz de modificar la estructura de la imagen al hacer una operación de la imagen misma con una estructura definida llamada estructura de referencia, que no es más que una matriz de coeficientes que contiene ceros y unos (0 y 255 para este proyecto). La estructura de referencia se puede definir con la siguiente ecuación $H(i,j) \in \{0,1\}$.

Los filtros morfológicos que se aplican a la imagen después de realizar la segmentación por umbral son conocidos como erosión y dilatación.

6.2.3.1.1 Filtro erosión.

Es aplicado para eliminar el posible ruido en la imagen que se presenta como puntos esparcidos. Como su nombre lo indica esta operación hace referencia al fenómeno físico de la erosión eliminando los pequeños grupos de píxeles que quedan dispersos en la imagen. Formalmente se define a esta operación como:

$$I \ominus H = \{x', y' | (x' + i, y' + j) \in P_I(i,j), \forall (i,j) \in P_H\} \quad (33)$$

En donde:

I:Es la matriz asociada a la imagen digital.

P_I: Valores que componen a la imagen.

H: Estructura de referencia.

P_H: Valores que componen a la estructura de referencia.

Esta expresión dice que el resultado de erosionar la imagen I a través de la estructura de referencia H estará dado por los posibles valores $(x' + i, y' + j)$ que pertenezcan a la imagen I para todos los valores (i,j) de la

estructura de referencia H. La Figura 6.4 muestra un ejemplo de esta operación en donde se observa que el pixel $(1,1)$ de la imagen I al sumarse con los elementos $(0,0)$ $(0,1)$ de la estructura de referencia H, se obtiene como resultado los puntos $(1,1)$ y $(1,2)$, los cuales pertenecen a la imagen I, por lo tanto el resultado de la erosión será 1 en la posición $(1,1)$.

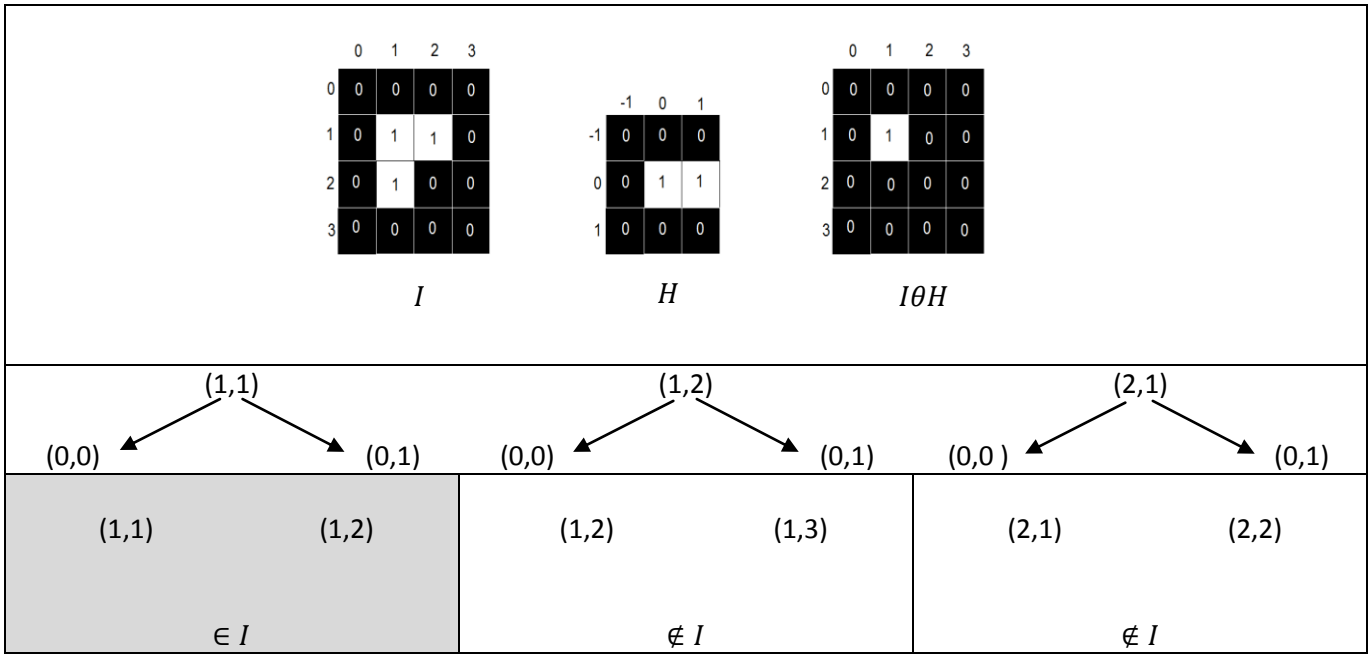


Figura 6.4 Ejemplo filtro erosión.

Para implementar este filtro se hace una llamada al método de la librería OpenCv *erode(scr,dst,element,size)* en donde el parámetro *scr* es el objeto *Mat* origen, *dst* es el objeto *Mat* de destino, *element* es la estructura de referencia, la librería de OpenCv cuenta con una serie de estructuras definidas que son: *MORPH_RECT* para una estructura en forma de caja rectangular, *MORPH_CROSS* para una estructura en forma de cruz y *MORP_ELLIPSE* para una estructura de referencia en forma de elipse. La estructura de referencia utilizada es *MORP_ELLIPSE* debido a que la forma que se pretende reconocer es una circunferencia y una elipse se acopla mejor a esta forma. Por último el parámetro *size* es un arreglo que hace referencia a las dimensiones de la estructura de referencia.

6.2.3.1.2 Filtro dilatación.

Podemos hacer una analogía de esta operación con el fenómeno físico del mismo nombre. Lo que se busca con este filtro es expandir las agrupaciones de pixeles que forman a la imagen y así adquiera una forma más definida. Este filtro se puede interpretar como el resultado de añadir a la imagen los pixeles que forman parte de la estructura de referencia empleada en esta operación. La definición formal de esta operación está dada por la siguiente expresión:

$$I \oplus H = \{(x' + y') = (x + i, y + j)|, (x', y') \in P_I, (i, j) \in P_H\} \quad (34)$$

En donde:

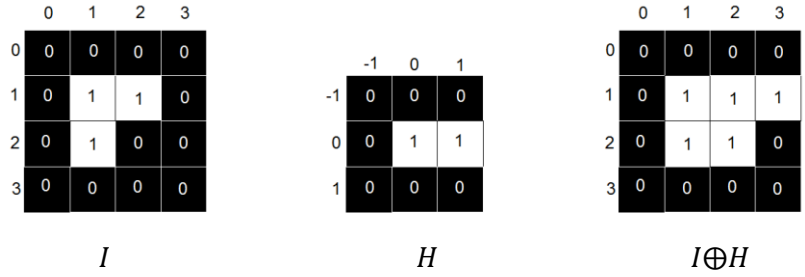
I:Es la matriz asociada a la imagen digital.

P_I: Valores que componen a la imagen.

H: Estructura de referencia.

P_H: Valores que componen a la estructura de referencia.

Esta expresión dice que el resultado de dilatar la imagen I a través de la estructura de referencia H está dado por la adición de todos los puntos que pertenecen a la imagen original P_I con todos los puntos de la estructura de referencia P_H. La Figura 6.5 muestra un ejemplo de esta operación en donde se observa que todos los elementos de la estructura de referencia H son añadidos cada uno de los pixeles que conforman a la imagen I.



I H $I \oplus H$

$$I \oplus H = \{(1,1) + (0,0), (1,1) + (0,1), (1,2) + (0,0), (1,2) + (0,1), (2,1) + (0,0), (2,1) + (0,1)\}$$

Figura 6.5 Ejemplo filtro dilatación.

Para implementar este filtro morfológico en el proyecto, se hace uso del método de la librería de OpenCv *dilate (scr,dst,element,size)* en donde *scr* es el objeto *Mat* de origen, *dst* es el objeto tipo *Mat* de destino , *element* es la estructura de referencia, la librería de OpenCv cuenta con una serie de estructuras definidas las cuales son *MORPH_RECT* para una estructura en forma de caja rectangular, *MORPH_CROSS* para una estructura en forma de cruz y *MORP_ELLIPSE* para una estructura de referencia en forma de elipse. La estructura de referencia utilizada es *MORP_ELLIPSE* debido a que la forma que se pretende reconocer es una circunferencia y una elipse se acopla mejor a esta forma. Por último el parámetro *size* es un arreglo que hace referencia a las dimensiones de la estructura de referencia.

Como se ha mencionado el propósito de aplicar estos filtros a la imagen es el de obtener una nueva imagen más definida así como eliminar el posible ruido que se produjo al realizar la transformación de espacios y la segmentación por umbral , en la Figura 6.6 se muestra el resultado de aplicar estos dos filtros a la imagen binaria.

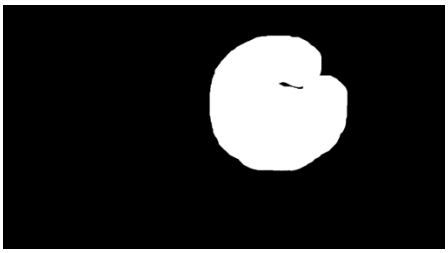


Figura 6.6 Imagen resultante después de aplicar los filtros morfológicos.

6.2.3.2 Algoritmo de Canny.

Con el procesamiento hasta ahora aplicado en la imagen captada por la cámara del dispositivo, se ha logrado obtener una imagen binaria (Figura 6.6) en donde cada pixel diferente de cero (color blanco) representa un pixel del color que se busca reconocer, por lo tanto, hasta el momento, el color se ha reconocido satisfactoriamente. Ahora el siguiente paso a realizar es el reconocimiento de forma.

Antes de poder hacer uso del proceso elegido para reconocer formas en la imagen, es necesario aplicar el algoritmo de Canny. Como se mencionó en el capítulo cuatro, el algoritmo de Canny es usado para la detección de bordes en imágenes, se trata de un método basado en el gradiente y se aplica en direcciones y resoluciones diferentes, buscando minimizar el número de bordes falsos, una mejor localización de bordes en la imagen y entregar una imagen con el borde de un pixel. La razón por la cual se aplica esté algoritmo es que es más fácil y eficiente buscar figuras en los contornos que forman a la imagen, que en la imagen completa. El método de la librería de OpenCv que se utiliza para implementar el algoritmo de Canny es *Canny(scr,dst,lowth,highth)* en donde el parámetro scr es el objeto asociado a la imagen original, dst es el objeto en donde se va a guardar la imagen resultante , *lowth* es el umbral inferior y *highth* es el umbral superior, si el gradiente calculado en el algoritmo de Canny está dentro de estos umbrales el pixel seleccionado será considerado como borde, en caso contrario se le asignará un valor de cero. La Figura 6.7 muestra el resultado de aplicar este filtro a nuestra imagen binaria.

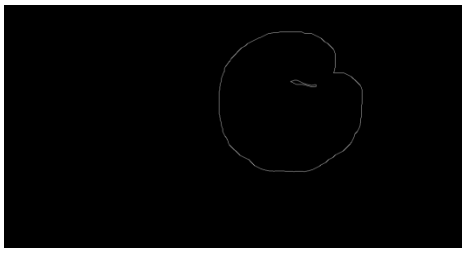


Figura 6.7 Imagen resultante de aplicar el algoritmo de Canny.

6.2.4 Reconocimiento de forma.

Después de haber obtenido los bordes de la imagen, es posible aplicar un método para el reconocimiento de forma, conocido como la transformada de Hough, como se mencionó en el capítulo cuatro, la transformada de Hough es la técnica más utilizada en el campo de visión por computadora para identificar figuras, con ella es

posible encontrar todo tipo de figuras que puedan ser expresadas matemáticamente, tales como líneas, circunferencias y elipses. El método de la librería de OpenCv que realiza está transformada es:

HoughCircles(scr,dst,CV_HOUGH_GRADIENT,dp,min_dis,min_radius,max_radius) en donde los parámetros que necesita este método son: *scr* es el objeto que hace referencia a la imagen origen, *dst* es el objeto en donde se almacenan los resultados de la transformada de Hough, este método arroja tres valores por cada circunferencia encontrada C_x, C_y y ρ que son la coordenada en x del centroide de la circunferencia, la coordenada en y del centroide y el radio de la circunferencia respectivamente, *CV_HOUGH_GRADIENT* esta constante determina el método de detección , actualmente solo este método está disponible en OpenCv, *dp=1* es la razón inversa de resolución, *min_dist* es la distancia mínima entre el centro de las circunferencias buscadas, *min_radius* es el radio mínimo de las circunferencias buscadas y *max_radius* es el radio máximo de las circunferencias buscadas.

Con la información obtenida de la transformada de Hough es posible dibujar otra circunferencia para indicar que el proceso de reconocimiento de color y de forma fue satisfactorio (Figura 6.8).

Se concluye que aplicando esta serie de técnicas, operaciones, algoritmos y transformadas es posible determinar la localización de un objeto con un color y forma determinados, que en este caso se trata de una pelota de color verde. El objetivo es reconocido exitosamente siempre y cuando no esté situado en un ambiente en donde la iluminación es deficiente.

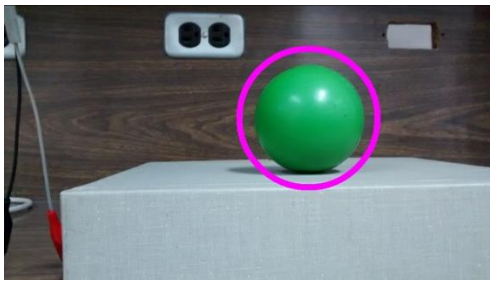


Figura 6.8 Resultados del reconocimiento de color y de forma.

6.3 Desarrollo de la aplicación móvil para Android.

Ya se ha descrito la manera de implementar el reconocimiento de color y de forma, lo siguiente es aplicar estos métodos en una aplicación para un dispositivo móvil con sistema operativo Android. En este apartado se describe el proceso seguido para la creación de la aplicación móvil así como su funcionamiento.

