Instituto Superior Politécnico
José Antonio Echeverría

Diseño e implementación de una plataforma robótica móvil con aplicaciones basadas en procesamiento digital de imágenes

Autor: **Gustavo Viera López**

Tutor: **Ing. Jorge Silvio Delgado Morales**

Trabajo de Diploma
presentado en opción al título de
Ingeniero en Telecomunicaciones y Electrónica

Facultad de Ingeniería Eléctrica
La Habana, Junio de 2015

A todo aquel que no ha necesitado
un título para demostrar lo que sabe.

Agradecimientos

- A mis padres, por haberme permitido la abstracción de tantas responsabilidades, y haberme apoyado en todo.
- A mis tíos y tías, por ser tan geniales y especiales.
- A mis hermanas, por las conversaciones que me han hecho crecer y por regalarme a los mejores sobrinos del mundo.
- A mis abuelos, primos y todos mis familiares, por tanto que me han enseñado.
- A mis amigos de toda la vida, en especial a Arturo, Amaro, Mesa, el Pichi y Potaje, por estar ahí siempre.
- A Manuel, porque la carrera hubiera sido mucho más difícil sin todo el... estudio que dedicamos.
- A Cesar, por su ayuda y su paciencia.
- A mis profesores, desde la primaria, hasta hoy, sobre todo José Luis, Yaser, Guznara, Jorgito, Céspedes, Dilaila, Cabrera, Raudel, Alexander y Alain.
- A mi tutor, también amigo, salvaje entre los salvajes, por hacerme sentir tan afortunado.
- A Malagón, profesor, tutor y amigo, sin el cual no hubiera conocido el apasionante mundo de la robótica.
- A Marco y Lorenzo, por el tiempo y los recursos que me dedicaron y por haberme enseñado electrónica de verdad.
- A Coto, por cada consejo, cada experiencia trasmitida y tanta ayuda prestada.

- A Danilo, Rolando y Fernando, sin los cuales este proyecto no fuera ni la sombra de lo que es.
- A Rhaúl, Kamil, Antonio y demás graduados del grupo de investigación, por las experiencias y los buenos momentos.
- A mi cuñado Osmany García, por enseñarme a aprender.
- A la gente de netlab, la comunidad técnica más asombrosa que he conocido.
- A quienes han sido mis alumnos o sin serlo, me han concedido el placer de enseñarles algo.

Resumen

El trabajo describe el proceso de diseño de un robot móvil orientado al estudio de técnicas basadas en procesamiento digital de imágenes. Se exponen los detalles de la implementación del hardware de la plataforma y del software de sus controladores. El robot cuenta con comunicación inalámbrica mediante tecnología WiFi, utilizada para la supervisión en tiempo real desde un terminal remoto. Se implementaron varios algoritmos para la detección de obstáculos, seleccionados de forma tal que requieran una sola cámara para su funcionamiento. Se comprueban las ventajas y desventajas de cada procedimiento, y se propone una técnica para la optimización de parámetros en cualquier algoritmo diseñado para este fin.

Abstract

This work describes the process of designing and prototyping a mobile robot oriented to study and develop Digital Image Processing Techniques. All hardware and software, implementations and design are exposed. The platform uses WiFi as the wireless technology for the real-time supervision from a remote terminal. Algorithms that may be used by robots for obstacle detection are explained and analyzed. All of them were selected for requiring only a single camera. Advantages and disadvantages are exposed on the results, and a technique is proposed for the optimization of any algorithm of this kind.

Índice general

Índice de figuras

Índice de tablas

Introducción

La robótica es una rama multidisciplinaria de la tecnología, usualmente definida como la percepción y manipulación de ambientes físicos usando dispositivos mecánicos controlados por computadoras. De ahí que para su estudio es necesario integrar conocimientos de varias ramas de la ciencia y la ingeniería [17].

En la actualidad cada vez es más frecuente encontrar robots realizando tareas que serían imposibles, o extremadamente costosas, de ser acometidas por seres humanos. Con el avance de la tecnología, las metas para los robots se han vuelto muy ambiciosas, aumentando considerablemente las áreas de investigación en este campo. Se han desarrollado sistemas robóticos exitosos como las plataformas de exploración planetaria, brazos robóticos en líneas de ensamblaje, automóviles capaces de desplazarse con autonomía por medios urbanos y manipuladores robóticos que asisten cirugías [22].

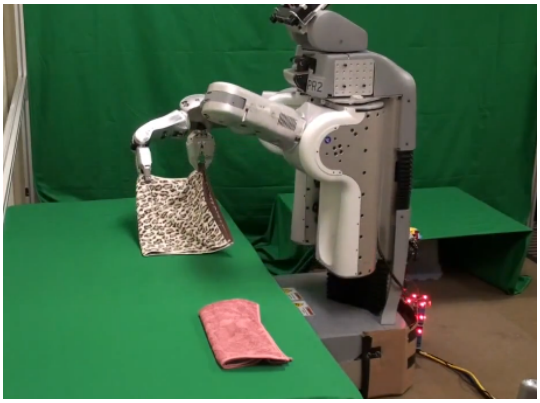

Figura 1: Robot doméstico PR2 doblando ropa

El sector doméstico es uno de los más beneficiados con el empleo de robots móviles. En la figura 1 se puede observar al robot PR2, desarrollado

por la compañía estadounidense Willow Garage. Ya a la venta, este robot es capaz de desarrollar varias tareas domésticas y de oficina, obtener objetos de un refrigerador, doblar ropa, fregar tasas de café y botellas, conectar equipos electrodomésticos a un toma-corriente e incluso celebrar el fin de la jornada laboral con un juego de billar [2].

Un clásico de la robótica móvil es el Rover Curiosity. Desarrollado por la NASA para explorar el planeta Marte con el objetivo de identificar características que pudieran evidenciar la existencia de procesos biológicos, investigar la composición química de la superficie marciana, interpretar procesos que hayan formado o modificado rocas y sólidos en el planeta y determinar el estado y la distribución de dióxido de carbono [1]. En la figura 2 se muestra un modelo tridimensional de Rover Curiosity en el suelo del planeta Marte.

Figura 2: Modelo de Rover Curiosity de la NASA explorando Marte.

Uno de los robots más prometedores de los últimos años es Google *self-driving car*, el automóvil autónomo de la empresa norteamericana, desarrollado para funcionar en ciudades y ambientes dinámicos respetando las leyes del tránsito. Desde mayo de 2012, el estado de Nevada puso en vigor una ley que permite circular automóviles autónomos, a lo que se han sumado paulatinamente otros estados de EE.UU. [21]. En la figura 3 se puede observar uno de los prototipos de Google transitando libremente por la calle.

El desarrollo de robots móviles de propósito específico se ha incrementado considerablemente en los últimos años. El acceso masivo a la información y la disminución gradual de los costos en los materiales y componentes necesarios así lo han condicionado [17].

Figura 3: Google self driving car.

Antecedentes y Motivación

El Departamento de Automática y Computación de nuestro Instituto mantiene una línea de investigación sobre temas de robótica móvil. Como resultado de varias tesis de grado se han construido y perfeccionado diferentes plataformas robóticas, que hasta el momento cuentan con sistemas de localización relativa, control de trayectoria, comunicación y aceleración de algoritmos usando hardware reconfigurable.

Se proyecta desarrollar sistemas de localización absoluta, planificación de trayectorias y generación de mapas. Para la implementación de cualquiera de estas funciones, los algoritmos usados demandan información del entorno. Por tanto no es posible continuar las investigaciones sin una plataforma capaz de detectar obstáculos a su alrededor.

Problema a resolver

La navegación, planificación, generación de mapas y SLAM son tareas de vital importancia para robots móviles. Estas demandan considerables capacidades de cómputo y requieren poder detectar y estimar la posición y dimensiones de los obstáculos del medio. Las plataformas robóticas con que cuenta el grupo de investigación actualmente no son capaces de detectar obstáculos.

Campo de acción

Plataformas robóticas móviles de accionamiento diferencial, sistemas de reconocimiento de objetos y procesamiento digital de imágenes. El objeto de estudio son los sistemas robóticos móviles que operan en ambientes diseñados para la investigación.

Hipótesis

Es posible detectar y localizar obstáculos usando una cámara de vídeo si se cuenta con una plataforma robótica con las prestaciones suficientes para procesar imágenes.

Objetivo General

Desarrollar una plataforma robótica móvil capaz de detectar obstáculos usando técnicas de procesamiento digital de imágenes.

Objetivos específicos y tareas a desarrollar

1. Diseñar una plataforma robótica móvil de altas prestaciones con soporte para el procesamiento de imágenes
   • Revisión bibliográfica sobre las plataformas robóticas móviles
   • Selección de los componentes
   • Diseño de la placa controladora de motores
   • Realización del prototipo

2. Adaptar los sistemas desarrollados en investigaciones precedentes del departamento a la nueva plataforma
   • Adaptación del controlador de velocidad de motores
   • Adecuación del controlador de trayectoria y el sistema de navegación basado en odometría

3. Implementar algoritmos de detección de obstáculos sobre la plataforma desarrollada
   • Estudio del estado del arte de la detección de obstáculos
   • Implementación de algoritmos de detección de obstáculos usando procesamiento digital de imágenes

4. Desarrollar un sistema de pruebas para valorar el desempeño en tiempo real de la detección de obstáculos
   - Comparación de los algoritmos implementados
   - Optimización de parámetros
   - Análisis de los resultados

Estructura del documento

La tesis está organizada en cuatro capítulos seguidos de las conclusiones generales y recomendaciones para futuras investigaciones. El primer capítulo expone el estado de la robótica móvil actual y los sistemas de detección de obstáculos. El segundo capítulo describe la plataforma diseñada, se precisan los detalles mecánicos, de hardware y software. En el capítulo 3 se resume la base teórica de los algoritmos seleccionados para realizar la detección de los obstáculos así como algunos detalles útiles para su implementación y se propone una técnica para la optimización de parámetros de este tipo de algoritmos. El cuarto capítulo expone la descripción y los resultados de varios experimentos diseñados para validar la investigación. Durante todo el documento se hace uso de diferentes términos por sus siglas. Al final del documento se incluye un glosario con los términos empleados y una breve definición de estos.

Capítulo 1

Revisión bibliográfica

Este capítulo sintetiza los conceptos fundamentales de robótica móvil y sistemas de detección de obstáculos empleados en la actualidad. Se resumen de forma breve las tecnologías que podrían ser empleadas en la construcción de un robot.

1.1. Clasificación de Robots

Según su estructura mecánica, los robots pueden clasificarse en plataformas robóticas móviles y en manipuladores robóticos. La principal diferencia es que los manipuladores tienen una base fija que limita sus tareas a un ambiente determinado. A continuación se explican otras clasificaciones, todas resumidas en la figura 1.1.