6.3.1 Android Studio.

La aplicación móvil se desarrolló en el entorno de desarrollo integrado IDE por sus siglas en inglés Android Studio en su versión 1.3, debido a que es un entorno completo y sencillo, además de que Android Studio es la herramienta oficial para desarrollar aplicaciones para teléfonos, tabletas, wearables[13] y televisiones que trabajen bajo el sistema operativo Android. Al ser el IDE oficial cuenta con la ventaja de que Google está bajo su soporte corrigiendo problemas y lanzando actualizaciones constantemente.

Android Studio fue creado en colaboración de Google con JetBrains[14] y está construido en base a IntelliJ IDEA el cual es un IDE creado por JetBrains para desarrollar aplicaciones en Java (Figura 6.9), es por esto que Android

[13] Un dispositivo wearable es el que se lleva sobre, debajo o en la ropa, como por ejemplo un reloj.

[14] JetBrains es una empresa dedicada al desarrollo de herramientas que faciliten la tarea de desarrolladores.

Studio es muy similar a este IDE y cuenta con todas sus herramientas y funciones. Android Studio cuenta con Gradle que es una herramienta eficiente encargada de la construcción de cada proyecto, además esta herramienta se encarga de compilar y probar la aplicación así como de empaquetar todas las librerías utilizadas en un proyecto.

Antes de que Android Studio fuera el IDE oficial para desarrollar aplicaciones para dispositivos que cuenten con sistema operativo Android, se necesitaba instalar un plug-in sobre un IDE para poder hacer uso del SDK (kit de desarrollo de software) de Android y así poder desarrollar aplicaciones móviles , el IDE más común para desarrollar aplicaciones para Android era Eclipse.

Android Studio tiene integradas todas las herramientas necesarias para el desarrollo de aplicaciones, tiene una interfaz gráfica amigable y eficiente tanto para el desarrollo de código como para el diseño de la interfaz de usuario de la aplicación, así mismo este IDE está en constante actualización, permitiendo a los desarrolladores hacer uso de las nuevas características y funciones de Android en cuanto se lancen nuevas versiones de este sistema operativo. Android Studio está disponible para Windows, Mac OS X y Linux.

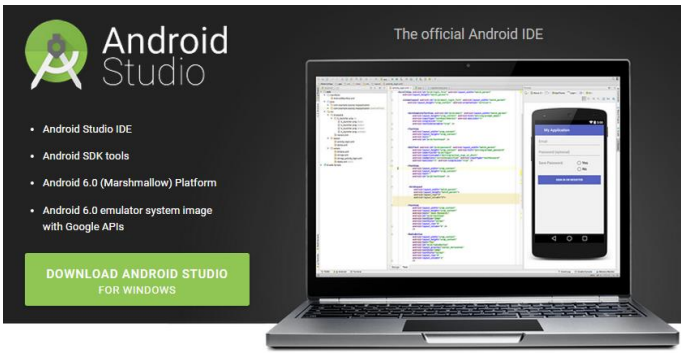


Figura 6.9 Android Studio [20].

6.3.2 Interfaz gráfica de la aplicación móvil.

La interfaz gráfica de la aplicación móvil desarrollada es sencilla debido a que el usuario no necesita interactuar mucho con ella. En la Figura 6.10 es posible observar la pantalla del dispositivo, la cual muestra lo que la cámara está captando, y en un recuadro que está situado en la parte superior derecha se pueden observar las etapas del procesamiento de la imagen. Las etapas que se pueden observar en este recuadro son: la imagen en el espacio de colores HSV, la imagen binaria tras haber realizado la segmentación por umbral y haber aplicado los filtros morfológicos, o la imagen después de haber aplicado el algoritmo de Canny.

Figura 6.10 Interfaz de la aplicación móvil.

Para poder seleccionar la etapa de procesamiento para mostrar en el recuadro superior, el usuario tiene que deslizar el dedo de izquierda a derecha sobre la pantalla, esta acción hará que se despliegue un menú en donde se muestra la opción a elegir (Figura 6.11).

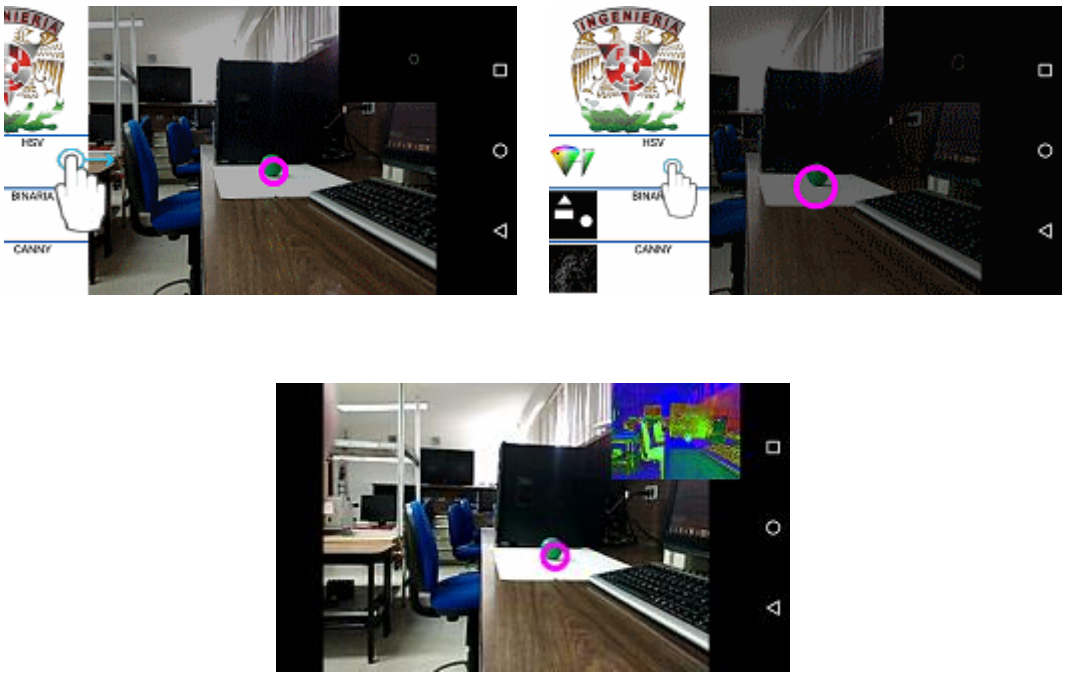


Figura 6.11 Menú de opciones.

6.3.3 Funcionamiento de la aplicación móvil.

El funcionamiento de la aplicación móvil se puede representar en el siguiente diagrama de flujo (Figura 6.12):

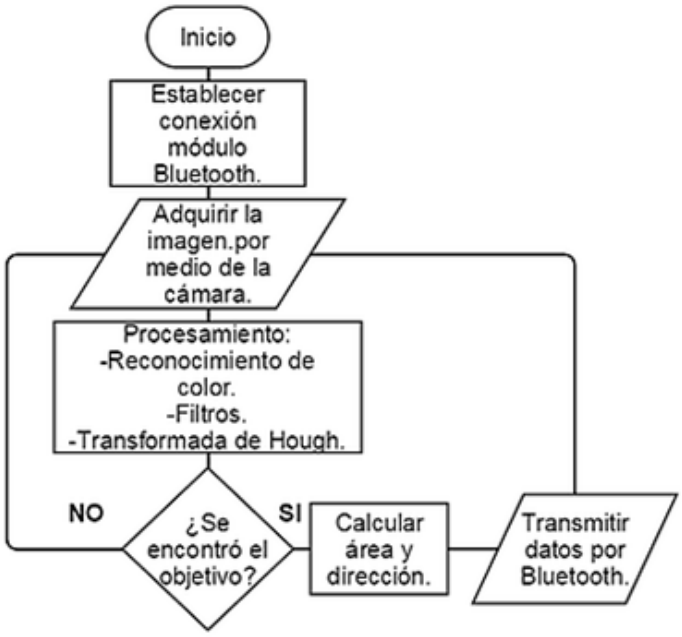


Figura 6.12 Diagrama de flujo del funcionamiento de la app.

De este diagrama (Figura 6.12) podemos resaltar cinco pasos que definen el funcionamiento de la aplicación móvil:

1. Establecer conexión con un módulo Bluetooth (Bt) externo.
2. Adquirir la imagen por medio de la cámara del dispositivo.
3. Procesar la imagen.
4. Si el objetivo fue encontrado, calcular área, dirección e ir al punto 5, de no ser así, regresar al punto 2.
5. Transmitir datos por medio de Bluetooth regresar al punto 2.

A continuación se describen estos pasos y como es que se implementan para hacer funcionar la aplicación móvil.

Es importante señalar que el módulo Bluetooth (Bt) es el puente entre la aplicación móvil y el robot, ya que este módulo está conectado en el puerto serial de un microcontrolador ATMega 328, el cual es el encargado de procesar los datos provenientes de la aplicación y ejecutar las acciones de control para que el robot se dirija hacia el objetivo. Por esta razón se decidió establecer la conexión con este módulo antes de que la aplicación móvil comience a procesar las imágenes. Como se mencionó en el capítulo dos, el módulo Bt utilizado es un módulo HC-05 debido a que es un módulo económico y sencillo de utilizar.

Por otra parte, toda aplicación en Android que requiera hacer uso de hardware o servicios que generen algún costo monetario tiene que implementar una serie de permisos llamados “permisos de usuario”, en este proyecto en donde se hace uso del Bt del dispositivo, es indispensable registrar dos permisos para que la aplicación funcione sin problemas, estos permisos son:

- android.permission.BLUETOOTH

- android.permission.BLUETOOTH_ADMIN

El primer permiso se declara para poder hacer uso de cualquier tarea que requiera conexión Bt, como por ejemplo establecer una conexión con otro dispositivo, aceptar una solicitud de conexión entrante o transmitir datos a través de una conexión. El segundo permiso es indispensable para poder hacer cambios en el Bt del dispositivo como apagarlo o encenderlo. Estos permisos se registran en un fichero dentro del proyecto llamado "Manifest" este fichero es un descriptor de la aplicación.

Ya que se han registrado los permisos correspondientes, el primer paso para establecer la comunicación con el módulo HC-05 es comprobar que el servicio de Bt esté disponible en el dispositivo móvil, esto se hace mediante la clase *BluetoothAdapter* la cual es una clase de Android que representa el adaptador Bt de nuestro dispositivo. Esta clase nos permite realizar tareas de Bt fundamentales: iniciar la búsqueda de otros dispositivos, consultar una lista de dispositivos vinculados, instanciar un dispositivo utilizando una dirección MAC conocida, y crear un *BluetotthServerSocket* para escuchar la solicitud de conexión con otros dispositivos. Para comprobar la disponibilidad del servicio de Bt se hace uso del método *getDefaultAdapter()* , el cual obtiene un identificador para el adaptador de Bt predeterminado. Si este adaptador es *NULL*, significa que el dispositivo no posee Bt o que el servicio no está disponible.

```
// Adaptador local Bluetooth
BluetoothAdapter AdaptadorBT = null;
// Obtenemos el adaptador de bluetooth
AdaptadorBT = BluetoothAdapter.getDefaultAdapter();
```

Después de verificar que el servicio de Bt está disponible, comprobamos que el Bt del dispositivo móvil esté activado, de no ser así solicitamos al usuario activarlo. Para realizar esta acción se hace uso del método *isEnabled()* , que retornará *true* si el Bt está activado y listo para usarse.

```
if (!AdaptadorBT.isEnabled()) {//Si el BT no está esta encendido,
    //Lanza un intent para encender al Bt
    Intent enableBluetooth = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    startActivityForResult(enableBluetooth, Constantes.REQUEST_ENABLE_BT);
}
```

En caso de que el Bt no esté activado se lanzará un *Intent*[15] que solicitará al usuario activarlo. La respuesta de este *Intent* será filtrada en el método *onActivityResult()* y su resultado puede ser:

- RESULT_OK: El Bt se activó correctamente.
- RESULT_CANCELED: El Bt no se activó correctamente o el usuario respondió "Rechazar" a la solicitud.

Debido a que es indispensable que el Bt se encuentre activado para poder realizar la conexión con el módulo HC_05, este proceso se realiza al arrancar la aplicación, en caso de que el Bt del dispositivo móvil se encuentre desactivado se mostrará un cuadro de dialogo que solicita al usuario activarlo (Figura 6.13), este cuadro de dialogo corresponde al *Intent* antes mencionado.

[15] Un Intent es un objeto en Android que representa la intención de realizar una acción.

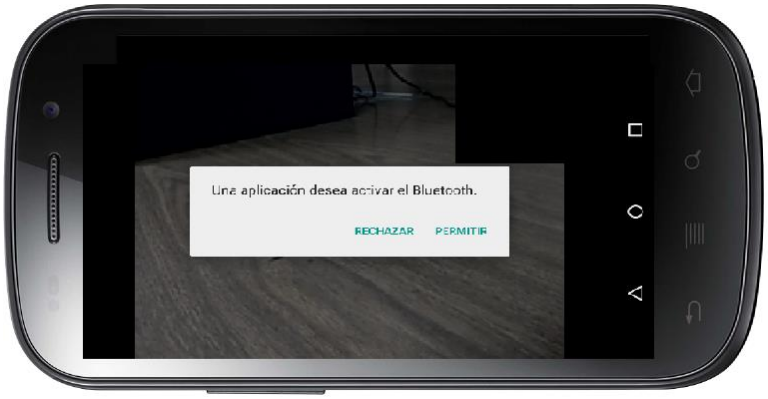


Figura 6.13 Solicitud para activar el Bt.

Ahora que se sabe que el servicio de Bt está disponible en nuestro dispositivo, y que está activado, podemos establecer una conexión con el módulo HC-05. Para realizar esta conexión es necesario:

1. El servidor: es el dispositivo que esté “escuchando” las peticiones de conexiones entrantes.

2. El cliente: realiza la instancia de un dispositivo Bt a partir de la dirección física (MAC) de dicho dispositivo.

3. Conexión: establece y mantiene la conexión entre ambos dispositivos y gestiona el flujo de datos.

De lo anterior se puede concluir que en nuestra conexión dispositivo móvil – módulo HC-05, el módulo jugará el papel de servidor, ya que estará a la espera de que el dispositivo móvil (que se comporta como cliente) solicite conectarse al módulo HC-05 por medio de su dirección MAC. A continuación en las Figuras 6.14, 6.15 y 6.16, se muestra el proceso para establecer la conexión entre el dispositivo móvil y el módulo HC-05.

Figura 6.14 Solicitud de conexión por Bt.

El dispositivo móvil que actúa como cliente lanza una solicitud de conexión al módulo HC-05 por medio de su dirección MAC.

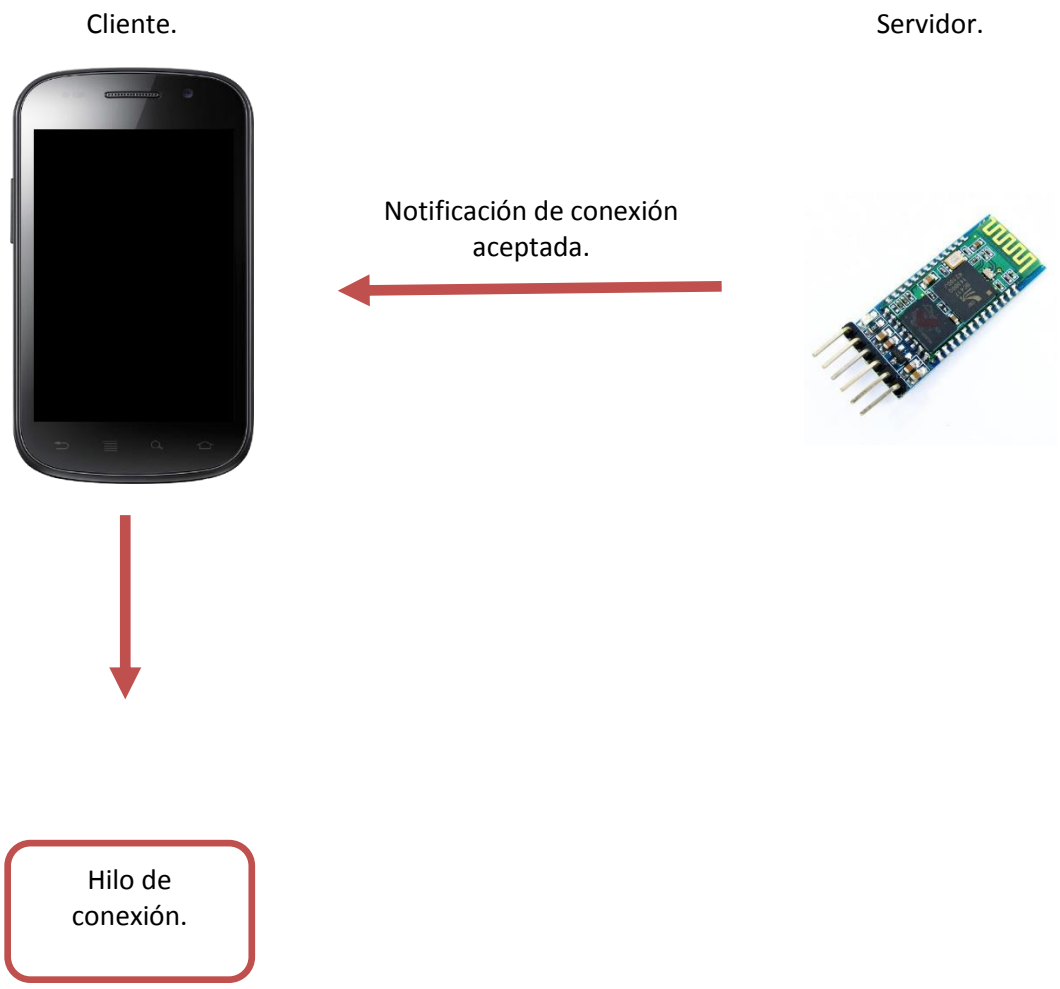


Figura 6.15 Conexión aceptada e hilo de conexión.

El cliente recibe la notificación del servidor, dando como resultado una conexión correcta. El dispositivo móvil abre un hilo de conexión en dónde se gestionaran los datos entrantes y salientes.

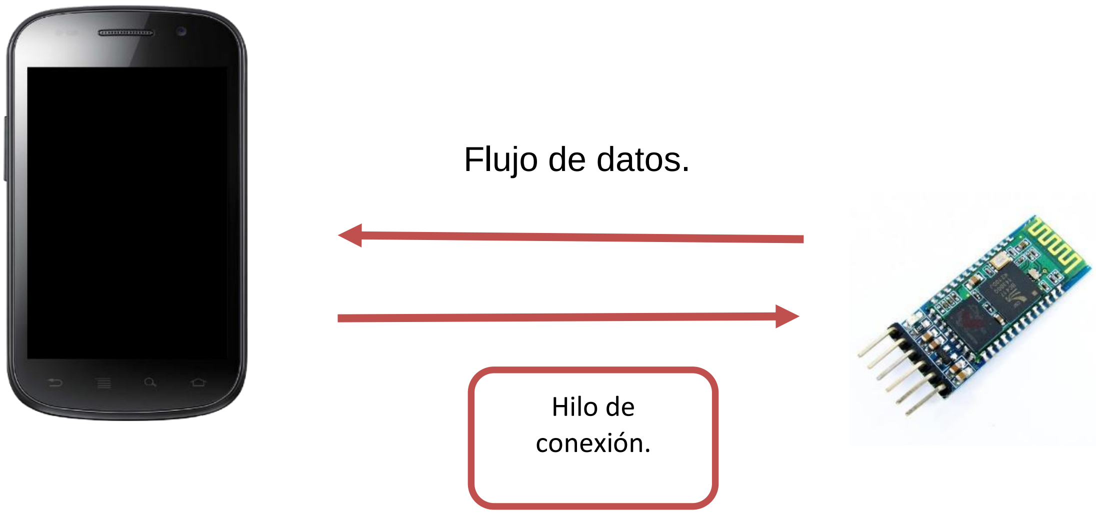


Figura 6.16 Conexión establecida y flujo de datos.

Establecida la conexión, el hilo de conexión se encargará de gestionar el flujo de datos entre ambos dispositivos. Todo este proceso se desarrolló dentro de una clase a la cual se le nombró ConexionBT.java, esta clase es la encargada de realizar la conexión entre el dispositivo móvil y el módulo HC-05. En esta clase se implementaron dos hilos, uno para cada parte de la conexión. Cada hilo se encarga de realizar su tarea:

• Hilo cliente: El hilo cliente se encarga de abrir un socket para solicitar la conexión al módulo HC-05 por medio de la dirección MAC del mismo. La dirección MAC del módulo se obtiene de la actividad[16] principal llamada Main.java, en esta actividad se solicita la conexión por medio de un objeto "*device*" que pertenece a la clase *BluetoothDevice*, este objeto contiene la dirección MAC del módulo la cual es: 20:15:05:20:02:20.

```
    /**
     *Hilo Cliente
     */
    private class HiloCliente extends Thread {
        private final BluetoothSocket mmSocket;
        private final BluetoothDevice mmDevice;

        public HiloCliente(BluetoothDevice device) {
            mmDevice = device;
            BluetoothSocket tmp = null;
            // Obtiene un BluetoothSocket para la conexion con el Dispositivo obtenido
            try {
                tmp = device.createRfcommSocketToServiceRecord(MY_UUID);
            } catch (IOException e) {

            }
            mmSocket = tmp;
        }
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

[16] En Android una actividad es una de las pantallas que pertenecen a una aplicación móvil.

```java
    public void run() {

        setName("HiloConectado");
        //Siempre cancela la busqueda debido a que esta hara lenta la conexion
        AdaptadorBT.cancelDiscovery();
        // Realiza la conexion con el socketBluetooth
        try {
            // Aqui solo recibiremos o una conexion establecida o una excepcion
            mmSocket.connect();
        } catch (IOException e) {
            connectionFailed();
            // Cierra el socket
            try {
                mmSocket.close();
            } catch (IOException e2) {

            }
            // Inicia el servicio a traves de reiniciar el modo de listening
            ConexionBT.this.start();
            return;
        }
        // Resetea el HiloConectado pues ya lo hemos usado
        synchronized (ConexionBT.this) {
            HiloDeConexion = null;
        }
        // Inicia el hiloconectado
        connected(mmSocket, mmDevice);
    }
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

    public void cancel() {
        try {
            mmSocket.close();
        } catch (IOException e) {

        }
    }
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
}//fin de HiloCliente
```

• Hilo conexión: El hilo de conexión se encarga de administrar la conexión y obtener los datos que transmite el módulo HC-05, estos datos son leídos por medio de un objeto *InputStream* y son transmitidos a la actividad principal por medio de un *Handler,* este hilo también se encarga de transmitir datos al módulo HC-05, los datos se envían por medio de un objeto *OutputStream*.

```java
    /**
     * Hilo conexion
     */
    private class HiloConexion extends Thread {
        private final BluetoothSocket BTSocket;
        private final InputStream INPUT_Stream;
        private final OutputStream OUTPUT_Stream;


        public HiloConexion(BluetoothSocket socket) {

            BTSocket = socket;
            InputStream tmpIn = null;
            OutputStream tmpOut = null;
            // Obtencion del BluetoothSocket de entrada y saldida
            try {
                tmpIn = socket.getInputStream();
                tmpOut = socket.getOutputStream();
            } catch (IOException e) {

            }
            INPUT_Stream = tmpIn;
            OUTPUT_Stream = tmpOut;
        }
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

        /**
         * Escribe al Stream de salida conectado
         * @param buffer  Los bytes a escribir
         */
        public void write(byte[] buffer) {
            try {
                OUTPUT_Stream.write(buffer); //Compartir el mensaje enviado con la UI activity
                mHandler.obtainMessage(Constantes.Mensaje_Escrito, -1, -1, buffer).sendToTarget();
            } catch (IOException e) {}
        }//FIN DE WRITE
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
        public void cancel() {
            try {
                BTSocket.close();
            } catch (IOException e) {

            }
        }
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

        public void run() {

            byte[] buffer = new byte[1024];
            int bytes;

            while (true) { //Mantiene escuchando el InputStream mientras este conectado
  try {
            //Lee desde el InputStream
  bytes = INPUT_Stream.read(buffer);
  // byte[] readBufX = (byte[]) buffer;  //Construye un String desde los bytes validos en el buffer
 // String readMessageX = new String(readBufX, 0, bytes);//Se envia el readNessagexxx en lugar del buffer pues ya se PARSE
      mHandler.obtainMessage(Constantes.Mensaje_Leido, bytes, -1, buffer).sendToTarget(); //readMessageX por buffer
 } catch (IOException e) {break; }
            }//_____________-----FIN DE WHILE (TRUE)-----__________________

        }//**************_________FIN DE PUBLIC VOID RUN__________**************
//<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

    }//--------------------*****FIN de la clase HiloConexion*****-------------------
```

Estas son las acciones para establecer la conexión entre el dispositivo móvil y el módulo HC-05. Una vez establecida la conexión se pueden transmitir datos al módulo. Cuando la aplicación móvil detecta que existe conexión con el módulo HC-05, arranca un hilo que se ejecuta cíclicamente en un intervalo de tiempo, este hilo es el encargado de realizar el procesamiento de la imagen, se hablará de esté hilo más adelante. Por ahora se describirá como es que se adquiere la imagen para procesarla y así determinar si la pelota de color verde a la cual la SRM tendrá que dirigirse se encuentra en ella.

De igual manera que para poder hacer uso del Bluetooth del dispositivo se declararon una serie de permisos de usuario, para hacer uso de la cámara se necesita declarar el permiso *android.permission.CAMERA*, este permiso nos permite habilitar y deshabilitar la cámara del dispositivo así como adquirir imagines a través de la misma.

Existen muchos métodos para habilitar la cámara del dispositivo móvil y obtener un cuadro de lo que está captando este sensor, debido a que se está utilizando la librería OpenCv se decidió hacer uso de sus clases y métodos para obtener información de la cámara del dispositivo.

La librería de OpenCv cuenta con la clase *CameraBridgeViewBase,* esta clase es la encargada de la interacción entre la cámara del dispositivo y la librería de OpenCV. A través de un objeto de esta clase se habilita la cámara del dispositivo y también se hace un escalamiento en la imagen capturada, esta operación se realiza por medio del método *setMaxFrameSize(w,h)*, los parámetros que recibe esté método son: *w* y h que representan el ancho y alto que se desea que tenga la imagen captada por la cámara , en este proyecto las dimensiones establecidas para la imagen son 648,364 píxeles de ancho y alto respectivamente. Una vez habilitada la cámara del dispositivo el siguiente paso es capturar la información que está captando, para llevar a cabo esta acción se utiliza la interfaz *CvCameraViewListener2,* esta interfaz se registra por medio del objeto utilizado para habilitar la cámara y ajustar sus parámetros. Los métodos que implementa la interfaz *CvCameraViewListener2* son:

- *public void onCameraViewStarted(int width, int height)* : este método es lanzado cuando la cámara comienza a captar información , los parámetros que se definen en este método son: *width* este parámetro representa el ancho de la imagen captada y *height* que representa el alto de la imagen. Debido a que los objetos que se utilizan en el procesamiento necesitan tener las mismas dimensiones que la imagen captada, es en este método en donde se inicializan dichos objetos haciendo uso de los parámetros *width* y *height*.

- *public void onCameraViewStopped():* se ejecuta cuando la cámara del dispositivo es detenida, ya sea porque el usuario salió de la aplicación o porque se registró un error. En este método se detiene el hilo encargado del procesamiento de la imagen.

- *public Mat onCameraFrame(CameraBridgeViewBase.CvCameraViewFrame inputFrame):* este método devuelve la imagen que está captando la cámara del dispositivo, es aquí en donde la imagen es guardada en una variable global que posteriormente retoma el hilo encargado del procesamiento.

Ya que se ha habilitado la cámara y se tiene una imagen en una variable global, lo siguiente es realizar las transformaciones, operaciones y algoritmos para identificar la pelota buscada. Como se mencionó anteriormente el procesamiento de la imagen se lleva a cabo en un hilo secundario, este hilo es un objeto de la clase *TimerTask*, está clase de java nos permite realizar tareas que se ejecuten repetidamente en base a un temporizador, para este proyecto se define el *TimerTask* de la siguiente manera:

```
timer=new Timer();
inicializaTimerTask();
timer.schedule(timerTask, 1000, 300);//La tarea que realiza el hilo se repite cada 300ms
```

Primero se crea un objeto de la clase *timer*, posteriormente se llama al método *inicializaTimerTask(),* dentro de este método se crea el objeto *TimerTask* y se definen las acciones que se llevan a cabo dentro de este hilo, por último se utiliza al método *schedule(timertask,delay, period)* en donde el primer parámetro es el objeto *timertask* a ser lanzado, el segundo parámetro (*delay)* es el tiempo de espera antes de que se ejecuten las tareas descritas dentro del *timertask* y el tercer parámetro (*period*) es el periodo de repetición de esta tarea , en este caso este tiempo es de 300 ms, esto significa que cada 300 ms se retoma la imagen captada por la cámara y se realizan todas las operaciones para determinar si la pelota de color verde está contenida en esta imagen.

El método *inicializaTimerTask* define las tareas para lograr el reconocimiento: conversión al espacio de colores HSV, segmentación por umbral, implementación de los filtros morfológicos erosión y dilatación, algoritmo de Canny y la transformada de Hough.