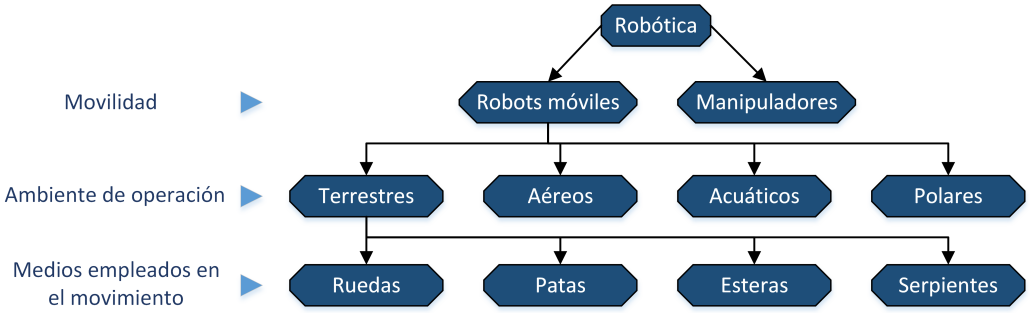

Figura 1.1: Clasificaciones de robots.

Según el ambiente para el cual son diseñadas, las plataformas móviles

se pueden clasificar en terrestres o domésticas, aéreas, acuáticas y robots polares [5]. La figura 1.2 muestra un ejemplo de cada tipo según esta clasificación.

Figura 1.2: Ejemplos de robots móviles.

Adicionalmente, los robots móviles terrestres pueden ser clasificados según los dispositivos que emplean para moverse en cuatro grupos: robots con patas, robots con ruedas, robots con esteras y serpientes [5]. La figura 1.3 muestra un ejemplo de cada tipo según esta clasificación.

En dependencia del terreno para el cual son diseñadas y las tareas que deben realizar, las plataformas robóticas móviles varían sus características, aunque de forma general se pueden describir mediante subsistemas con funciones bien definidas, que serán explicadas en los siguientes epígrafes.

1.1.1. Robots Móviles con Ruedas

De todos los tipos de robots móviles, los robots terrestres con ruedas son sin dudas los más populares. Su estructura mecánica puede ser bastante simple a la par de económica y efectiva en prototipos de pequeña escala, haciéndolos ideales para la investigación. La movilidad de los robots con ruedas está caracterizada por dos factores: el tipo de ruedas que poseen y su disposición sobre una estructura mecánica [17].

Figura 1.3: Ejemplos de robots móviles terrestres.

Clasificación de las ruedas

Según sus características mecánicas las ruedas pueden clasificarse en:

Rueda fija: El eje de la rueda está fijo a la estructura del robot (ver figura 1.4 a). Por lo general está asociada al sistema de tracción del robot [17].

Rueda orientable centrada: Es aquella en la que el movimiento del plano de la rueda con respecto a la estructura es una rotación alrededor de un eje vertical que pasa a través del centro de la rueda (figura 1.4 b). Suele cumplir funciones como rueda de dirección o como rueda de tracción-dirección [17].

Rueda orientable no-centrada (rueda loca): También conocida como rueda castor es una rueda orientable con respecto a la estructura, tal que la rotación del plano de la rueda es alrededor de un eje vertical que no pasa a través del centro de la rueda (figura 1.4 c). Su principal función es la de dar estabilidad a la estructura mecánica del robot como rueda de dirección [17].

Figura 1.4: Tipos de ruedas.(a) Rueda fija. (b) Rueda orientable centrada. (c) Rueda loca.

Configuraciones mecánicas

Existen numerosas posibilidades en cuanto a la cantidad, el tipo y la disposición de las ruedas en una plataforma móvil:

Robot con tracción y dirección en una rueda: Está formado por dos ruedas fijas sobre un mismo eje y una rueda centrada orientable que concentra las funciones de tracción-dirección (figura 1.5). La estructura mecánica y la electrónica de control son sencillas, su tratamiento cinemático resulta de interés en áreas específicas del control de robots móviles. En cuanto a aplicaciones industriales esta configuración es apta para el transporte de cargas pesadas a baja velocidad [5].

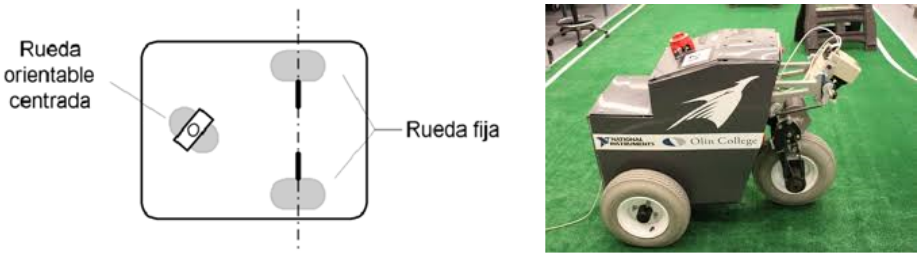

Figura 1.5: Configuración mecánica con dirección y tracción en una rueda.

Configuración Ackerman: Un problema asociado con la configuración anterior es que el centro de gravedad del vehículo se posiciona, en

algunas ocasiones, en los límites de la superficie de equilibrio, definida por las tres ruedas, cuando el vehículo esta en movimiento. Esto produce una pérdida de tracción en el vehículo y es fuente de error a la hora de estimar la posición del robot. Una solución a este problema lo presenta el sistema de dirección Ackerman. Como se observa en la figura 1.6, los ejes de las dos ruedas frontales se interceptan en un punto C que pertenece al eje común de las ruedas traseras. El lugar de los puntos en el plano trazados por cada rueda, alrededor de este punto C, es un conjunto de arcos concéntricos donde todos los vectores velocidad instantánea son tangentes a estos arcos. Esta estructura, además de brindar mayor estabilidad, evita el deslizamiento en la ruedas y por lo tanto reduce los errores de odometría [5].

Si bien su cinemática, su estructura mecánica y su electrónica de control no son tan sencillas, esta configuración presenta un gran interés para los amantes de los robots todo terreno, en donde los principales desarrollos se producen en la recolección de información sensorial y su posterior tratamiento para lograr reconstrucción de entornos o para el control en tiempo real [5].

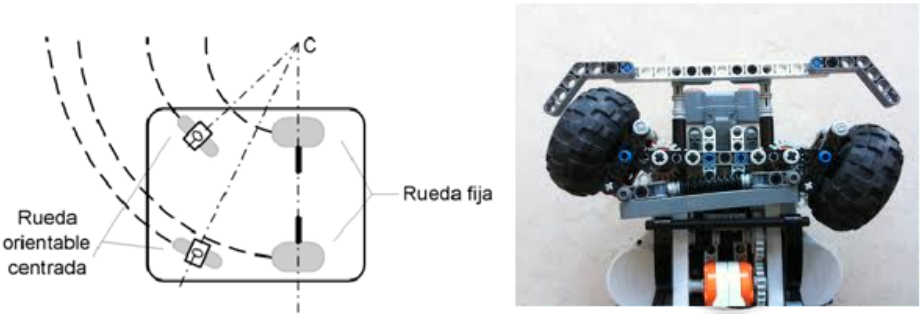

Figura 1.6: Configuración mecánica tipo Ackerman.

Robot omnidireccional: En la figura 1.7 se presenta una versión de robot omnidireccional con ruedas orientables centradas. En este caso el robot puede cambiar la dirección de su movimiento simplemente cambiando la orientación de las ruedas. Este movimiento sincronizado se puede logar por medios mecánicos (figura 1.7 b), empleando sistemas de tracción y dirección por correas o por medios electrónicos, mediante señales de accionamientos simultáneas a partir de la electrónica que comanda cada uno de los motores en las ruedas [17].

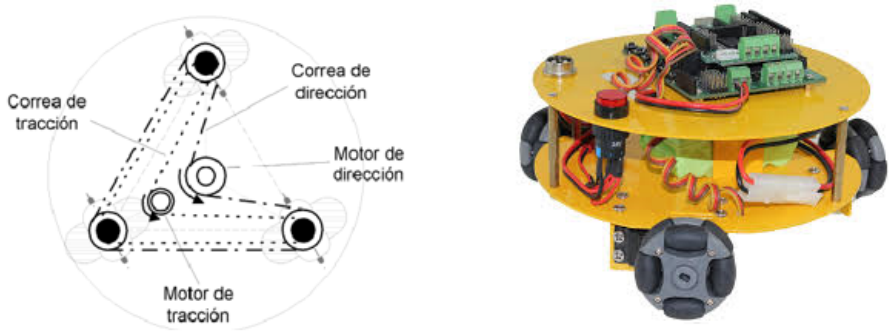

Figura 1.7: Configuración mecánica tipo omnidireccional.

Robot de accionamiento diferencial: Generalmente el elegido por los investigadores a la hora de probar nuevas estrategias de control por tener una cinemática sencilla. Es una estructura que consta de dos ruedas fijas convencionales sobre el mismo eje, controladas de manera independiente y una rueda loca que le confiere estabilidad (figura 1.8). La tracción y la dirección están asociadas a las ruedas fijas. Las ventajas que se derivan de la estructura mecánica y de la electrónica de control hacen de esta configuración la preferida para robots de laboratorio [5].

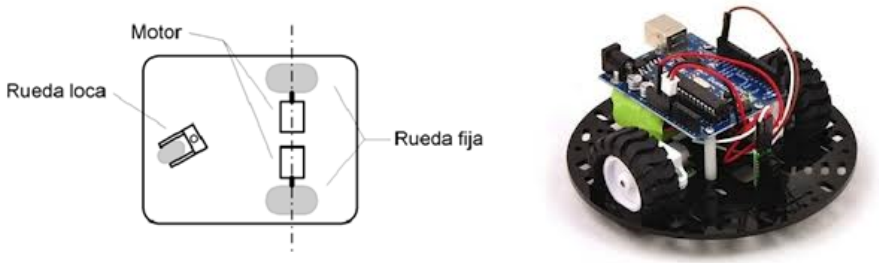

Figura 1.8: Configuración mecánica tipo diferencial.

1.2. Actuadores para Robots Móviles

En un sistema de control automático el actuador es el elemento sobre el cual se ejerce una acción de control, impactando directamente sobre la variable que se está controlando. Son diversas las variantes empleadas como actuadores en sistemas de locomoción para robótica móvil. El uso de motores de corriente continua es uno de los más difundidos.

1.2.1. Motores de Corriente Continua

Los motores de corriente continua pueden ser motores de corriente continua estándares, servomotores y motores de paso.