```
public void inicializaTimerTask(){

    timerTask = new TimerTask() {
        @Override
        public void run() {
            //Se inicia la variable ti
            ti=System.currentTimeMillis();
            //Trasforma a HSV
            Imgproc.cvtColor(mat_original, mat_hsv, Imgproc.COLOR_RGB2HSV);
            //Segementacion para filtrar el color
            Core.inRange(mat_hsv, hsv_min, hsv_max, mat_segmentacion);

            //Elimina el riudo en la imagen
            Imgproc.erode(mat_segmentacion,mat_segmentacion, Imgproc.getStructuringElement(
                    Imgproc.MORPH_ELLIPSE, new Size(5, 5)));


            Imgproc.dilate(mat_segmentacion,mat_segmentacion, Imgproc.getStructuringElement(
                    Imgproc.MORPH_ELLIPSE, new Size(10, 10)));


                //Filtro Canny
                Imgproc.Canny(mat_segmentacion,mat_canny, 500, 100);

                //Seleccionna la vista para mostrar
                switch (Constantes.MUESTRA_VISTA) {
                    case Constantes.VISTA_CANNY:
                        mensajeVistaPrevia(mat_canny);
                        break;
                    case Constantes.VISTA_HSV:
                        mensajeVistaPrevia(mat_hsv);
                        break;
                    case Constantes.VISTA_BINARIA:
                        mensajeVistaPrevia(mat_segmentacion);
                        break;
                }

                tarsformadaDeHough(mat_canny);
            }
        };

    }
```

Una vez que se realiza el procesamiento de la imagen, y se reconoce correctamente el objeto, se obtienen tres datos como salida: las coordenadas (x, y) del centro de la circunferencia de la pelota buscada y el radio de la

misma, en caso de que la pelota no se encuentre dentro de la imagen procesada o que las condiciones no permitan encontrar dicho objeto estos datos serán nulos, en este caso se obtendrá nuevamente una imagen de la cámara del dispositivo y se repetirá el procesamiento.

Ya que se ha encontrado el objeto buscado y con los datos que se obtienen del procesamiento es posible calcular el área de la pelota así como estimar su orientación dentro de la imagen. Estos datos servirán para guiar a la SRM para que pueda dirigirse hacia donde se encuentra la pelota.

El cálculo del área de la pelota es muy sencillo puesto que se conoce el radio y con este dato se hace uso de la ecuación:

$$A = \pi r^2 \tag{35}$$

El área calculada se utiliza para estimar la distancia virtual a la que se encuentra la SRM del objetivo. El área de la pelota guarda una relación inversamente proporcional con la distancia a la que se encuentra el robot de este objeto, es decir, si el área encontrada es grande, esto significa que la SRM se encuentra muy cerca del objetivo y por el contrario, si el área resultó ser pequeña esto quiere decir que la pelota se encuentra a mayor distancia.

El cálculo de la orientación de la pelota es un poco más complejo, puesto que la orientación de un objeto contenido en la imagen se debe de asemejar bastante a un objeto que se encuentre en el mismo espacio que la SRM.

Para explicar el cálculo de la orientación de la pelota dentro de la imagen primero hablaremos de la convención que se estableció para orientar un objeto respecto a la SRM.

Se asignó convenientemente un sistema de coordenadas a la SRM, de tal modo de que el eje X es positivo hacia el frente del robot y el eje Y es positivo hacia el lado derecho del robot (Figura 6.17 (a)), de esta forma si colocamos un objeto exactamente frente al robot, este objeto quedará situado sobre el eje X , si trazamos un vector desde el origen, que es el centro de la base de la SRM hasta el objeto antes mencionado, el ángulo que formará este vector con respecto a la SRM será de 0°, esto significa que para dirigirse hacia este objeto el robot no tendrá que desviarse hacia ninguna dirección , solo tendrá que avanzar (Figura 6.17 (b)).

Si por otro lado colocamos un objeto del lado derecho de la SRM, dicho objeto quedará dentro del primer cuadrante del sistema de coordenadas establecido, si trazamos un vector desde el origen hasta el objeto antes mencionado, el ángulo que tendrá este vector con respecto al robot estará en un intervalo de $1° - 90°$, esto significa que para dirigirse hacia este objeto la SRM tendrá que desviarse hacia la derecha (Figura 6.17 (c)).

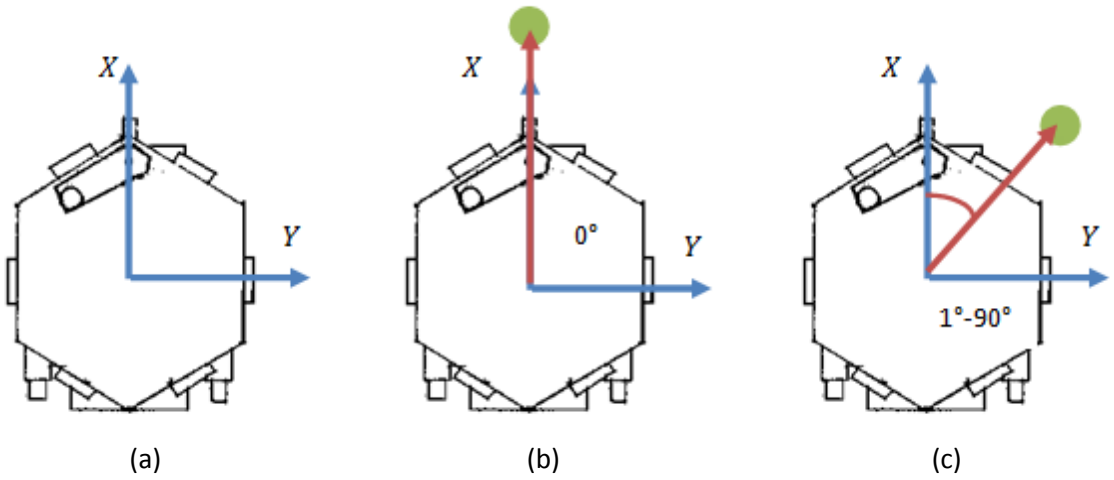


Figura 6.17 (a) sistema de coordenadas establecido en la SRM, (b) objeto frente a la SRM forma un ángulo de $0°$ con respecto al robot, (c) objeto a la derecha del robot forma un ángulo de $1° - 90°$.

Puesto que el dispositivo móvil es el encargado de reconocer el objeto hacia el cual se tiene que dirigir la SRM, se decidió establecer un sistema de coordenadas sobre de la imagen captada, de tal manera que coincida con el sistema de coordenadas establecido en la SRM. Tomando en cuenta que el dispositivo móvil está acoplado en la base del robot y con la cámara apuntando hacia el frente, se estableció el sistema de coordenadas de tal modo que el origen esté situado en la parte central inferior de la imagen, el eje X es positivo hacia arriba y el eje Y es positivo hacia la derecha (Figura 6.18 (a)). De esta manera si el objetivo se encuentra exactamente frente a la cámara y se traza un vector del origen del sistema de coordenadas hasta dicho objeto, el vector coincidirá con el eje X formando un ángulo de $0°$ respecto al sistema de coordenadas (Figura 6.18 (b)), si por otra parte el objetivo se encuentra en la parte derecha de la imagen y trazamos un vector desde el origen hasta el objetivo, este vector formará un ángulo en el intervalo de $1° - 90°$, de esta manera el robot "sabrá" hacia dónde dirigirse (Figura 6.18 (c)).

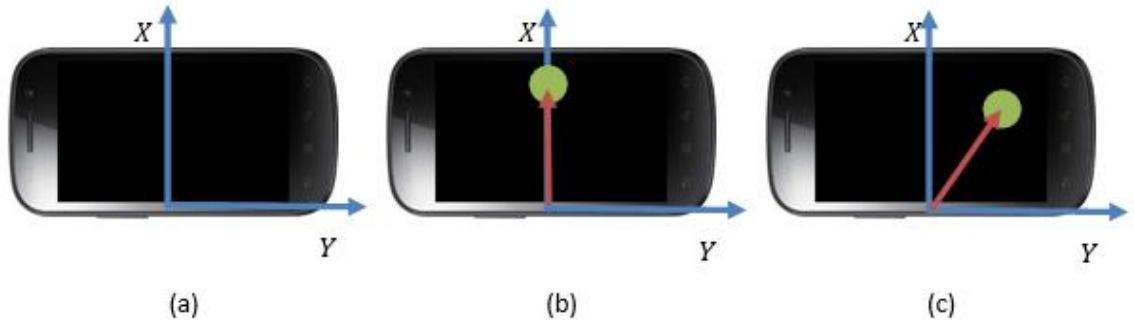


Figura 6.18 (a) sistema de coordenadas establecido en la imagen, (b) objetivo frente a la cámara del dispositivo, (c) objetivo a la derecha del dispositivo.

De esta manera la orientación del objeto reconocido por el dispositivo móvil estará dada por el ángulo que forma el vector de posición de la pelota con respecto al sistema de coordenadas antes descrito.

Si recordamos que uno de los resultados que se obtiene después de haber aplicado la transformada de Hough es el par de coordenadas del centroide del objeto encontrado, se pensará que basta con calcular el ángulo de orientación con el vector de posición que está formado por estas coordenadas, pero existe un inconveniente, ya que estas coordenadas están referidas al sistema de referencia que maneja el dispositivo móvil para todas las imágenes, y no al sistema antes propuesto. Para solucionar esto se le aplica un cambio de sistema de referencia a las coordenadas del centroide obtenido (Figura 6.19).

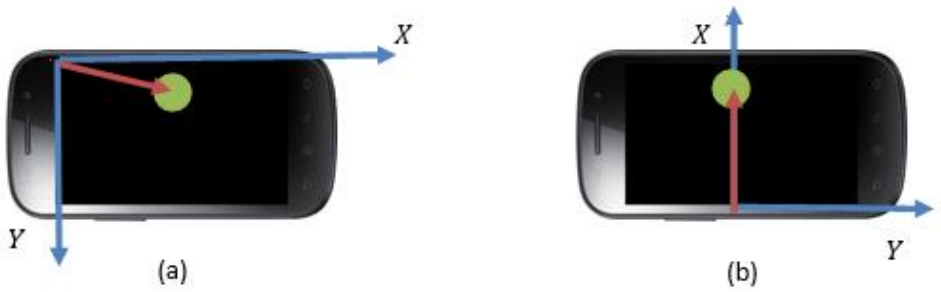


Figura 6.19 (a) sistema de coordenadas establecido por el dispositivo móvil, (b) sistema de coordenadas propuesto.

Para cambiar el punto que representa el centroide del objeto reconocido al sistema de coordenadas propuesto (Figura 6.19 (b)), primero se aplica una rotación de ejes, de tal modo que el eje Y sea positivo hacia la derecha y el eje X sea positivo hacia arriba, si se observa el sistema de coordenadas establecido por el dispositivo móvil (figura 6.19 (a)), se puede llegar a la conclusión de que basta rotar este sistema de coordenadas 90° para que los ejes tengan la dirección que buscamos. Para rotar el sistema de referencia se emplean las siguientes ecuaciones de transformación:

$$y = x' \sin \emptyset + y' \cos \emptyset \tag{36}$$

$$x = x' \cos \emptyset + y' \sin \emptyset \tag{37}$$

Si se sustituye $\emptyset = 90°$ las ecuaciones de transformación quedaran de la siguiente manera:

$$y = x' \tag{38}$$

$$x = y' \tag{39}$$

En donde $(x\ ,\ y)$ son las coordenadas del centroide de la pelota en el sistema de referencia original y (x', y') son las coordenadas en el sistema de referencia rotado $90°$.

Tras haber realizado la rotación de los ejes originales, las coordenadas estarán referidas al sistema que se muestra en la Figura 6.20:

Figura 6.20 Sistema de referencia rotado 90°

Para lograr que el centroide de la pelota quede referido al sistema de coordenadas propuesto, faltaría aplicar una traslación de los ejes ya obtenidos (Figura 6.21). Para aplicar la traslación se hace uso de las siguientes ecuaciones de transformación

$$x' = x - h \tag{40}$$

$$y' = y - k \tag{41}$$

En donde (x', y') es el centroide referido al sistema de referencia trasladado, (x, y) es el centroide referido al sistema de referencia original y (h, k) es el punto en donde se va a trasladar el origen del nuevo sistema de referencia. Recordando que el sistema de referencia propuesto tiene como origen la parte central de la pantalla: $h = \frac{width}{2}$ y $k = high$, en donde $width$ es el ancho de la pantalla y $high$ es el alto de la pantalla del dispositivo móvil.

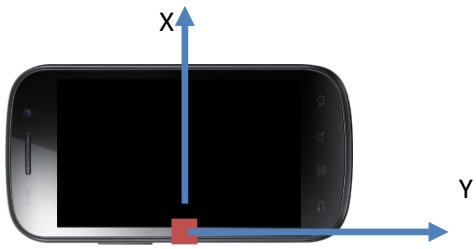


Figura 6.21 Nuevo origen del sistema de referencia con coordenadas $(\frac{width}{2}, high)$.

De tal modo que para que el centroide del objeto quede referido al sistema de coordenadas propuesto se tienen que aplicar las siguientes ecuaciones de transformación sobre las coordenadas obtenidas de la transformada de Hough:

$$x' = y - \frac{width}{2} \tag{42}$$

$$y' = x - high \tag{43}$$

Una vez que se ha cambiado el sistema de referencia solo falta calcular el ángulo que forma el vector de posición de las coordenadas del centroide con respecto al nuevo sistema de referencia, para calcular este ángulo se utiliza el método *atan2(x,y)* este método pertenece a la clase *Math* de java y lo que hace es convertir coordenadas rectangulares (x, y) a coordenadas polares $(r, \emptyset)$ y regresa el valor de $\emptyset$. Estas operaciones son realizadas por la aplicación móvil antes de que los datos sean transmitidos por Bt.

De tal manera que los datos de salida de la aplicación móvil son: área en píxeles y el ángulo $\emptyset$ de la pelota encontrada, estos datos son recibidos por un microcontrolador ATMega 328 que se encargará de procesarlos y realizar las acciones de control para que la SRM se dirija hacia la pelota.

6.4 Implementación de Campos Potenciales

Una vez resuelta la parte del reconocimiento, en donde se indica al robot la dirección y sentido hacia donde debe dirigirse, la siguiente cuestión a resolver es ¿cómo evadir obstáculos?

Retomando la teoría del capítulo 5 acerca de los campos potenciales, en donde se menciona que el espacio se modela como un campo escalar en donde a cada punto se le asigna un potencial, resulta más sencillo de entender si imaginamos un espacio en donde los objetos están definidos por figuras, y esas figuras están dibujadas en una manta, y pensemos que levantamos esas figuras dibujadas en la manta colocando objetos por debajo, formando superficies o regiones más altas; esto representa a los obstáculos, y las partes bajas, es decir a las que no se les colocó objeto alguno por debajo, representan trayectorias libres de colisión o incluso la meta. Ahora, si consideramos el punto más alto como el punto inicial, y el punto más bajo como la meta, pongamos un balín en el punto inicial, entonces es sencillo observar como el balín es atraído al punto de meta, pero al mismo tiempo evitando los obstáculos.