Motores de corriente continua estándares

Son dispositivos de dos terminales controlados eléctricamente. Estos generan una cantidad considerable de revoluciones por minuto para su tamaño y pueden variar su sentido de rotación si se invierte la polaridad aplicada entre sus terminales. A bajas velocidades, proveen poco torque y en general la precisión en el control de su posición es bastante baja [15].

El empleo de motores de corriente continua estándares para el manejo de plataformas robóticas móviles, se debe a su potencia, la relación velocidad máxima - torque, niveles de ruido y la simplicidad en su control. El accionamiento de estos motores debe permitir controlar la velocidad y el sentido de giro. Lo primero se logra mediante la variación de la tensión media de entrada mientras que el sentido se puede modificar invirtiendo la polaridad de la tensión de alimentación. Para determinar la posición se pueden emplear *encoders* de diversos tipos, algunos de los cuales se analizan en la sección 1.5.1.

Servomotores

Los servomotores son dispositivos diseñados específicamente para aplicaciones que requieren alta precisión en la estimación de la posición del eje del motor. La posición del eje del motor puede ser controlada con exactitud mediante una señal modulada en ancho de pulso, donde la posición del eje está directamente relacionada con el ancho del pulso de la señal. El ángulo de rotación máximo de un servomotor está limitado generalmente entre los 180 o 210 grados. Al contrario de los motores de corriente continua estándares, estos pueden proveer una cantidad significativa de torque a baja velocidad [15].

Motores de paso

Los motores de paso son controlados digitalmente para rotar una cantidad fija de grados (un paso). El número de grados por pasos de un servomotor puede ser tan pequeño como 0.72 grados, o tan grande como 90 grados. A diferencia de los servomotores, los motores de paso pueden girar de forma continua como un motor de corriente directa estándar, pero a una velocidad menor, si cuentan con un circuito de control digital adecuado. Son ideales para aplicaciones de baja velocidad y torque elevado, o cuando se requiere de alta precisión en la estimación de la posición [15].

1.2.2. Modulación de Ancho de Pulso

En el epígrafe anterior se mencionó que las variaciones de velocidad en los motores de corriente continua estándares se lograban mediante la variación de la tensión media de entrada. Para variar la tensión promedio de alimentación de un motor sin requerir de circuitos de potencia analógicos, se emplea una técnica que explota la latencia de los sistemas mecánicos, activando y desactivando la alimentación a una frecuencia elevada [18].

Se genera una señal periódica modulada en ancho de pulso (PWM por sus siglas en inglés) con ciclo útil variable. Esta señal activa o desactiva la alimentación del motor. De esta forma el ciclo útil de la señal permite modificar el nivel de potencia promedio en el tiempo entregado al motor. Un valor 0 de ciclo útil causa que no haya tensión aplicada en la carga (ver figura 1.9 a), mientras que un ciclo útil de 1 permite aplicar al motor la tensión de alimentación íntegramente (figura 1.9 d). Cualquier otro valor de ciclo útil causará que la tensión resultante aplicada a los motores se vea reducida en función del valor del ciclo útil. La figura 1.9 muestra las formas de onda de señales moduladas en ancho de pulso para diferentes valores de ciclo útil.

1.2.3. Puentes en H

En la mayoría de las aplicaciones de robótica es necesario poder variar el sentido de movimiento de los motores. Para lograr este fin, es necesario variar la polaridad de la tensión aplicada a los terminales del motor. Esto se consigue mediante el empleo de un circuito analógico puente en H [18], este término se utiliza por la similitud del esquema del circuito con la letra H, ver figura 1.10 a.

Para lograr una tensión positiva en el motor se cierran los interruptores 1 y 4, y se mantienen abiertos 2 y 3, con esto se logra un giro en un sentido del

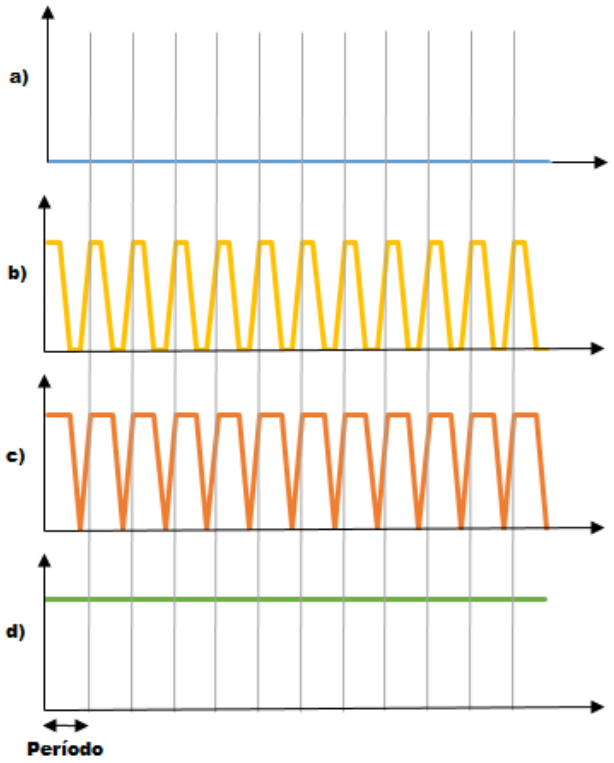

Figura 1.9: Ejemplo de señales moduladas en ancho de pulso.

motor. Si se invierte la lógica anterior el motor varía el sentido de rotación. Para detener el motor se pueden dejar abiertos los 4 interruptores o hacer un cortocircuito en sus terminales, cerrando 1 y 2 con 3 y 4 abiertos o viceversa [18].

En la figura 1.10 b se observa el esquemático a transistores de un puente en H básico. En este caso los transistores se emplean para conmutar entre sus estados de corte y saturación, variando bruscamente su punto de operación, obteniéndose el efecto deseado en el motor según la combinación de estados de los transistores.

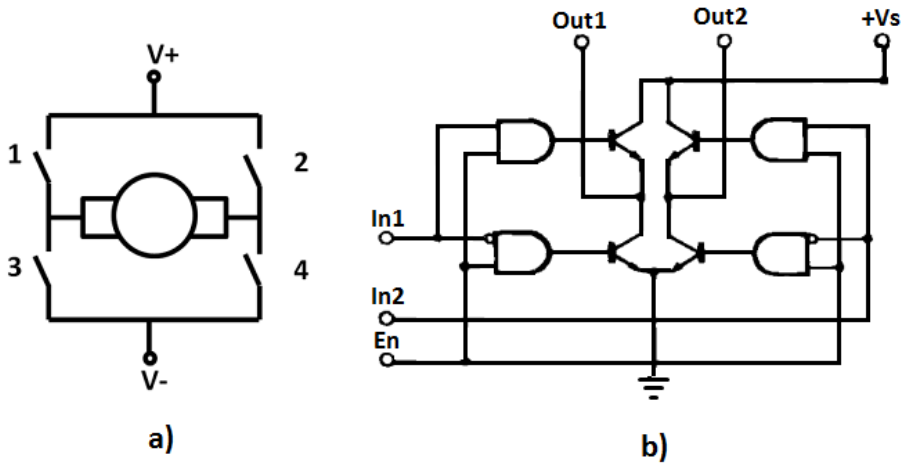

Figura 1.10: Circuito puente en H (a) Esquema lógico (b) Esquema eléctrico.

1.3. Controladores para Robots Móviles

Para la ejecución de las tareas específicas del robot, se emplean circuitos controladores. Un controlador es generalmente un sistema digital encargado de leer los sensores, procesar la información, manejar los actuadores y atender los medios de comunicación. Por las considerables ventajas económicas y simplicidades en el diseño hay una tendencia al empleo de microcontroladores como controlador principal en los robot móviles. De igual forma, sistemas como computadoras personales y hardware reconfigurable son también empleados comúnmente como controladores [24].

1.3.1. Microcontroladores

Desde la década del 70 del siglo pasado, con la aparición de los microprocesadores, se hizo posible procesar cantidades significativas de información usando un único circuito integrado y se disminuyeron los costos y las dimensiones de los sistemas de cómputo. A la par del desarrollo de las computadoras, surgió un uso alternativo de los microprocesadores. Equipos que nada tenían que ver con computadoras incorporaban en su diseño un microprocesador, tales como un refrigerador, un automóvil o las puertas de un garaje. Para estas aplicaciones no era necesario un alto poder de cómputo ni grandes cantidades de memoria. Luego de varios años estos dispositivos

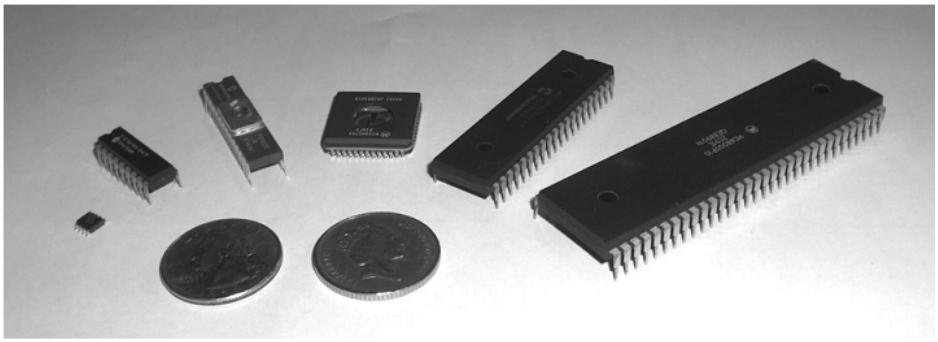

Figura 1.11: Microprocesadores y microcontroladores. De izquierda a derecha: PIC 12F508, PIC16F84A, PIC 16C72, Motorola 68HC05B16, PIC 16F877, Motorola 68000.

de bajas prestaciones diseñados para propósitos específicos fueron ganando su propia identidad y comenzaron a llamarse microcontroladores [24].

Tal como lo hace un microprocesador, un microcontrolador debe ser capaz de realizar cálculos, aunque no necesariamente con grandes cantidades de datos. Por otra parte, los microcontroladores poseen otros requerimientos. Deben contar con interfaces de entrada y salida de datos que permitan el acoplamiento con múltiples dispositivos. Han de ser pequeños, económicos, autocontenidos y deben ser capaces de funcionar en ambientes de industrias o temperaturas extremas [24].

Los microcontroladores se encapsulan en un único circuito integrado, usualmente de plástico o cerámica. Las conexiones con los dispositivos externos se realizan mediante pines de entrada y salida dispuestos en el encapsulado. La cantidad de pines de entrada y salida, y el espaciamiento entre estos son los factores determinantes en el tamaño del circuito integrado [24].

Resulta razonable asumir que para la mayoría de las aplicaciones, el microcontrolador cuenta con toda la memoria necesaria en el interior del circuito integrado. De esta forma, no se requieren la gran cantidad de pines que emplean los microprocesadores para acceder a la memoria externa a través de buses de datos y direcciones. Para transferir el programa a la memoria del microcontrolador se emplean interfaces estándares según el fabricante y el modelo. La figura 1.11 muestra una selección de microprocesadores y microcontroladores donde se observa la diversidad de tamaños y formas de encapsulado [24].

1.3.2. Arduino

Arduino es una plataforma de prototipos electrónicos de código abierto (del inglés *open-source*) basada en hardware y software flexibles y fáciles de usar. Está pensado para artistas, diseñadores, como hobby y para cualquiera interesado en crear objetos o entornos interactivos [10].

Arduino posee dos componentes fundamentales: La placa Arduino, que es un sistema digital basado en un microcontrolador, cuyas características dependen del modelo específico de Arduino; y el IDE de Arduino, la herramienta de software que se ejecuta en la computadora, con la cual se escriben, compilan y envían los programas hacia la placa. Las placas pueden ser fabricadas o compradas pre-ensambladas, puesto que el esquemático es libre y puede ser reproducido [10].

Características que han condicionado el éxito de Arduino

- Cuenta con un entorno de desarrollo de software de código abierto y un lenguaje de programación basado en C.
- Las herramientas usadas para el trabajo con Arduino son multiplataforma, pudiendo ser utilizadas en Windows, MacOS, Linux e incluso Android.
- La programación del microcontrolador interno se realiza mediante una interfaz USB.
- Es de hardware y software libre, puede ser construido desde cero y modificado para propósitos específicos.
- Cuenta con una amplia comunidad de usuarios y una gran diversidad de proyectos de código abierto basados en Arduino.
- Al ser un proyecto desarrollado con fines educacionales está optimizado para obtener resultados de forma rápida.

La figura 1.12 muestra dos placas del modelo Arduino Uno, en sus diferentes versiones. Esta es una placa basada en el microcontrolador ATmega328. Cuenta con 14 pines digitales de propósito general, 6 de los cuales pueden ser usados como salidas de PWM, 6 entradas analógicas, un oscilador de 16 MHz, interfaz USB, conector para alimentación y un botón de reinicio. La programación del microcontrolador se realiza por la interfaz USB. La tabla 1.1 muestra las características técnicas fundamentales de la placa Arduino Uno [3].

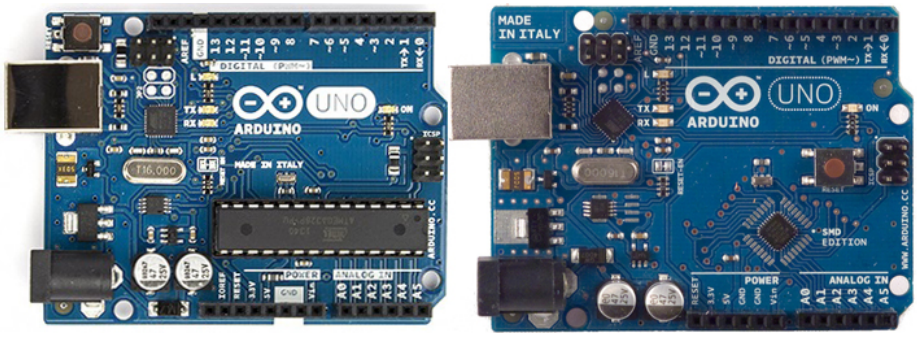

Figura 1.12: Placas Arduino Uno. (a) Versión clásica (b) Versión SMD.

Arduino cuenta además con una amplia gama de circuitos externos diseñados para extender sus funcionalidades. Estas placas son conocidas como *shields* y se caracterizan por tener una distribución de conectores dispuestos de forma tal que la conexión con la placa Arduino pueda realizarse sin soldaduras ni cables. La figura 1.13 muestra como se realiza la conexión entre un *shield* y una placa Arduino [10].

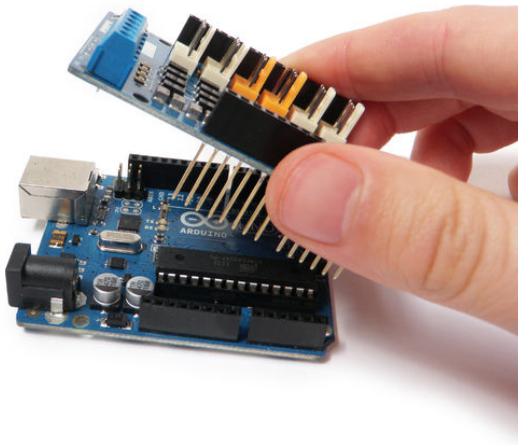

Figura 1.13: Shield de Arduino.

Microcontrolador	ATmega328
Voltaje de operación	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límites)	6-20V
Pines digitales	14 (6 con posibilidad de PWM)
Pines analógicos	6
Voltaje de entrada (límites)	6-20V
Corriente máxima por pines digitales	40 mA
Corriente máxima por el pin de 3.3V	50 mA
Memoria Flash	32 KB (0.5 KB se usan para bootloader)
Memoria RAM	2 KB
Memoria ROM	1 KB
Frecuencia del reloj	16 MHz
Precio	20 USD

Tabla 1.1: Características técnicas de la placa Arduino Uno.

1.3.3. Raspberry Pi

Es un error común pensar que Raspberry Pi es una placa basada en microcontroladores como Arduino. En realidad, puede ser clasificada como una computadora de una placa (SBC por sus siglas en inglés) [14].

Raspberry Pi se ha convertido en uno de los productos más revolucionarios en el mundo de la electrónica de los últimos años. En la figura 1.14 se puede observar una Raspberry Pi, con varios periféricos conectados, siendo usada como una computadora de escritorio y ejecutando el sistema operativo Raspbian.

Hasta la fecha se encuentran disponibles cuatro modelos diferentes de Raspberry Pi: A, B, A+ y B+, cuyas diferencias radican principalmente en componentes particulares de la placa, que no representan ningún problema de compatibilidad para los softwares. El modelo A, se diseñó para ser una variante de bajo costo que pudiera ser usada en aplicaciones donde no se necesite una gran cantidad de interfaces de comunicación. Con sus dimensiones de 8.5 cm por 5.5 cm, puede ejecutar un sistema operativo basado en Linux y una amplia gama de programas. A continuación se listan las principales características de los modelos A y B de la placa y una breve descripción de las mismas [14].

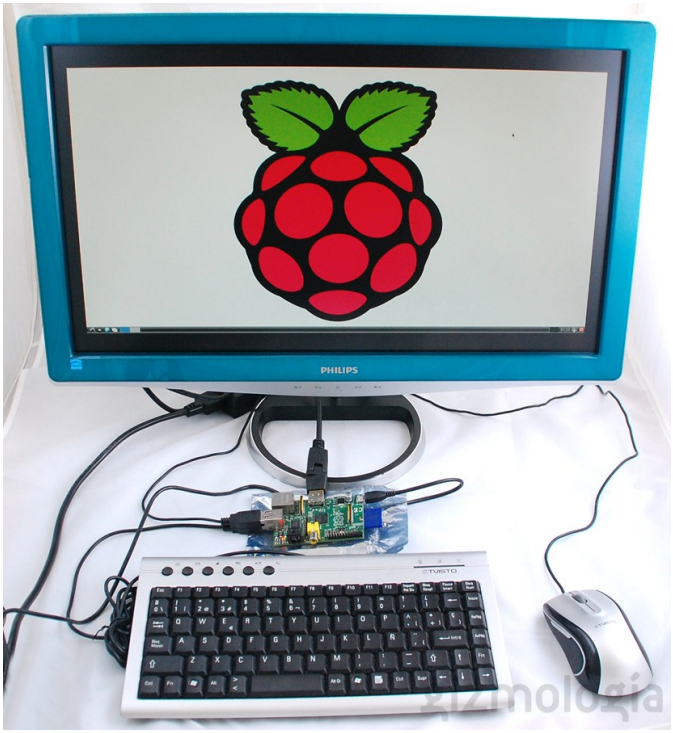

Figura 1.14: Raspberry Pi como computadora personal.

Modelos A y B de Raspberry Pi

A continuación se describen las prestaciones fundamentales de los modelos inciales de Raspberry Pi, especificando las diferencias entre ambos en caso de que existan [14].

Microprocesador: Se emplea el ARM1176JZF-S, es un chip de 32 bits, con un reloj de 700 MHz construido sobre la arquitectura ARM11. El corazón de Raspberry Pi es el mismo microprocesador con que cuentan teléfonos como iPhone 3G y Kindle 2, por lo cual las capacidades de cómputo de esta placa son comparables con las de estos dispositivos. El modelo B cuenta con 512MB de memoria RAM mientras que el modelo A posee 256 MB.

Ranura para tarjetas SD: Para la instalación del sistema operativo y el resto de los programas se emplea una memoria de almacenamiento externa, que cumple la misma función del disco duro en las computadoras

personales.

USB: El modelo B cuenta con dos puertos USB 2.0, a diferencia del modelo A que solo posee uno. Inicialmente las placas Raspberry Pi estaban limitadas en cuanto a la corriente que podía proveer por la interfaz USB, aunque las versiones nuevas cumplen completamente las especificaciones de USB 2.0.

Ethernet: Esto es una característica exclusiva del modelo B, que cuenta con un puerto Ethernet estándar mediante el conector RJ45. Para la conexión a redes de datos también es posible emplear un convertidor USB-Ethernet o un adaptador USB-Wi-Fi, opciones válidas para todos los modelos.

HDMI: Provee una salida de audio y vídeo digital, compatible con 14 resoluciones de video diferentes.

LEDs indicadores de estado: La placa cuenta con 5 LEDs que permiten al usuario conocer información específica de varios procesos sin tener que emplear un monitor.

Salida de audio analógico: Se emplea un conector de audio estándar de 3.5mm, pensado para conectar cargas de alta impedancia como bocinas amplificadas. La calidad del sonido para audífonos o bocinas sin alimentación externa no es la óptima.

Salida de vídeo compuesta: Se emplea el estándar RCA compatible con señales de vídeo NTSC o PAL. Es recomendable emplear la interfaz HDMI debido a la diferencia considerable de resolución que se puede alcanzar.

Entrada de alimentación: Para suministrar la energía se emplea un conector micro USB.

Pines de propósito general (GPIO) y otros pines: Además de las funcionalidades de cualquier computadora personal, con el uso de los GPIO Raspberry Pi puede ser conectada a diversos componentes electrónicos e interactuar con estos de forma similar a como lo hacen los microcontroladores.