La implementación del algoritmo de campos potenciales en la SRM se puede dividir en 2 procesos: evasión de obstáculos y seguimiento de pelota. El primero es útil cuando existe un obstáculo tentativo que represente un daño potencial para el robot, y el otro proceso es útil para el caso en el que se deba dirigir hacia la pelota pero con la seguridad de que no existe obstáculo alguno.

En la figura 6.22, se observa la SRM conformada por una plataforma hexagonal, en donde también se menciona en el capítulo anterior, es considerada como una partícula (q) de coordenadas $(0,0)$. Se observan los ejes coordenados (x,y) en donde los sentidos positivos van hacia el frente y a la derecha del robot, respectivamente. Se cuenta con 6 sensores, ubicados cada $45°$ con respecto al eje x, es decir uno en cada lado de la figura, de tal manera que si existiera un obstáculo, es posible estimar las coordenadas cartesianas en donde se sitúa éste.

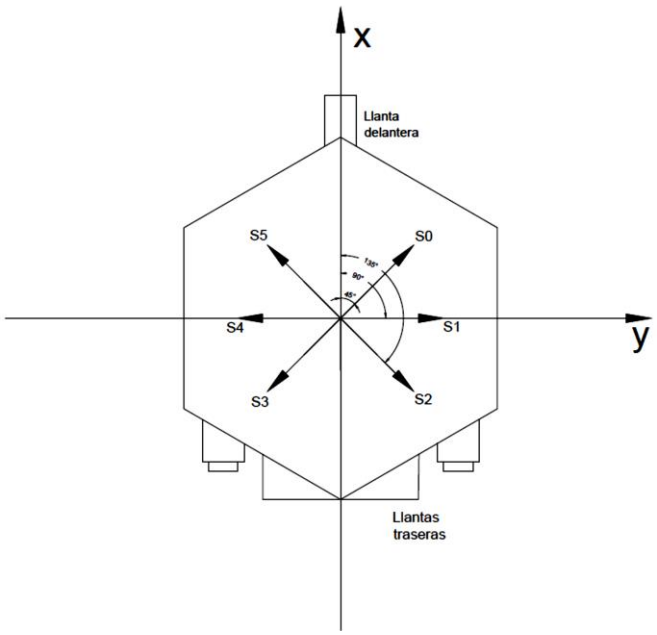
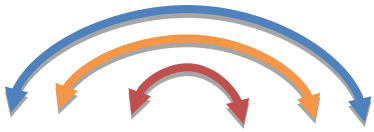


Figura 6.22 Distribución de los sensores en la SRM y ejes de referencia para la evasión de obstáculos. (Vista superior)

El proceso de evasión trabaja de manera conjunta con el proceso de reconocimiento. La evasión comienza con los datos recibidos por el microcontrolador de manera inalámbrica por medio de un dispositivo Bluetooth HC-05 en el puerto serial. Los datos recibidos son área y ángulo, correspondientes a una estimación de la posición de la pelota que funge como meta para la SRM, determinada en el teléfono móvil. El microcontrolador ATMega 328 procesa estos datos para calcular la fuerza atractiva (F_{atr}), cuyas ecuaciones se detallaron en el capítulo anterior. Para calcular q_{atr}, que es el vector de posición del punto al que se quiere llegar, es decir el punto en donde está situada la pelota, se considera que la distancia del robot a la pelota es de $50\ [cm]$; con esto y con el valor del ángulo recibido en el puerto serial se calcula por simple trigonometría las coordenadas de q_{atr}.

La SRM en todo momento está conmutando los 6 sensores para monitorear la presencia de algún objeto y así prevenir cualquier amenaza de colisión; de tal manera que por cada sensor que selecciona, y con ayuda del microcontrolador obtiene la distancia (d) a la que se encuentra un obstáculo, sin embargo, el microcontrolador determina por medio de una constante preestablecida ($d_0\ Distancia\ de\ influencia$) si el objeto detectado representa un riesgo de colisión, o no. Es decir, si la distancia del obstáculo detectado es mayor a la distancia d_0, ese obstáculo no representa un riesgo para la integridad de la SRM, lo cual implica que la fuerza repulsiva (F_{rep}) para ese sensor es igual a cero en sus dos componentes (x, y). Sin embargo, en el caso contrario, o sea que la distancia del obstáculo detectado sea menor a la distancia d_0, se procede a calcular para ese sensor la fuerza repulsiva, cuyo valor puede determinarse por medio de la ecuación (28) del capítulo anterior.

Para obtener el vector de posición del obstáculo a evadir (q_{obs}), lo mejor es tomar en cuenta la Tabla 3 que sintetiza el posicionamiento de los sensores, y en ella también se puede observar la correspondencia que manejan geométricamente, de modo que los sensores frontales (S0 y S5) presentan una dirección de $45\ °$, pero para S5 se trata de un ángulo negativo, o lo que es lo mismo, un ángulo de $315\ °$. Sucede algo similar con el par S1 y S4, y con el par S2 y S3. La tabla también incluye los valores de las funciones seno y coseno de las direcciones de cada uno de los sensores en cuestión. Las flechas indican la correspondencia entre los sensores, por lo que basta con cambiar los signos dependiendo del sensor con el que se esté trabajando.



Sensor	S0	S1	S2	S3	S4	S5
Ángulo [°]	45	90	135	225	270	315
Ángulo [rad]	$\frac{\pi}{4}$	$\frac{\pi}{2}$	$\frac{3\pi}{4}$	$\frac{5\pi}{4}$	$\frac{3\pi}{2}$	$\frac{7\pi}{4}$
Seno	$\frac{1}{\sqrt{2}}$	1	$\frac{1}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}$	-1	$-\frac{1}{\sqrt{2}}$
Coseno	$\frac{1}{\sqrt{2}}$	0	$-\frac{1}{\sqrt{2}}$	$-\frac{1}{\sqrt{2}}$	0	$\frac{1}{\sqrt{2}}$

Tabla 3 Relación de las direcciones de los sensores

Con lo anterior y con la distancia (d) es posible determinar las componentes (x, y) de q_{obs} y consecuentemente la F_{rep} para cada uno de los sensores. Posteriormente se obtiene un promedio de las seis fuerzas repulsivas calculadas, para obtener una sola fuerza ($F_{rep_{Prom}}$). En el caso de que alguna de las componentes de la $F_{rep_{Prom}}$ sea igual a cero, la fuerza resultante (F_{res}) será igual a F_{atr}; esto significa que no hay obstáculo y se puede proceder a ejecutar el otro proceso que es el seguimiento de la pelota, cuya explicación se puede consultar más adelante en este mismo capítulo. Por otro lado, si alguna de las componentes de $F_{rep_{Prom}}$ es diferente de cero, entonces la fuerza resultante, que representa el punto al que debe dirigirse el robot con el fin de evitar la colisión, se calcula como $F_{res} = F_{atr} + F_{rep_{Prom}}$.

Con las componentes (x, y) ya calculadas de F_{res}, el microcontrolador también calcula la dirección y la distancia a la que debe dirigirse la plataforma móvil (anguloFres y distanciaResultante), para que a continuación con la colaboración del magnetómetro y del encoder, proceda a comandar el posicionamiento de la SRM, primero girando hasta encontrar la dirección, y después avanzando la distancia determinada anteriormente (haciaAngRes). Es importante saber que el perímetro de las ruedas traseras de la plataforma móvil (que es en donde está instalado el encoder) es de $42.1\ [cm]$, y como el encoder está construido de tal forma que en una vuelta complete 10 pulsos (pasos), entonces significa que en cada pulso avanza $4.21\ [cm]$, permitiendo así, estimar la distancia recorrida por el robot.

Todo el proceso anterior lo repite indefinidamente, siempre y cuando el teléfono móvil esté enlazado vía Bluetooth con el microcontrolador y que también se haya detectado la pelota. En caso de que no se detecte la pelota, la SRM permanecerá estática.

El proceso de seguimiento de pelota, mencionado en párrafos anteriores, se lleva a cabo del siguiente modo.

El magnetómetro HMC63332 que es utilizado para este proyecto se describió en el capítulo dos, cuyo funcionamiento es sencillo ya que trabaja por medio del protocolo I2C[17]. Este dispositivo maneja una resolución de $1\ [°]$, detecta el campo magnético terrestre, y de manera directa reporta la dirección del norte magnético.

Para esta parte, el microcontrolador pide al magnetómetro la dirección que lleva en ese momento el robot, y comienza a verificar algunas condiciones, para consecuentemente tomar la decisión pertinente:

- Si el área recibida por Bluetooth es menor al área establecida previamente (área equivalente al momento en que la pelota se sitúa aproximadamente a $40\ [cm]$ de la plataforma móvil), el robot avanza, en caso contrario, permanece inmóvil.
- Si el ángulo recibido ("angulo_seguir") por el puerto serial varía sólo en $\pm\ 10\ °$ con respecto al ángulo reportado en ese momento por el magnetómetro, el robot puede dirigirse directamente hacia el frente, no sin antes verificar que la llanta delantera esté alineada y así poder desplazarse directo hacia la pelota.
- Si angulo_seguir es mayor a $0\ °$ pero menor a $180\ °$, el robot avanza a la derecha. En el caso de que angulo_seguir sea mayor a $180\ °$ pero menor a $359\ °$ el robot avanza hacia la izquierda.

Haciendo ilustrativo todo lo anteriormente expuesto, se agregan dos figuras. En la Figura 6.23 se ilustra cada paso que realiza el proceso de evasión de obstáculos, mientras que en la figura 6.24 se enumeran las acciones a seguir para el proceso del seguimiento de pelota. Este proceso está implicado casi al final del proceso de evasión de obstáculos pero es conveniente tratarlos por separado para su mejor comprensión.

[17] protocolo de comunicación serie diseñado por Philips que se utiliza esencialmente entre dispositivos que pertenecen al mismo circuito

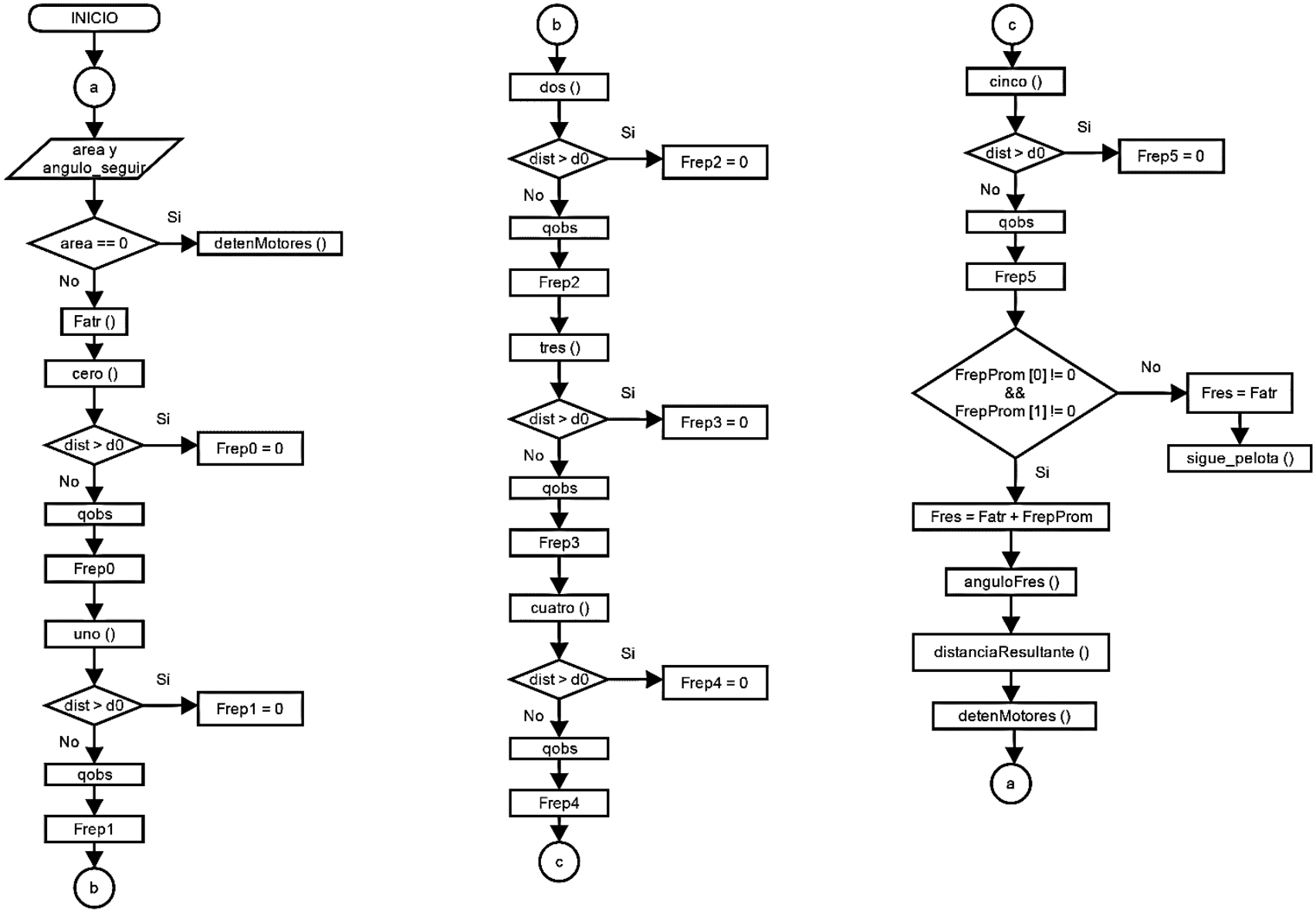


Figura 6.23 Proceso de evasión de obstáculos.

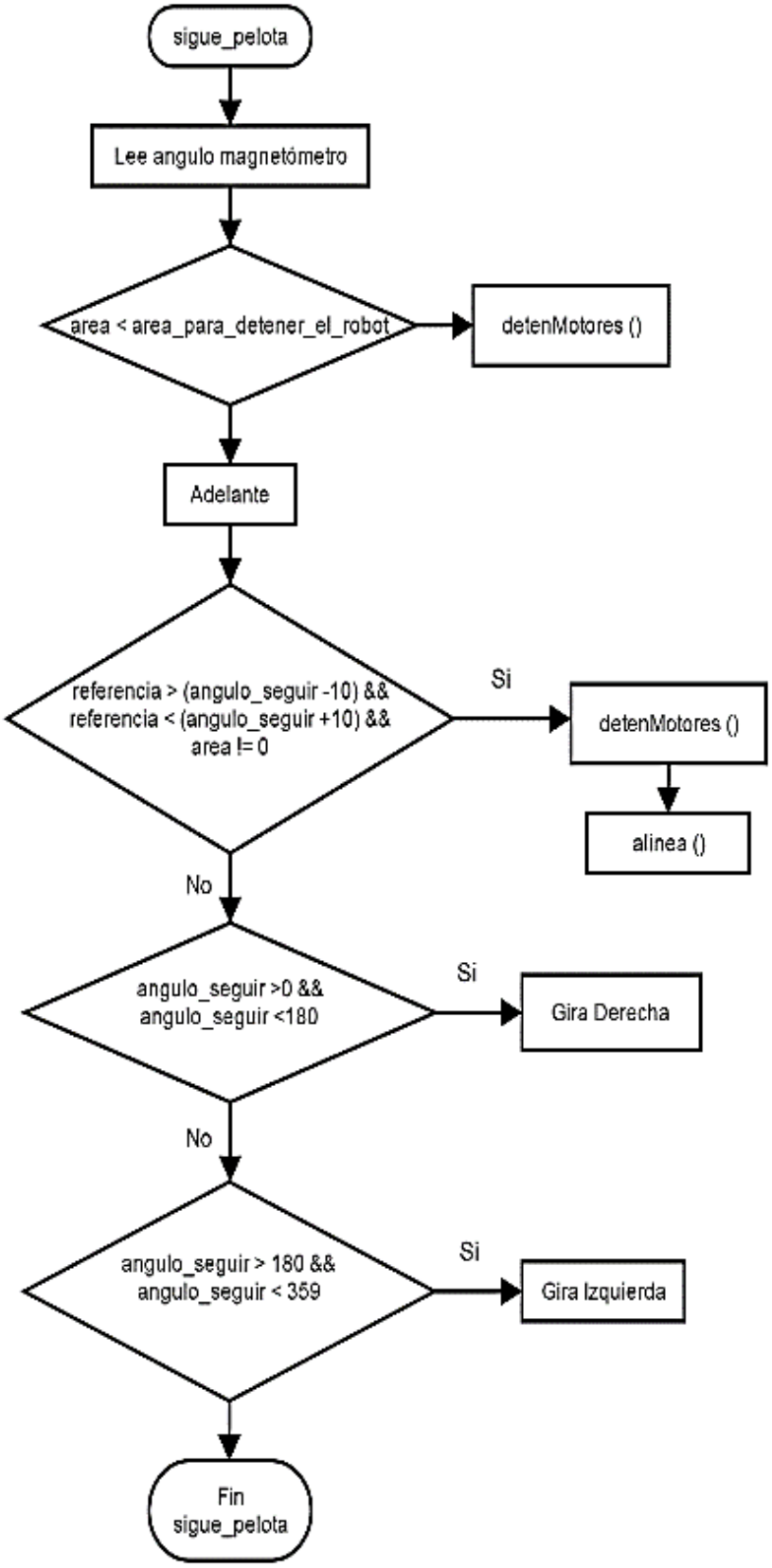


Figura 6.24 Proceso de seguimiento de pelota.

CAPÍTULO 7

PRUEBAS Y RESULTADOS.

Las pruebas realizadas para determinar el comportamiento del sistema desarrollado se dividen en tres etapas. La primera es una prueba del rendimiento de la aplicación móvil y la segunda consiste en determinar el desempeño del trabajo conjunto de la aplicación y el algoritmo de navegación en la SRM, la tercera prueba corresponde al desempeño de la navegación de la SRM en un entorno con obstáculos establecidos.

7.1 Aplicación móvil.

La prueba realizada para la aplicación móvil consiste en colocar el objeto que se programó reconocer (la pelota verde) a determinadas distancias: $40\ [cm]$, $60[cm]$, $80\ [cm]$ y $100\ [cm]$ de tal modo que sea visible para el dispositivo móvil y así obtener los datos del resultado de reconocimiento. Estos datos son: el área del objeto reconocido así como el tiempo en el que se realizó dicho reconocimiento.