Conector DSI (Display Serial Interface): Es un conector para una cinta de 15 pines que puede ser usado para comunicarse con una pantalla LCD o OLED.

Conector CSI (Camera Serial Interface): Permite conectar un módulo PiCamera directamente a la placa de la Raspberry Pi.

Luego del éxito alcanzado por los modelos descritos anteriormente salieron a la venta versiones mejoradas de estos bajo los nombres de A+ y B+. En la figura 1.15 se pueden observar los modelos B y B+ de izquierda a derecha respectivamente.

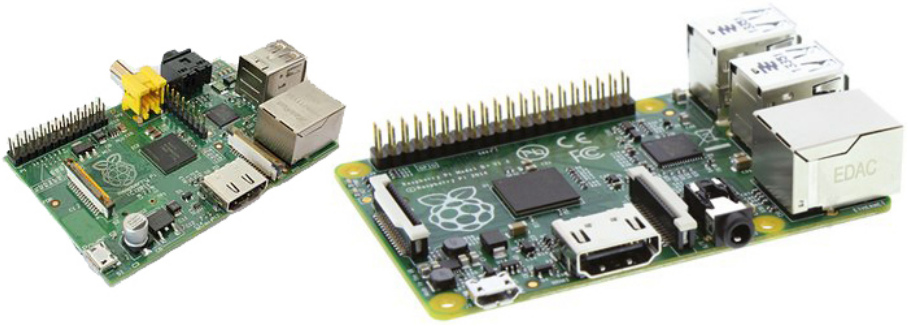

Figura 1.15: Raspberry Pi modelos B y B+.

Diferencias entre los modelos B y B+ de Raspberry Pi

- Se añadieron 2 puertos USB, haciendo un total de 4.
- Se mejoró la capacidad de conexión en caliente de dispositivos de consumos relativamente elevados, como los adaptadores WiFi, que en el modelo clásico causaban el reinicio del sistema.
- Renovación del puerto ethernet.
- Integración del audio y el vídeo analógico en un único conector de 3.5mm.
- Se modificó la posición de varios conectores en la placa para facilitar la conexión de los cables.
- Aumento en la cantidad de pines de propósito general.
- Cambio de ranura SD por microSD.
- Disminución en el consumo de energía.

- Mejoras en la calidad del audio, provistas por el empleo de una fuente de energía dedicada a esta función.

Diferencias entre los modelos A y A+ de Raspberry Pi

- Se disminuyó su tamaño en 2.1cm.
- Se mejoró la capacidad de conexión en caliente de dispositivos de consumos relativamente elevados, como los dongles WiFi, que en el modelo clásico causaban el reinicio del sistema.
- Integración del audio y el vídeo analógico en un único conector de 3.5mm.
- Aumento en la cantidad de pines de propósito general.
- Cambio de ranura SD por ranura microSD.
- Disminución en el consumo de energía.
- Mejoras en la calidad del audio, provistas por el empleo de una fuente de energía dedicada a esta función.

1.3.4. Raspberry Pi 2

A inicios de 2015 salió a la venta la versión 2 de Raspberry Pi. La figura 1.16 muestra la placa de Raspberry Pi 2 modelo B. Entre sus novedades se destaca el cambio de CPU a uno de cuatro núcleos, de arquitectura ARMv7 que garantiza la compatibilidad con distribuciones de linux como Ubuntu y otros sistemas operativos como Windows 10 que tengan soporte para dicha arquitectura [4].

Como se puede comprobar en la tabla 1.2 el nuevo modelo mejora considerablemente a su predecesor en cuanto a CPU, Memoria RAM y soporte para otros sistemas operativos producto de la modernización de la arquitectura del microprocesador. Según los fabricantes, la nueva placa es seis veces más potente que el último modelo de Raspberry Pi en cuanto a poder de cómputo. Por otra parte, se puede comprobar que pese a las mejoras incluidas en Raspberry Pi 2, el precio de venta del fabricante es el mismo [4].

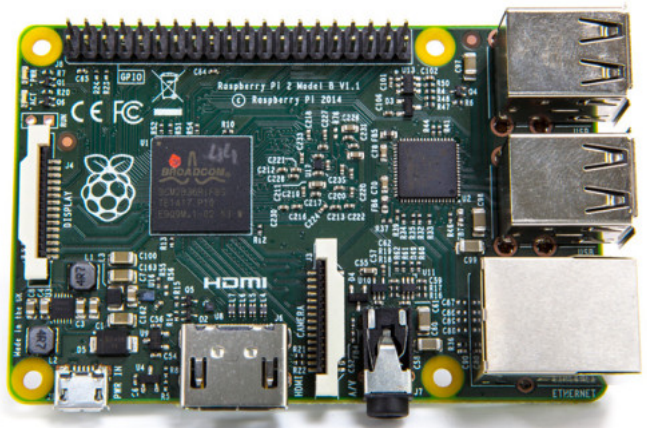

Figura 1.16: Raspberry Pi 2 modelo B.

	Raspberry Pi Modelo B+	Raspberry Pi 2 Modelo B
SoC	Broadcom BCM2835	Broadcom BCM2836
CPU	ARM11 ARMv6 700 MHz	ARM11 ARMv7 4 núcleos 900 MHz
GPU	Broadcom VideoCore IV 250 MHz	Broadcom VideoCore IV 250 MHz
RAM	512 MB LPDDR SDRAM 400 MHz	1 GB LPDDR2 SDRAM 450 MHz
USB 2.0	4	4
Vídeo	HDMI 1.4 @ 1920x1200 píxeles	HDMI 1.4 @ 1920x1200 píxeles
Almacenamiento	microSD	microSD
Ethernet	10/100 Mbps	10/100 Mbps
Tamaño	85,60 x 56,5 mm	85,60 x 56,5mm
Peso	45 g	45 g
Consumo	5v, 600mA	5v, 900mA
Precio	35 dólares	35 dólares

Tabla 1.2: Características técnicas de Raspberry Pi 2 modelo B y B+.

1.4. Comunicaciones en la robótica móvil

Existen diversas tareas donde una red de comunicación entre varias plataformas robóticas o entre un robot y una computadora personal juegan un

papel fundamental. Ejemplos de esto pudieran ser:

Permitir que varios robots se comuniquen entre ellos. Por ejemplo, compartir información de sensores, datos o cooperar para completar una tarea en común.

Controlar de forma remota uno o varios robots. Por ejemplo, enviar comandos de manejo para guiar una trayectoria o especificar metas que luego serán alcanzadas con los medios que cuente el robot.

Supervisar de forma remota los sensores de uno o varios robots. Por ejemplo, mostrar la imagen de una cámara a bordo de un robot, o los resultados de las mediciones de sensores de distancia.

Supervisar estados de uno o más robots Por ejemplo, la posición y la orientación en un ambiente común de múltiples robots. Esto permitiría un análisis del comportamiento y la efectividad de un equipo de robots en una tarea específica.

1.4.1. Bluetooth

Es una tecnología definida en el estándar IEEE 802.15.1. Se emplea en comunicaciones de muy corta distancia, mayormente en teléfonos móviles, computadoras portátiles y otros periféricos como audífonos. Es una alternativa conveniente al uso de cables para el intercambio de información entre dispositivos cercanos. No es necesario un ancho de banda grande para su funcionamiento y usa bajos niveles de potencia para trasmitir la información [13].

Bluetooth opera en la banda de 2.4 GHz a 2.4835 GHz. Típicamente el ancho de banda necesario para los enlaces se encuentra entre 1 y 3 Mbps pudiendo alcanzar una distancia de hasta 10 metros. La configuración de red estándar de Bluetooth se denomina Piconet. Consiste en un dispositivo maestro y hasta 7 dispositivos esclavos. Todas las comunicaciones ocurren entre el maestro y uno de los esclavos, los esclavos no pueden comunicarse directamente entre ellos [13].

1.4.2. ZigBee

En el campo de las comunicaciones inalámbricas de baja potencia y corto alcance existen tecnologías alternativas a Bluetooth. Un ejemplo es ZigBee, estandarizada según IEEE 802.15.4. Está especialmente diseñada para situaciones donde los requerimientos de ancho de banda y consumo de potencia

sean muy bajos pudiendo garantizar mayor tiempo de funcionamiento en dispositivos con baterías [13].

Se diseñó para ser más simple y económico que Bluetooth, haciéndolo rentable para incorporarse en dispositivos como sensores. La comunicación ZigBee opera en la banda de 2.4 GHz, su velocidad de transmisión es de 250 kbps y su configuración de red puede tener un máximo de 65535 nodos distribuidos en subredes de 255 nodos [13].

1.4.3. WiFi

El estándar 802.11, conocido popularmente como WiFi, es una de las tecnologías de comunicación inalámbricas más empleadas en la actualidad. WiFi define una serie de capas físicas que operan en varias frecuencias y proveen varias tasas de intercambio de datos, que varían en dependencia de la especificación. En el caso de la norma 802.11n se alcanzan velocidades de hasta 600Mbps [13].

La arquitectura de red WiFi que ha predominado con el tiempo es la denominada de infraestructura. Se compone de un dispositivo utilizado como punto de acceso a la red y el resto de los clientes. El punto de acceso difunde la información de su disponibilidad y mediante algún protocolo de autentificación, dependiendo de la seguridad que se requiera en la red, los clientes se conectan y acceden a los recursos disponibles mediante el punto de acceso. En función de la especificación, las tecnologías WiFi pueden operar en las bandas del espectro electromagnético correspondientes a 2.4 GHz o 5.0 GHz [13].

Al ser un estándar diseñado para redes locales, posee algunas ventajas sobre Bluetooth y ZigBee, distancias de cobertura más abarcadoras, mayores velocidades de intercambio de datos y facilidades propias del empleo de la torre de protocolos TCP/IP, totalmente compatibles con WiFi [13].

1.5. Tareas fundamentales de robots autónomos

Los robot móviles autónomos son aquellos que pueden realizar tareas en ambientes estructurados o no estructurados sin la supervisón continua de un humano. Un robot completamente autónomo debe ser capaz de [7]:

- Obtener información del ambiente donde se encuentra
- Trabajar un período considerable de tiempo sin la intervención de un humano

- Desplazarse sin dificultad por el ambiente de operación sin la asistencia de un humano
- Evitar situaciones perjudiciales para las personas, propiedades o el propio robot, a menos que sean parte de las especificaciones en su diseño

Se define como navegación a la técnica que guía a un robot móvil hacia su destino o a través de la trayectoria deseada en su ambiente de operación, evitando los obstáculos que pueden ser estáticos o dinámicos. El problema de la navegación puede ser dividido en cuatro sub-problemas [7]:

Percepción del ambiente. Consiste en extraer información de los sensores y procesarla.

Planificación de trayectorias. A partir de la información obtenida de los sensores y de la ubicación del destino, se crea una secuencia ordenada de puntos por los cuales el robot debe pasar.

Generación de la trayectoria. Se procesan los puntos obtenidos y se genera una trayectoria que permita recorrerlos de forma continua.

Controlador de trayectoria. Se debe garantizar que el robot siga la trayectoria calculada y corrija cualquier error durante su ejecución.

Para darle solución a estos problemas se han definido conceptos y sistemas generales que describen el modo de operación y las tareas que deben desarrollar los robots como parte de su funcionamiento. Estas tareas son la base de cualquier objetivo que se programe sobre un robot autónomo; constituyen recursos básicos, que independientemente de los detalles de su implementación, brindan resultados que son en gran medida muy similares para todos los robots [7].

1.5.1. Localización

La localización de un robot se describe normalmente mediante un número de variables. Para los robots móviles, son típicamente la posición y la orientación. En el caso particular de los robot móviles terrestres con ruedas, se puede describir completamente su posición y orientación mediante tres variables, como se muestra en la figura 1.17 [11].

La localización es la estimación de la posición a partir de los resultados obtenidos mediante la adquisición de datos de los sensores. Es una de las

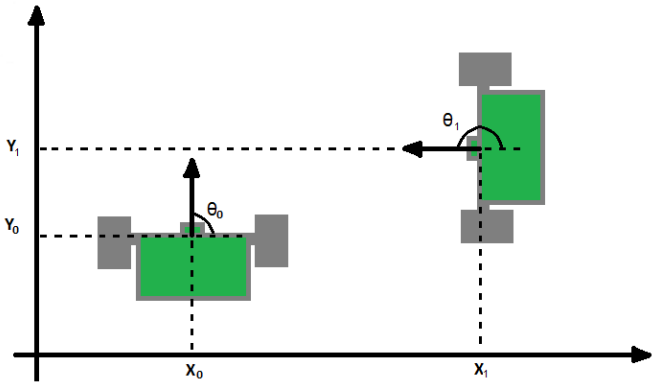

Figura 1.17: Posición y orientación de un robot móvil terrestre.

tareas fundamentales durante el proceso de navegación. Existen diversos mecanismos de localización, que dependen fundamentalmente de los sensores que se emplean y se dividen generalmente en dos categorías, localización absoluta y localización relativa [11].

Localización Relativa

También conocida por su nombre en inglés como *dead reckoning*, la localización relativa consiste en calcular la posición y la orientación del robot móvil a partir de una posición inicial conocida, utilizando sensores internos, sin conocer el entorno. Estos sistemas pueden clasificarse en dos tipos, debido la información que utilizan para llevar a cabo el posicionamiento [11].

- *Sistemas odométricos.*

  La odometría es una técnica que permite estimar la posición y orientación de un robot móvil terrestre a partir del ángulo de giro de sus ruedas. Para esto se utilizan comúnmente sensores en los motores o en las propias ruedas. Uno de los sensores más populares para medir velocidad angular en el eje de una rueda, o un mecanismo, es el *encoder*. Estos sensores pueden variar su principio de funcionamiento y entre ellos el más conocido es el *encoder* óptico, aunque también existen los resistivos, los magnéticos, entre otros [18].

  La precisión de un *encoder* está dada por la cantidad de pasos por vuelta, donde la separación angular entre dos pasos representa la resolución máxima del sensor. En la figura 1.18 se puede observar como

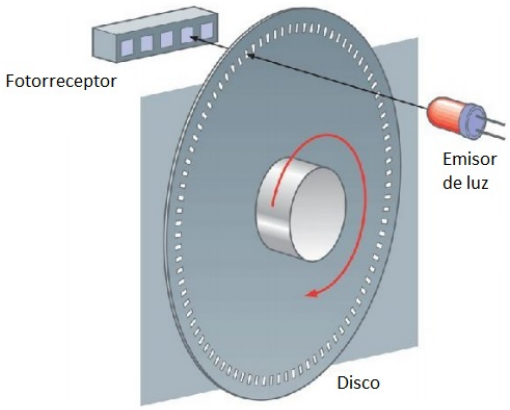

Figura 1.18: *Encoder* óptico incremental.

un *encoder* óptico, mediante un material fotosensible, recibe un pulso de luz cada vez que el LED incide sobre un agujero. En este caso los agujeros determinan los pasos del *encoder*. Para determinar el ángulo de giro de una rueda, basta con contar la cantidad de pulsos recibidos y multiplicar por el ángulo de separación entre los pasos, si el sentido de rotación se mantuvo invariable [18].

Una vez conocido el ángulo de giro de las ruedas, y factores como las dimensiones y la separación entre ellas, se puede estimar la posición del robot relativa al punto inicial, mediante una expresión matemática que depende del modelo mecánico del robot [18].

- *Sistemas de navegación inercial.*

  Un sistema de navegación inercial está conformado por una Unidad de Navegación Inercial o IMU (del inglés Inertial Measurement Unit) y un sistema de cómputo asociado a esta [18].

  La IMU es un dispositivo electrónico formado por una combinación de giroscopios y acelerómetros utilizados para medir velocidad de giro y aceleración, respectivamente. La navegación inercial se basa en la localización relativa a partir de la integración de las medidas tomadas por los sensores mencionados. Debido a la integración, pequeños errores repercuten notablemente en la posición estimada, por lo que la precisión de los sensores es un factor crítico en la estimación de la posición.

En aplicaciones de robots móviles, los valores de aceleración obtenidos de los sensores son pequeños, por lo que la relación señal-ruido es pequeña, lo que dificulta aún más la estimación [18].

La ventaja que poseen estos sistemas respecto a los sistemas odométricos es que no se ven afectados por los problemas del deslizamiento o de las irregularidades del suelo en que se desplaza el robot. En la práctica, estos sistemas son más fiables y precisos que los sistemas odométricos, pero son más caros y más frágiles [18].

Ambos sistemas se suelen combinar con sistemas de posicionamiento absoluto debido a que el error en la estimación de la posición es acumulativo en ambos casos [18].

Localización Absoluta

Los métodos de localización absoluta se basan en obtener la posición y orientación del sistema de acuerdo a una referencia externa al mismo, aprovechando elementos de un entorno conocido. A continuación se analizan algunos de los más empleados según [18].

- *Brújulas Electrónicas.*

  Las brújulas electrónicas permiten medir el ángulo de orientación con respecto al campo magnético terrestre. Estos dispositivos basan su funcionamiento en fenómenos físicos relacionados con el magnetismo. El campo magnético terrestre es distorsionado por diversos factores, entre los cuales se encuentran estructuras metálicas y líneas de transporte de energía eléctrica; por lo tanto estas brújulas limitan su empleo a interiores. Estos dispositivos normalmente forman parte de sistemas de navegación inercial para garantizar una referencia externa fiable [18].

- *Sistema de Posicionamiento Global.* El Sistema de Posicionamiento Global (GPS, por sus siglas en inglés) es un sistema de navegación por satélites, que fue inicialmente desarrollado para uso militar, pero actualmente está disponible para sistemas de navegación civiles. Existen al menos 24 satélites GPS operando en todo momento, orbitando a una altura de 20 kilómetros en seis planos inclinados respecto al plano del ecuador de la Tierra [18].

  Cada satélite transmite constantemente su ubicación y la hora actual. Estos están sincronizados de manera tal que la transmisión de sus señales se realiza al mismo tiempo. Los receptores leen las transmisiones

de dos o más satélites y teniendo en cuenta la posición instantánea de los satélites y la diferencia de tiempo en que las transmisiones son recibidas pueden inferir la posición del receptor. En las aplicaciones de GPS el tratamiento de los tiempos es crítico. Debido a que las mediciones de los intervalos llevadas a cabo por el receptor están en el orden de los nanosegundos es muy importante la precisión del reloj. Además es muy importante que los satélites estén bien sincronizados por lo que estos disponen de relojes atómicos y son actualizados regularmente por estaciones en tierra [18].

El receptor GPS requiere cuatro satélites para obtener cuatro variables: tres posiciones referidas a ejes y una corrección de tiempo. Esta necesidad constituye una limitación; debido a que las transmisiones de los satélites son de muy baja potencia, las lecturas requieren una línea de comunicación directa con el satélite, por esto, en espacios confinados como ambientes urbanos con altos edificios o bosques densos, es difícil recibir las señales de cuatro satélites de manera acertada. La mayoría de los espacios cerrados tampoco permiten la visibilidad del cielo para un correcto funcionamiento de los receptores GPS. Debido a estas limitaciones la utilización del GPS en aplicaciones de robótica móvil se ha enfocado en proyectos de robots en espacios abiertos y máquinas de vuelo [18].

- *Balizas.* Una manera de determinar posición para entornos cerrados es el sistema de balizas o *beacons* (por su nombre en inglés). Este sistema utiliza la triangulación, la trilateración, o la combinación de estas para obtener la ubicación del robot con respecto a las balizas de las cuales se conoce su posición.

  Estas balizas deben distribuirse en el entorno de manera que el robot pueda reconocerlas fácilmente. Existen dos tipos de balizas:

  - Balizas activas, que son aquellos dispositivos que emiten señales que el robot debe procesar, por ejemplo, señales de radio frecuencia, infrarrojos, ultrasonidos, entre otros.
  - Balizas pasivas, que pueden ser marcas naturales del terreno, reflectores, códigos de barra, entre otros, que el robot debe reconocer.

1.5.2. Generación de mapas

Para una amplia gama de aplicaciones de robótica es necesario un modelo del ambiente donde opera el robot. Ejemplos de esto son las tareas de transporte, levantamiento de planos o rescate y salvamento. Adquirir un modelo preciso del ambiente es una condición necesaria para la autonomía del robot. El aprendizaje de mapas ha sido uno de los tópicos de más amplia investigación en la comunidad científica vinculada a la robótica en las últimas décadas. Es un problema con alto grado de complejidad, especialmente si se debe llevar a cabo en ambientes dinámicos o usando técnicas cooperativas mediante varios robots [16].

Para alcanzar una autonomía real y realizar un correcto aprendizaje del mapa del terreno, el robot deberá integrar este proceso con la localización y la planificación de trayectorias de forma concurrente. De forma general, estas tareas no pueden ser resueltas independientemente. De ahí que frecuentemente se haga referencia a la generación de mapas como a un proceso más general conocido en la robótica como SLAM, Localización y Mapeo simultáneo por sus siglas en inglés. En el epígrafe 1.5.5 se analiza el SLAM con mayor detalle.