Puesto que la iluminación juega un papel importante en cualquier sistema de visión artificial, la prueba antes mencionada se realizó en un entorno con tres condiciones de iluminación diferentes: $100\ \%$ de iluminación, $50\ \%$ de iluminación y luz ambiental. Si tomamos en cuenta que se trabajó dentro de un laboratorio con lámparas de plafón de luz blanca, se considera que la iluminación al $100\ \%$ consiste en tener las lámparas encendidas dentro del espacio de trabajo, la iluminación al $50\ \%$ consiste apagar la mitad de las lámparas e iluminación ambiente es con las lámparas apagadas.

Imagen.	Tiempo [*ms*]	Distancia [*cm*]	Área [*pxs*]	Iluminación.
I_1	211	40	4072	100%
I_2	202	60	1134	100%
I_3	201	80	907	100%
I_4	183	100	706	100%

Tabla 4 Prueba iluminación 100%

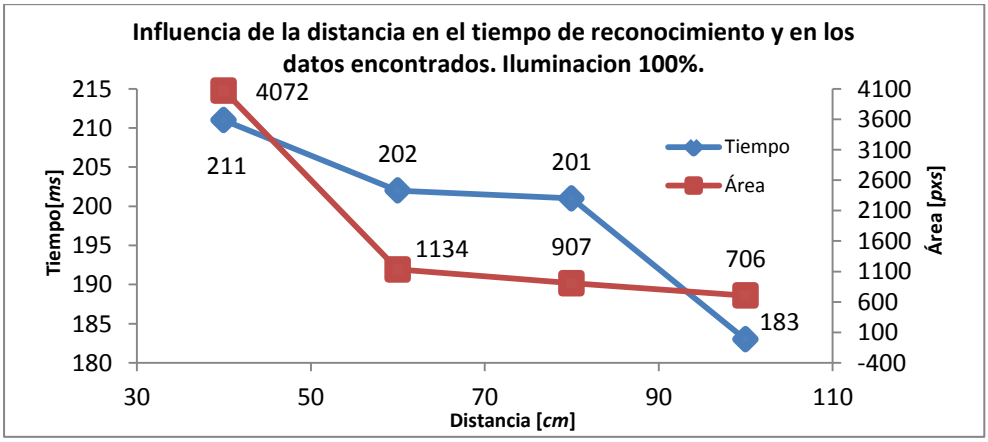


Figura 7.1 Relación distancia con el tiempo de reconocimiento y el área obtenida. Iluminación al $100\ \%$.

La Figura 7.1 corresponde a las gráficas de la relación que guardan tanto el tiempo de reconocimiento como el área del objeto encontrado con la distancia a la cual se coloca este. De la Figura 7.1 se puede observar que el tiempo como el área, tienden a disminuir conforme la distancia aumenta. El proceso de reconocimiento bajo estas condiciones de iluminación es satisfactorio para todos los casos.

Imagen.	Tiempo [*ms*]	Distancia [*cm*]	Área [*pxs*]	Iluminación
I_1	132	40	1809	50%
I_2	130	60	1385	50%
I_3	130	80	615	50%
I_4	118	100	531	50%

Tabla 5 Pruebas iluminación 50 %

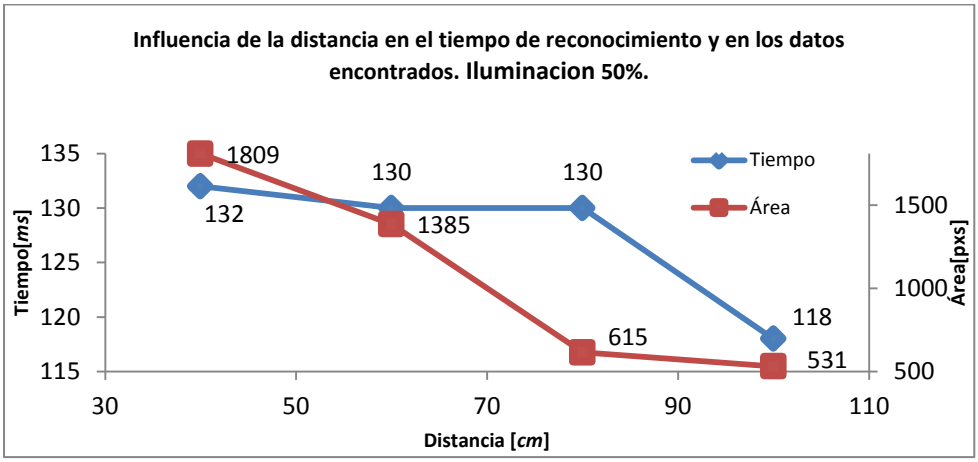


Figura 7.2 Relación distancia con el tiempo de reconocimiento y el área obtenida con una iluminación al 50 %.

La Figura 7.2 corresponde a las gráficas de la relación que guardan tanto el tiempo de reconocimiento como el área del objeto encontrado con la distancia a la cual se coloca este. De la Figura 7.2 se puede observar que el tiempo como el área tienden a disminuir conforme la distancia aumenta, también se observa que el área del objeto reconocido es menor respecto a la prueba realizada con iluminación al 100 %, esto se debe a que el color es una característica que se deforma fácilmente bajo diferentes condiciones de iluminación, es por esto que una cantidad menor de pixeles son considerados como verdes. El proceso de reconocimiento bajo estas condiciones de iluminación es satisfactorio para todos los casos.

Imagen.	Tiempo (ms)	Distancia (cm)	Área (pixeles)	Iluminación.
I_1	131	40	254	0%
I_2	119	60	201	0%
I_3	-	80	-	0%
I_4	-	100	-	0%

Tabla 6 Pruebas iluminación ambiente.

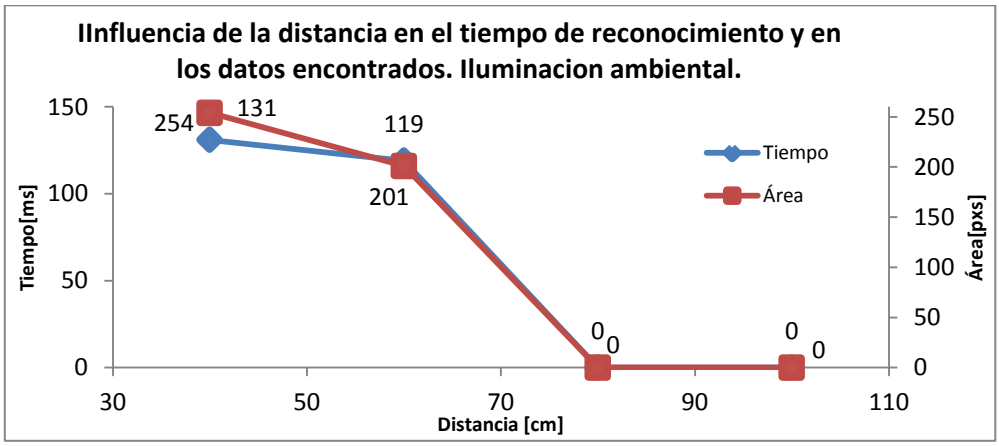


Figura 7.3 Relación distancia con el tiempo de reconocimiento y el área obtenida con una iluminación ambiente.

La Figura 7.3 corresponde a las gráficas de la relación que guardan tanto el tiempo de reconocimiento como el área del objeto encontrado con la distancia a la cual se coloca este. De la Figura 7.3 se puede observar que el tiempo como el área tienden a disminuir conforme la distancia aumenta, también se observa que el área del objeto reconocido es mucho menor respecto a las pruebas realizadas con iluminación al $100\,\%$ como con iluminación al $50\,\%$. El proceso de reconocimiento bajo estas condiciones de iluminación es satisfactorio solo cuando el objeto a reconocer se encuentra a una distancia menor o igual de $60\,[cm]$, en caso contrario las condiciones de iluminación no permitirán el reconocimiento del objeto. Para solventar esta situación realizó una prueba extra con iluminación ambiente haciendo uso del led flash del teléfono, los datos obtenidos de esta prueba se registran en la Tabla 7:

Imagen.	Tiempo (ms)	Distancia (cm)	Área (pixeles)	Iluminación.
I_1	140	40	266	Flash led
I_2	137	60	157	Flash led
I_3	133	80	89	Flash led
I_4	124	100	79	Flash led

Tabla 7 Prueba iluminación ambiente utilizando el flash del dispositivo.

De los datos de la Tabla 7 se observa que, para todas las distancias a las cuales se coloca la pelota verde, el proceso de reconocimiento es satisfactorio, aun así se puede ver que el área reconocida decae drásticamente para las distancias de 80 y 100 $[cm]$, si bien es cierto que la luz proporcionada por el flash del teléfono auxilia a la aplicación en el reconocimiento del objeto, son pocos los pixeles reconocidos como la tonalidad buscada a esto se debe que el área encontrada sea mucho menor en estas distancias que para los casos de cantidad de iluminación antes estudiados.

Como conclusión general de esta serie de pruebas se puede decir que el reconocimiento del objeto hacia el cual se debe de dirigir la SRM es satisfactorio siempre y cuando se cuente con una iluminación adecuada, esto era de esperarse debido a que la iluminación es un factor que influye bastante en los resultados de los sistemas de visión artificial. También se puede concluir que el área del objeto disminuye a medida que la distancia a la cual se coloca aumenta, esto también es algo natural, puesto que si tomamos en cuenta cómo opera la visión natural, veríamos un objeto más grande a medida que nos acercamos a él y más pequeño si nos alejamos, esto pasa de la misma forma con la visión artificial, esta característica es aprovechada para estimar la distancia a la cual se encuentra la SRM de su objetivo. Respecto a el tiempo de procesamiento ocurre algo un poco curioso, puesto que si el objeto a reconocer se encuentra a una distancia de 100 $[cm]$, el tiempo de reconocimiento ronda entre 110 $[ms]$ y 180 $[ms]$ y conforme la distancia disminuye el tiempo de reconocimiento aumenta, obteniendo así el mayor tiempo de reconocimiento, que fue de 211 $[ms]$ para una distancia de 40 $[cm]$ y con iluminación al 100 %, esto se debe a que el tiempo de reconocimiento está ligado a el área del objeto encontrada, puesto que a mayor área, se tienen que procesar más datos para reconocer el objeto. Aun así el tiempo de reconocimiento promedio es de 199.25 $[ms]$ para el caso de iluminación al 100 % y de 127.5 $[ms]$ para el caso de iluminación al 50 %, por lo tanto, este último se considera el mejor escenario para el reconocimiento del objetivo para la SRM en cuanto a rendimiento.

7.2 Trabajo conjunto de la aplicación y el algoritmo de navegación.

Con la segunda prueba se pretende evaluar el desempeño en cuestión de navegación para la SRM. La prueba consiste en colocar la meta para la SRM (pelota verde) en 3 diferentes posiciones, al frente, a la izquierda y a la derecha, a una distancia de 1.5 $[m]$ del robot, de tal modo que la cámara del teléfono sea capaz de captar dicho objeto para que así, la SRM pueda dirigirse hacia la meta y detenerse a una distancia aproximada de 40 $[cm]$ de esta. Es importante señalar que esta prueba se realizó en un espacio libre, es decir, no existen obstáculos para el robot.

Un factor importante en el desempeño del sistema es la iluminación del entorno, pues influye para el reconocimiento que lleva a cabo el dispositivo móvil con Android, es por eso que además de variar las posiciones del objeto al cual el robot se debe de dirigir, también se repiten las pruebas pero para diferentes condiciones de iluminación.

Las pruebas son realizadas en el DISCA en el laboratorio de electrónica y automatización del IIMAS. Para esto se definieron 3 escenarios:

- Iluminación al 100 %
- Iluminación al 50 %
- Luz deficiente (Luz Ambiental)

Para el primer escenario la SRM se situó en un espacio de trabajo con dos lámparas de plafón de luz blanca, ambas lámparas se mantuvieron encendidas, para el segundo escenario sólo se mantuvo encendida una lámpara y para el tercer escenario ambas lámparas se mantuvieron apagadas pero las persianas del laboratorio se mantuvieron abiertas y las pruebas se realizaron en un día soleado. Los resultados son los siguientes:

Objetivo al frente con luz al 100 %		
Distancia [cm]	**Error relativo**	**Tiempo de llegada [min]**
29	-0.275	1:00
66	0.65	0:42
66	0.65	0:40
31	-0.225	1:07
32	-0.2	1:00
Objetivo a la derecha con luz al 100 %		
Distancia [cm]	**Error relativo**	**Tiempo de llegada [min]**
13	-0.675	1:25
27	-0.325	1:23
34	-0.15	1:09
60	0.5	0:52
35	-0.125	1:10
Objetivo a la izquierda con luz al 100 %		
Distancia [cm]	**Error relativo**	**Tiempo de llegada [min]**
24	-0.4	1:21
21	-0.475	1:28
16	-0.6	1:27
47	0.175	1:24
19	-0.525	1:22

Tabla 8 Pruebas con objetivo en diferentes direcciones. Iluminación al 100 %

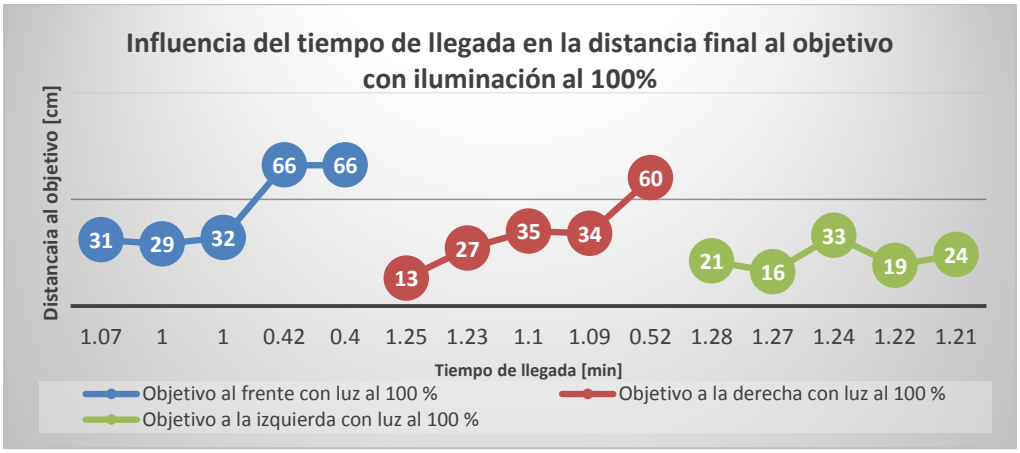


Figura 7.4 Tiempo de llegada vs distancia final. Iluminación al 100 %

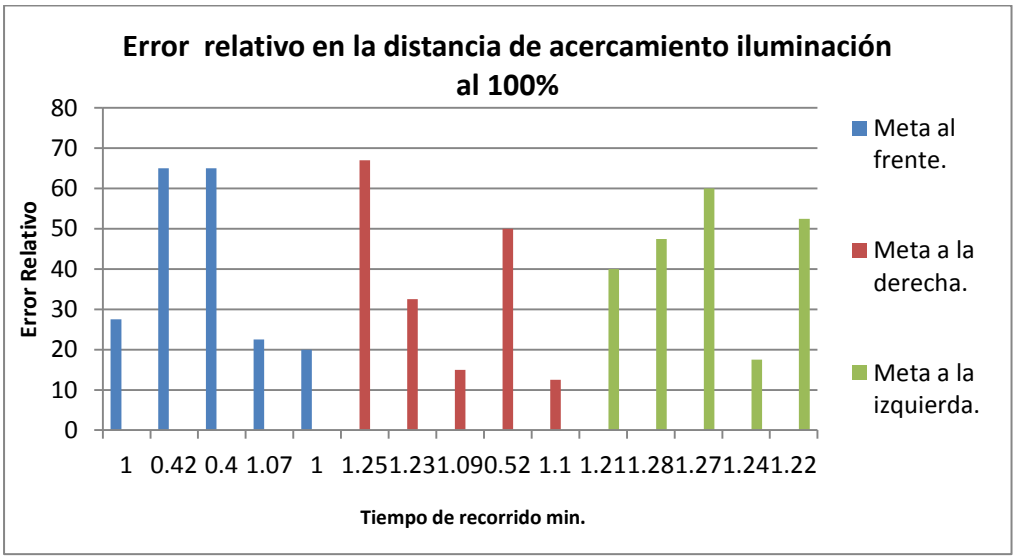


Figura 7.5 Error relativo. Iluminación 100 %

	Objetivo al frente con luz al 50 %	
Distancia [cm]	Error relativo	Tiempo de llegada [min]
22	-0.45	1:00
26	-0.35	1:05
29	-0.275	0:59
55	0.375	0:56
17	-0.575	1:10
	Objetivo a la derecha con luz al 50 %	
Distancia [cm]	Error relativo	Tiempo de llegada [min]
30	-0.25	1:12
40	0	1:08
29	-0.275	1:17
49	0.225	1:15
21	-0.475	1:28
	Objetivo a la izquierda con luz al 50 %	
Distancia [cm]	Error relativo	Tiempo de llegada [min]
24	-0.4	1:16
25	-0.375	1:11
4	-0.9	2:00
16	-0.6	1:49
7	-0.825	2:48

Tabla 9 Pruebas con objetivo en diferentes direcciones. Iluminación al 50 %

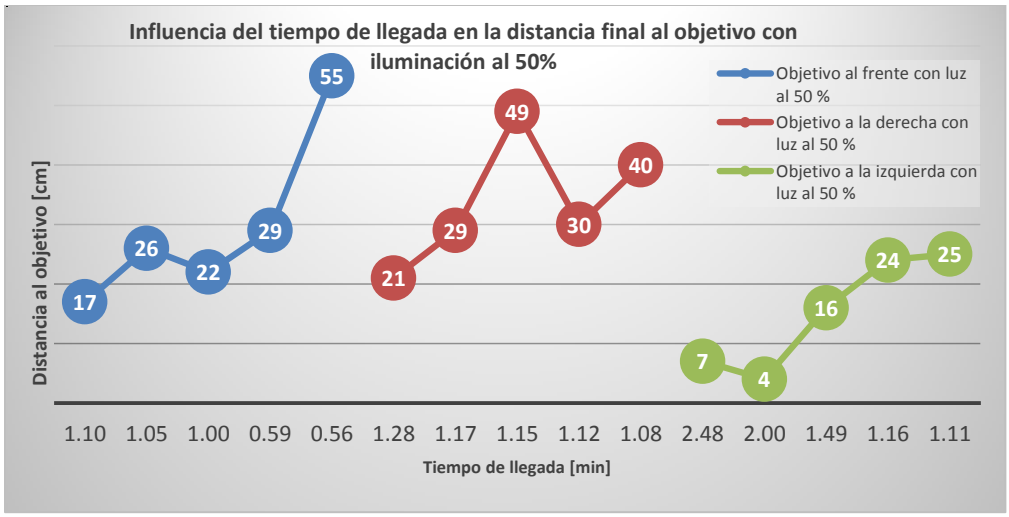


Figura 7.6 Tiempo de llegada vs distancia final. Iluminación al 50 %

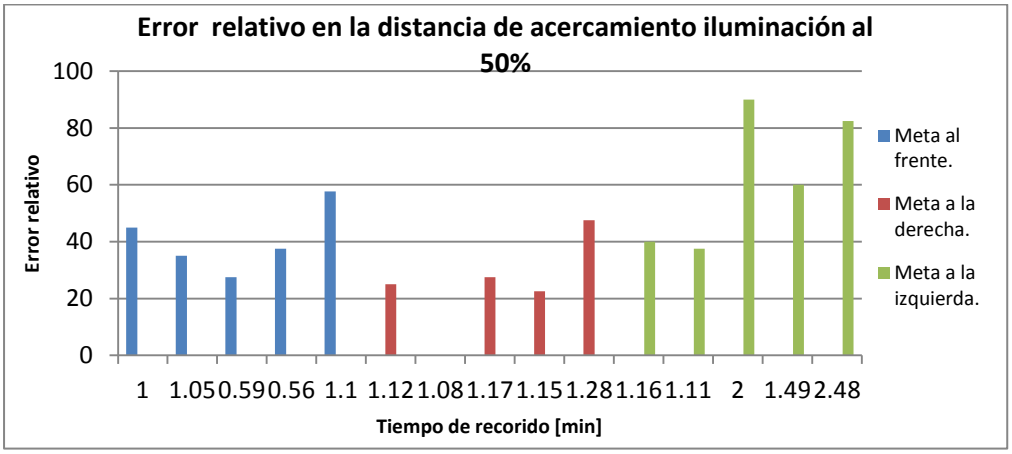


Figura 7.7 Error relativo. Iluminación 50 %

Objetivo al frente con luz deficiente (ambiental)		
Distancia [cm]	**Error relativo**	**Tiempo de llegada [min]**
39	-0.025	1:00
10	-0.75	1:18
15	-0.625	1:10
28	-0.3	1:03
25	-0.375	1:14
Objetivo a la derecha con luz deficiente (ambiental)		
Distancia [cm]	**Error relativo**	**Tiempo de llegada [min]**
9	-0.775	1:31
5	-0.875	1:22
24	-0.4	1:11
5	-0.875	1:24
17	-0.575	1:27
Objetivo a la izquierda con luz deficiente (ambiental)		
Distancia [cm]	**Error relativo**	**Tiempo de llegada [min]**
2	-0.95	1:21
1	-0.975	1:33
24	-0.4	1:14
1	-0.975	1:13
40	0	0:54