1.5.3. Planificación de Trayectorias

El movimiento de un robot puede ser considerado una trayectoria en un espacio de puntos que representan posibles localizaciones. Además dicha trayectoria debe permanecer en el subespacio de puntos en los cuales no exista ninguna colisión entre el robot y los obstáculos del ambiente. El problema de la planificación de trayectorias puede ser formulado como la determinación de una trayectoria para llegar de un punto a otro de forma eficiente evadiendo los obstáculos del camino [11].

Planificación de Trayectoria Holonómica

Este método se emplea cuando todos los grados de libertad de las variables de posición del robot pueden ser cambiados independientemente [11].

En este caso, el problema se resume a construir un mapa del espacio libre de obstáculos y encontrar un conjunto de puntos que generen la trayectoria deseada [11].

Las primeras soluciones para la planificación de trayectorias se basaron en algoritmos heurísticos y métodos aproximados. Generalmente se descomponía el espacio en celdas adyacentes, diferenciándolas según la disponibili-

dad basada en los obstáculos, y luego se analizaba el grafo resultante para encontrar soluciones [11].

A principios de la década de 1980, investigaciones demostraron como describir entornos mediante ecuaciones e inecuaciones algebraicas con coeficientes enteros. De esta forma se comenzó a tratar el problema de la planificación con métodos basados en Geometría Algebraica Real. Al mismo tiempo la Geometría Computacional con restricciones combinatorias proporcionó varios métodos eficientes para sistemas robóticos específicos, teniendo en cuenta restricciones prácticas, como los cambios en el ambiente [11].

Planificación de Trayectoria no Holonómica

Una restricción no holonómica es aquella que reduce el espacio de velocidades admisibles de las variables de configuración, sin reducir el espacio de variables de configuración. Los sistemas mecánicos no-holonómicos poseen al menos una restricción no holonómica. Un ejemplo de sistema no holonómico es un automóvil, que no puede rotar sobre su eje sin variar su posición [11].

En la formulación del problema de la planificación se comenzaron a considerar restricciones no holonómicas. Esto trajo como consecuencia que cualquier trayectoria encontrada en un espacio determinado, no tenía por qué corresponder necesariamente a una trayectoria realizable. De esta forma surgió la Planificación de Trayectorias No Holonómicas, cuyo modelo resulta mucho más acertado para la mayoría de los robots [11].

1.5.4. Control de Trayectorias

El controlador de trayectorias es el sistema encargado de garantizar una buena precisión en el movimiento durante el proceso de navegación. Existen tres tipos diferentes de tareas que requieren de control de trayectorias:

Movimiento punto a punto: El robot debe alcanzar una meta determinada dada una posición inicial conocida.

Seguimiento de rutas: El robot debe alcanzar y seguir una ruta determinada en un espacio cartesiano dada una posición inicial conocida que pudiera estar dentro de la ruta, aunque no es una condición necesaria.

Seguimiento de trayectorias: El robot debe alcanzar y seguir una trayectoria en el espacio cartesiano. La principal diferencia con el seguimiento de rutas es que las trayectorias tienen restricciones de tiempo asociadas a cada punto de las mismas.

1.5.5. SLAM

La localización y mapeo simultáneos (SLAM por sus siglas en inglés) se ha propuesto para resolver de forma concurrente la estimación de la posición del robot y la generación de mapas. En las últimas décadas algoritmos de SLAM basados en filtros de Kalman extendidos (EKF por sus siglas en inglés) y filtros de Partículas de Rao-Blackwellized han arrojado resultados notables para ambientes estáticos [12].

Los ambientes reales en los que deben operar los robots móviles suelen cambiar mientras ocurre el SLAM, ejemplos de esto pueden ser cambios en la posición de muebles en el interior de una casa, o personas y vehículos que se desplazan por exteriores. El SLAM en ambientes dinámicos tiene que enfrentar inconsistencias entre las observaciones realizadas con posterioridad a objetos que cambiaron su posición y las predicciones elaboradas a partir del mapa generado. Estas inconsistencias inducen errores en la estimación de la posición debido a la asociación incorrecta de información [12].

Para solucionar el SLAM en ambientes de continuos cambios se han desarrollado técnicas basadas en el aprendizaje continuo de nuevos puntos claves del mapa, sensores de muy alta precisión y posibilidad de filtrar observaciones que se puedan inferir como corruptas producto de cambios detectados en el ambiente. No obstante, no es trivial determinar cuando una observación puede ser considerada corrupta, especialmente usando sensores de alta incertidumbre o baja resolución. El problema del SLAM aun continúa abierto a mejores soluciones y constituye uno de los objetos de estudio más desarrollados por los investigadores en el campo de la robótica en las últimas décadas [12].

1.6. Sistemas de detección de obstáculos

Para la generación de mapas, la localización y la planificación, los robots requieren sistemas que les permitan detectar elementos del ambiente y determinar cuales representan obstáculos para su movimiento. Con este propósito se han desarrollado sistemas basados en distintos tipos de sensores, que varían sus características y efectividad dependiendo de la aplicación y del medio donde se emplean. Estos sistemas también han sido usados para extraer características específicas del terreno [9].

1.6.1. Sensores ultrasónicos

Los sensores ultrasónicos, también conocidos como sonares, son de los más empleados para la detección de obstáculos en interiores y para distancias relativamente pequeñas, debido a su simplicidad y sus bajos costos. El funcionamiento de estos sensores se basa en analizar el eco resultante de enviar una onda sonora hacia el frente del sensor, para obtener información de la superficie impactada, particularmente la profundidad a la que se encuentra. Una desventaja considerable de este tipo de sensores es la cantidad necesaria para obtener una referencia panorámica del ambiente alrededor del vehículo. Es bastante común emplear varios anillos de múltiples sensores en robots omnidireccionales. El número de sensores requerido para un campo visual adecuado no es la única desventaja: algunos estudios sobre el tema demuestran que para adquirir información con precisión, los sensores deben estar colocados de forma perpendicular a los obstáculos [9].

Las principales desventajas de hacer un sistema de detección de obstáculos basado en sensores ultrasónicos son [9]:

- Poca direccionalidad que limita la precisión en la determinación de la posición de bordes en un rango de 10 a 15 cm, dependiendo de la distancia del obstáculo y el ángulo entre su superficie y el haz.
- Frecuentes falsas lecturas causadas por el ruido provocado por fuentes externas o reflexiones de sensores cercanos, que no siempre pueden ser filtrados y provocan falsas detecciones de obstáculos.
- Reflexiones especulares que ocurren cuando el ángulo entre el frente de onda y el vector normal a una superficie impactada es muy grande. En este caso la superficie refleja las ondas incidentes lejos del sensor y como resultado el obstáculo no es detectado.

1.6.2. Láser

Estos sistemas emplean un rayo láser que es reflejado en un espejo que apunta a la región que se quiere explorar. La resolución angular puede ser tan pequeña como 0.25 grados. Existen dos variantes fundamentales usadas actualmente. La primera consiste en la emisión de un haz de forma continua, que al regresar incide en un sensor fotosensible y se estima la distancia en función del tiempo que demoró la propagación. La otra variante envía varios pulsos de luz y promedia el tiempo de demora por cada pulso para los cálculos de distancia. A diferencia del primero este sistema no resulta

peligroso para los ojos humanos y los errores de medición pueden ser filtrados con menor dificultad [9].

La tecnología láser se han empleado para estimar la posición de robots y se ha combinado con cámaras para la detección de meteoritos en la Antártida. Los resultados obtenidos han demostrado que la fidelidad de los sistemas basados en láser es superior a la obtenida con sensores ultrasónicos [9].

El empleo de este tipo de sistemas ha aumentado en los últimos años debido a que la información obtenida de distancia es independiente a la cantidad de luz en el ambiente, lo que representa una ventaja fundamental sobre sensores como cámaras. La principal desventaja es que la detección se realiza únicamente en un plano, si el obstáculo se encuentra por encima o por debajo del sensor, este no será detectado. Estos sensores además se ven afectados por el polvo, la lluvia y la nieve, causando falsas lecturas [9].

1.6.3. Cámaras

Los sistemas de visión basados en cámaras, con posibilidades de detección y evasión de obstáculos, son relativamente complejos de desarrollar, especialmente si se desea un desempeño en tiempo real con el mínimo costo computacional posible. En las últimas dos décadas, estos sistemas se han desarrollado considerablemente y se ha extendido su uso tanto para interiores como exteriores, convirtiéndose en una de las mayores áreas de investigación en la comunidad de robótica móvil [9].

Existen numerosas técnicas para realizar detección de obstáculos empleando cámaras. Entre las más utilizadas se pueden señalar:

Estereoscopía se basa en la extracción de información de profundidad a partir de dos cámaras situadas a una distancia conocida y enfocadas hacia la misma escena

Detección basada en apariencia se centra en el análisis de las características del ambiente

Flujo óptico se basa en el análisis de imágenes tomadas de forma consecutiva por la misma cámara

1.6.4. Radares

Otra variante de sensores de detección de obstáculos, son los radares de ondas milimétricas. Poseen muy buena resolución angular y es posible extraer una imagen tridimensional del ambiente en la trayectoria del robot. La

principal desventaja de su empleo son los altos precios que poseen, haciendo que no sean asequibles para muchos investigadores. Han sido utilizados en robots móviles diseñados para trabajar en ambientes polares, donde la nieve constantemente provoca la detección de falsos obstáculos en sistemas basados en tecnología láser o visión estereoscópica [9].

Capítulo 2

Diseño de la plataforma

A continuación se describen detalladamente los elementos que conforman la plataforma desarrollada para la experimentación. El capítulo está dividido en tres partes: inicialmente se expone el diseño mecánico, luego el hardware, y al final los detalles fundamentales de la implementación software de los subsistemas con que cuenta el robot.

2.1. Estructura mecánica

El prototipo se concibió según el modelo de tracción diferencial que se explica en el epígrafe 1.1.1, por las facilidades en el control anteriormente expuestas. Las dimensiones y componentes fueron seleccionadas en función de optimizar la precisión en el movimiento y garantizar la escalabilidad de la plataforma.

2.1.1. Motores

Se emplearon dos motores de corriente directa, con reductores de velocidad en el eje de salida y *encoders* de efecto hall[1] (ver figura 2.1). Los parámetros de mayor peso en la selección fueron el torque y la resolución de los *encoders*. En la tabla 2.1 se muestran las características de los motores seleccionados.

[1]Ocurre cuando se sitúa un imán de forma perpendicular a una cara de un fino rectángulo de oro por el cual fluye una corriente. Aparece una diferencia de potencial en los bordes opuestos proporcional a la corriente que fluye por el conductor.