Tabla 10 Pruebas con objetivo en diferentes direcciones. Iluminación ambiental

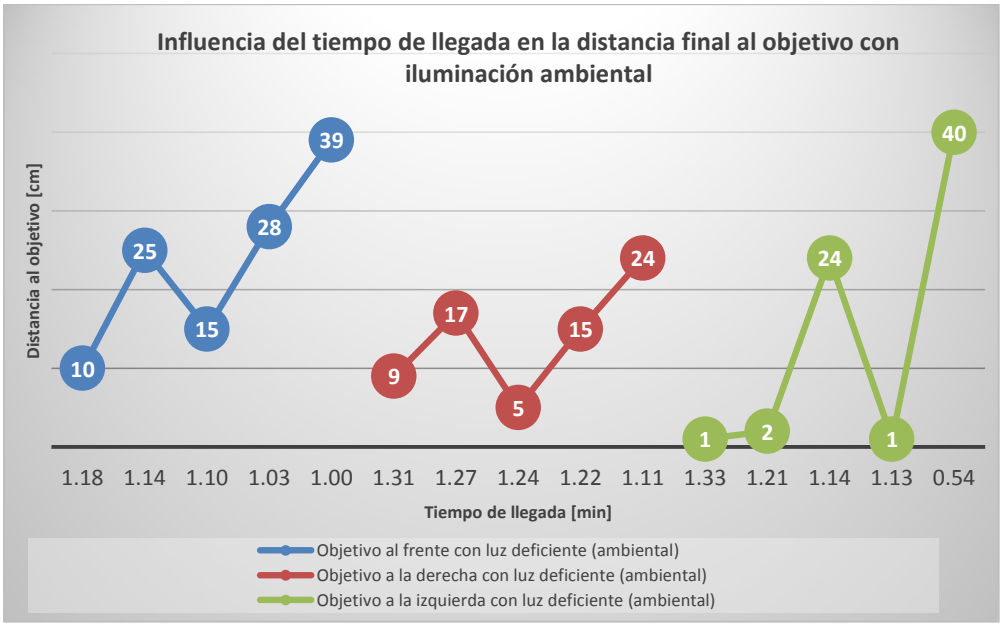


Figura 7.8 Tiempo de llegada vs distancia final. Iluminación ambiental

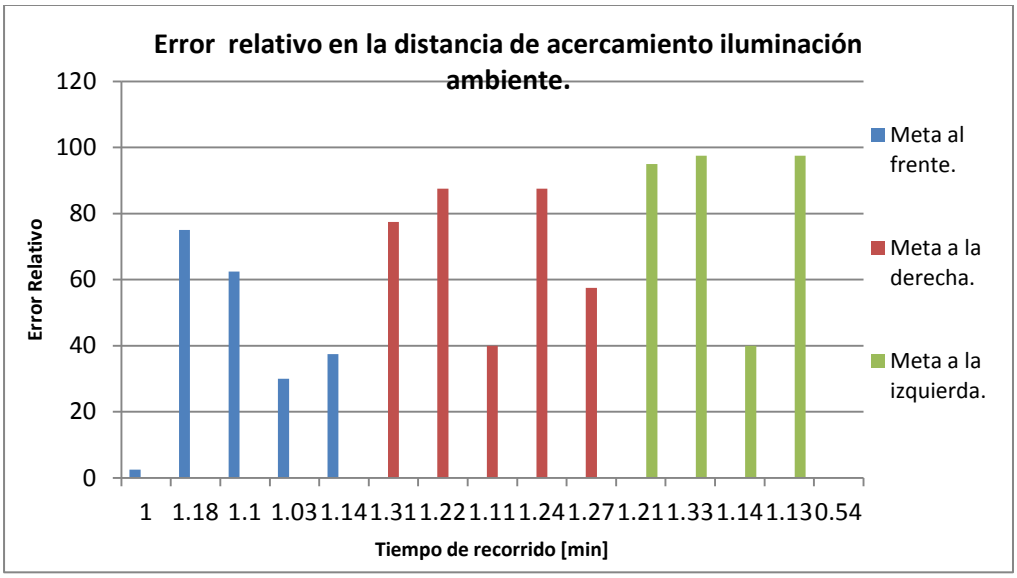


Figura 7.9 Error relativo. Iluminación ambiental

De los datos obtenidos de estas pruebas se puede apreciar que el error en la distancia de acercamiento de la SRM a la meta (pelota verde) se incrementa cuando las condiciones de iluminación en el área de trabajo son deficientes o malas, por ejemplo, en la Tabla 9 se observa que el error es grande y este se incrementa en los casos en donde la meta no está situada al frente del robot. Esto se debe a que la distancia que existe entre la SRM y la meta es estimada por medio del área que se obtuvo del procesamiento de imágenes por medio del dispositivo móvil. En el primer apartado de este capítulo se hicieron pruebas del reconocimiento que realiza la aplicación desarrollada, en dichas pruebas se observa que el área del objeto reconocido es bastante sensible a los cambios de iluminación del entorno, es por esto que la distancia a la que se acerca la SRM a su objetivo no es tan estable y empeora si no se cuenta con la iluminación adecuada.

Por otra parte, de la Tabla 8 se puede apreciar que el error en la distancia de acercamiento es aceptable, estos datos son obtenidos de las pruebas realizadas con iluminación al $100\ \%$, en este escenario el error oscila entre el $15\ \%$ y $20\ \%$ en los mejores casos, definiendo así las condiciones idóneas para que el sistema desarrollado tenga un buen desempeño.

También se puede apreciar que la distancia final del robot a la pelota, posee una relación con el tiempo de llegada, es decir, mientras más tiempo transcurrió más cerca quedó situada la SRM del objetivo. Lo anterior es aplicable aunque se haya cambiado la posición del objetivo, pero conservando las condiciones de iluminación, lo cual se puede observar en las Figuras 7.4, 7.6, y 7.8. Cabe mencionar que en las pruebas en las que se usó la iluminación ambiental tuvieron el mejor desempeño en la cuestión de acercarse más al objetivo, sin embargo son las pruebas que tomaron más tiempo. Por el contrario, las pruebas que llevaron menos tiempo son las que se desempeñaron con la iluminación al $100\ \%$, con la particularidad de presentar las mayores distancias finales al objetivo, lo que representa un resultado inesperado para la prueba; este resultado es ocasionado por la señal de paro, establecida por medio del área captada por la cámara del teléfono móvil, correspondiente a una distancia de $40\ [cm]$ entre el robot y la pelota, es decir, si el área es grande implica que la pelota está muy cerca, por el contrario, si el área es pequeña, la pelota está alejada. Las variaciones de iluminación influyeron en la percepción del sensor de la cámara del dispositivo móvil, distorsionando de algún modo el resultado esperado.

7.3 Navegación en un entorno con obstáculos.

Finalmente se realiza una prueba en donde se coloca la SRM en el espacio de trabajo, en el cual se distribuyen cajas de cartón que representan obstáculos para el robot, también se sitúa la meta del robot (pelota verde) en una posición arbitraria. La prueba consiste en que la SRM llegue a la meta sin colisionar con los obstáculos. Cabe mencionar que esta prueba se realizó bajo las mejores condiciones de iluminación, puesto que el objetivo es probar el correcto funcionamiento del algoritmo de campos potenciales. Los resultados obtenidos se muestran en el siguiente diagrama en donde cada símbolo de "x" corresponde a las posiciones de la SRM en el recorrido de la prueba.

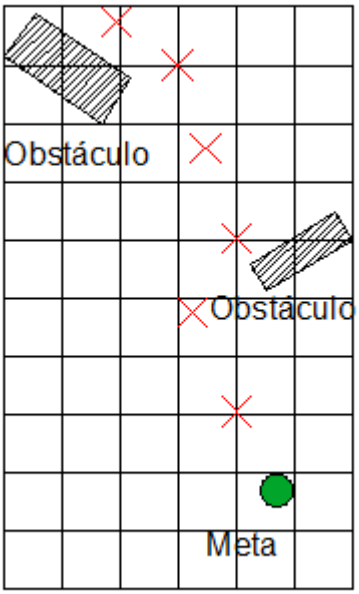


Figura 7.10 Diagrama del recorrido de la SRM evadiendo obstáculos.

Esta prueba es de gran importancia, pues en ella se comprueba la capacidad de evitar daños en la plataforma móvil al navegar, cumpliendo con uno de los principales objetivos del proyecto.

Campos potenciales es un algoritmo reactivo, por lo que en conjunto con la aplicación desarrollada son parte de un proyecto innovador para la ingeniería mexicana, al integrar diversas tecnologías, se conforma un sistema cada vez más complejo que es útil en la resolución de problemas de la sociedad en el país. En este caso la SRM puede ser de utilidad en diversos campos, pues la manera en que hace su trabajo es adaptable para diversas tareas, basta con establecer un objeto de interés para que la plataforma se dirija hacia este. Esto es el primer paso, pero con el desarrollo de nuevas tecnologías y la mejora constante en la electrónica sería imposible no pensar en que en poco tiempo la plataforma pueda ser capaz de cumplir tareas más delicadas.

Índice de figuras

Índice de tablas

Bibliografía

[1] Dorf y R. C., International Encyclopedia of Robotics, New York: John Wiley & Sons, 1988.

[2] Y. S. R. W. Chiaki Aoyama y S. M. N. H. Kikuo Pujimura, «The intelligent ASIMO: System overview and integration.,» *International Conference on Intelligent Robots and Systems.,* pp. 2478-2483, 2002.

[3] M. M. j. B. Suhrid Bhat, S. E. D. H. Tim Hilden y G. H. B. H. Doug Johnston., «Junior: The Stanford Entry in the Urban Challenge.,» *Journal of Field Robotics,* pp. 570-596, 2008.

[4] K. Greene., «MIT Technology Review.,» 15 Junio. 2997. [En línea]. Available: http://www.technologyreview.com/news/408059/stanfords-new-driverless-car/.

[5] N. B. N. K. Parnavi Tamhancar, «Android based object recognition and motion detection to aid visually impaired,» *International Journal of Advances in Computer Science and Technology,* pp. 461-466, 2014.

[6] K. A. W. Q. Karam Melkon y B. F. B. S. Amjed Al-mousa, «A Compact Portable Object Tracking System,» *Information and Communication Systems(ICICS),* 2014.

[7] A. O. Baturone, Robótica Manipuladores y robots móviles, España: Marcombo Alfaomega, 2001.

[8] «El Universal,» 28 Diciembre 2010. [En línea]. Available: http://archivo.eluniversal.com.mx/articulos/62235.html. [Último acceso: 5 Octubre 2015].

[9] J. A. G. M, M. P. Cabrera, H. G. y R. v. O. , «Manejo de una plataforma móvil omnidireccional mediante comunicación Bluetooth a través de un dispositivo móvil,» *SOMI Congreso de Investigación,* pp. 1-9, 2014.

[10] [En línea]. Available: http://oni.escuelas.edu.ar/2013/BUENOS_AIRES/1763/hardware.htm.

[11] L. U. Hernández Muñoz, Percepcion de un robot movil durante su navegacion para evadir obstaculos, Ciudad de México: Facultad de Ingeniería, UNAM, 2002.

[12] Sparkfun, «Sparkfun,» Sparkfun, 2014. [En línea]. Available: https://www.sparkfun.com/products/retired/7915. [Último acceso: 18 Agosto 2015].

[13] Arduino, «Arduino,» ATmega, [En línea]. Available: https://www.arduino.cc/en/Hacking/PinMapping168. [Último acceso: Agosto 2015].

[14] Fastech, «FastTech,» [En línea]. Available: https://www.fasttech.com/product/1129200-jy-mcu-hc-05-bluetooth-wireless-serial-port-module. [Último acceso: Agosto 2015].

[15] E. Marco Pérez, Procesamiento digital de imagenes con MATLAB y Simulink, México: Alfaomega, 2010.

[16] Wikipedia, «Wikipedia,» 4 Septiembre 2005. [En línea]. Available: https://es.wikipedia.org/wiki/Modelo_de_color_HSV. [Último acceso: 14 Octubre 2015].

[17] Scientific & Academic Publishing, «Scientific & Academic Publishing,» 2012. [En línea]. Available:

http://article.sapub.org/10.5923.j.scit.20120205.02.html. [Último acceso: 2015].

[18] K. G. Amara Tasneem, «A nobel aproach face detection using artifiacial neural network,» *International conference on inteligent systems and signal processing ,* pp. 179-183, 2013.

[19] F. Arámbula Cosío y M. Padilla Castañeda, «Autonomous robot navigation using adaptative potential fields,» *Mathematical and Computer Modelling,* nº 40, pp. 1141-1162, 2011.

[20] Google, «Android Developers,» [En línea]. Available: http://developer.android.com/intl/es/sdk/index.html. [Último acceso: 22 Octubre 2015].

[21] A. Barrientos, L. F. Peñin, C. Balaguer y R. Aracil, Fundamentos de robótica, 2 ED., Segunda ed., Madrid: Mc Graw-Hill, 2007, p. 640.

[22] J. Craig, Robótica, Tercera ed., México: Pearson Educación, 2006, p. 408.

[23] J. Iovine, Robots Androids And Animatrons: 12 Incredible Projects You Can Build, Segunda ed., McGraw Hill, 2002.

[24] R. Iñigo Madrigal y E. Vidal Idiarte, Robots industriales manipuladores, Barcelona: Alfaomega, 2004.

[25] S. Kumar Saha, Introducción a la robótica, Noida: McGraw Hill, 2010.

[26] J. A. S. Sánchez., Avances en robótica y visión por computador., España.: Ediciones de la Universidad de Castilla. , 2002.

[27] «IEEE Spectrum,» 8 Noviembre 2011. [En línea]. Available: http://spectrum.ieee.org/automaton/robotics/humanoids/honda-robotics-unveils-next-generation-asimo-robot.

[28] R. Pallás Areny, Sensores y acondicionadores de señal, Barcelona: Alfaomega, 2001.

[29] P. Ponce Cruz, Inteligencia artificial con aplicaciones a la ingeniería, Primera Edición ed., México: Alfaomega Grupo Editor, 2010.

[30] «OpenCV,» [En línea]. Available: http://opencv.org/. [Último acceso: 5 - 11 Julio 2015].

[31] O. Kathib, «Real-time obstacle avoidance for manipulators and mobile robots.,» *The International Journal of Robotics Research,* pp. 90-98.

[32] M. Pérez, J. Alvarez, J. Campo, F. Javier y G. Grillo, Instrumentación electrónica, España: Thomson, 2004.

[33] Autosolar, Autosolar, 2014. [En línea]. Available: https://autosolar.es/baterias/baterias-agm/bateria-agm-12v-7ah-monoblock-ritar_precio. [Último acceso: Agosto 2015].

[34] Expansis, 2014. [En línea]. Available: http://www.expansys.in/motorola-moto-g-xt1032-unlocked-16gb-black-257563/. [Último acceso: Agosto 2015].