Parámetro	**Valor Nominal**	**Unidad**
Tensión nominal	12	Vdc
Consumo libre de torque	140	mA
Consumo máximo	440	mA
Torque máximo	2.55	Kg/cm
Relación de transmisión	90.3:1	
Velocidad libre de torque	75	RPM
Velocidad a torque máximo	62	RPM
Diámetro del eje	4	mm
Largo	67.5	mm
Resolución de los *encoders*	270.9	pasos/vuelta
Peso	100	gr

Tabla 2.1: Características técnicas de los motores empleados.

Figura 2.1: Motores de corriente continua

2.1.2. Ajuste del ángulo de la cámara

La posición vertical y el ángulo de inclinación de la cámara son factores claves para la detección de obstáculos mediante visión computacional. Debido a que el funcionamiento y la precisión de estos métodos son dependientes del ajuste de la cámara, no es trivial encontrar a priori la posición óptima donde situarla. Para lograr que cubra el rango más amplio posible, se precisa de varias mediciones y pruebas con la cámara ya instalada. Por este motivo se diseñó una pieza que permite variar su ángulo con respecto al suelo y poder calibrar la región de visualización en cualquier momento. En la figura 2.2 se observa un esbozo de la forma y dimensiones de la plataforma. Dicha pieza puede ser observada en la parte delantera.

2.1.3. Dimensiones

Las dimensiones de un robot están sujetas a varias restricciones. En este caso, se priorizó la escalabilidad y la posibilidad de soportar mayores cargas, en lugar de minimizar su tamaño. En la figura 2.2 se puede observar un esbozo del diseño mecánico y las dimensiones.

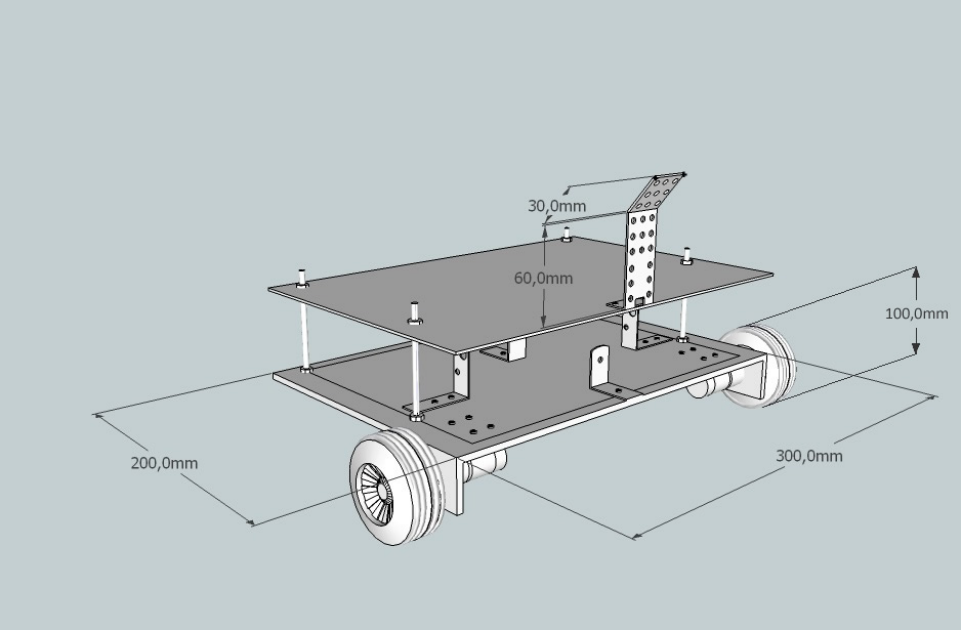

Figura 2.2: Forma y dimensiones del prototipo

2.2. Hardware

Para el diseño del hardware, la premisa fue desarrollar elementos modulares que interactúen entre sí. De esta forma la sustitución o mejora de cualquier módulo hardware no debe implicar modificaciones en el resto. Otra ventaja de desarrollar los módulos con cierta independencia, es la facilidad que brindan en la detección de cualquier falla en el funcionamiento global de la plataforma.

En cuanto a los elementos de hardware, el robot puede ser descrito según los módulos de control de motores, interfaces de comunicación, sensores, fuentes de alimentación y la unidad central de procesamiento.

2.2.1. Circuito de control de motores

Para controlar los motores se empleó el circuito integrado L298N, que contiene dos puentes en H completos y una entrada de habilitación para cada motor. Para el control del sentido de los motores el circuito cuenta con dos entradas digitales por motor. Si ambas entradas están en el mismo estado el motor se detiene. De lo contrario, gira hacia un sentido o el otro dependiendo de cual entrada esté en estado activo. Para controlar la velocidad es necesario conectar una señal de PWM a la entrada de habilitación de cada motor. De esta forma, variando el ciclo útil de la señal PWM, se logra variar la componente de voltaje promedio que se le suministra al motor. En la figura 2.3 se observan las señales digitales necesarias para manejar el circuito L298N, nombradas *IN1*, *IN2*, *IN3*, *IN4*, *PWM1* y *PWM2*.

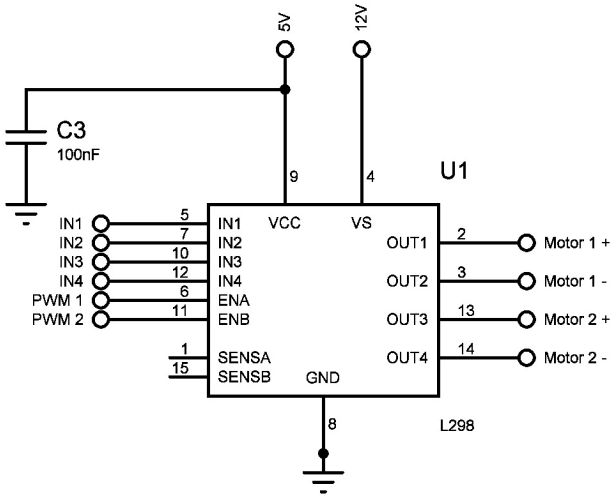

Figura 2.3: Conexiones de las señales del circuito L298N

La frecuencia máxima de la señal eléctrica generada por los *encoders* se puede calcular usando la ecuación 2.1.

$$f_m = w_m E_r \tag{2.1}$$

Donde:

f_m	Frecuencia máxima de la señal	$[pasos/s]$.
w_m	Velocidad angular máxima del motor	$[vueltas/s]$
E_r	Resolución de los *encoders*	$[pasos/vueltas]$

La velocidad máxima que pueden alcanzar los motores seleccionados es de 75 RPM, equivalente a 1.25 vueltas en un segundo. La resolución de los *encoders* es de 270.9 pasos por vuelta. Empleando la ecuación 2.1 se obtiene que la frecuencia máxima de los pulsos de salida de los *encoders* es 338.7 pulsos por segundo.

Una vez conocido esto, resulta conveniente filtrar la señal de salida de los *encoders* con un filtro pasa-bajo, que elimine altas frecuencias que pudieran causar falsas lecturas. La figura 2.4 muestra los filtros pasa-bajos diseñados para la frecuencia de corte 400Hz, donde *Encoder1* y *Encoder2* son las señales que deben ser leídas para el conteo de pulsos.

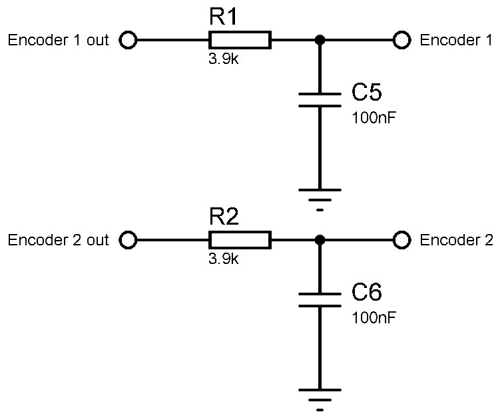

Figura 2.4: Filtros pasa-bajos a la salida de los codificadores

Se añadió un sistema de adquisición de datos para medir el voltaje de la batería, con la intención de estimar la carga restante. La figura 2.5 muestra el circuito de entrada de la alimentación de los motores y la generación de la señal analógica *SupplyFB* que permite conocer el voltaje de la batería.

Debido a la alta inductancia de los motores, se agregó un arreglo de diodos de conmutación rápida para la protección contra sobre-voltaje de los circuitos de control. En la figura 2.6 se observa la conexión de las salidas del L298N con los motores.

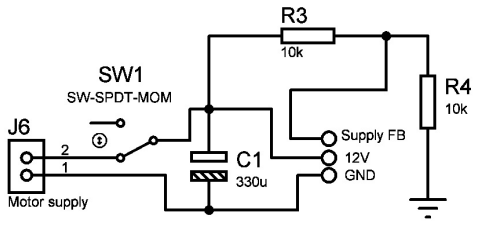

Figura 2.5: Alimentación de motores y medición del voltaje de la batería

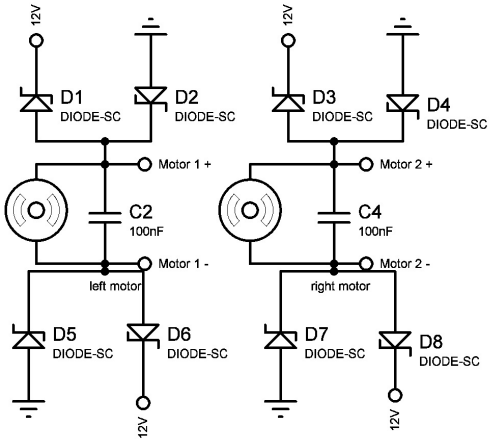

Figura 2.6: Protección de los motores

Para el control del sentido y la velocidad de los motores se requiere un circuito que se encargue de leer la salida de los *encoders*. En función de la cantidad de pulsos recibidos en un intervalo periódico de tiempo, han de realizarse acciones de control. En este caso, estás acciones implican variar el ciclo útil de la señal PWM que habilita los motores o los bits que controlan el sentido de cada motor en el puente en H.

Se realizó un diseño basado en la plataforma Arduino Uno, por ser un sistema completamente abierto, las facilidades que brinda para la implementación de un controlador de velocidad, las ventajas de utilizar los estándares de *shields*, la posibilidad de delegar algunas tareas de cómputo en este procesador y la interfaz de comunicación UART sobre USB que facilita la conexión

con múltiples dispositivos. La figura 2.7 muestra un diagrama con los nombres de todos los pines de Arduino y a la derecha se observa la asignación realizada a cada señal de las mencionadas anteriormente.

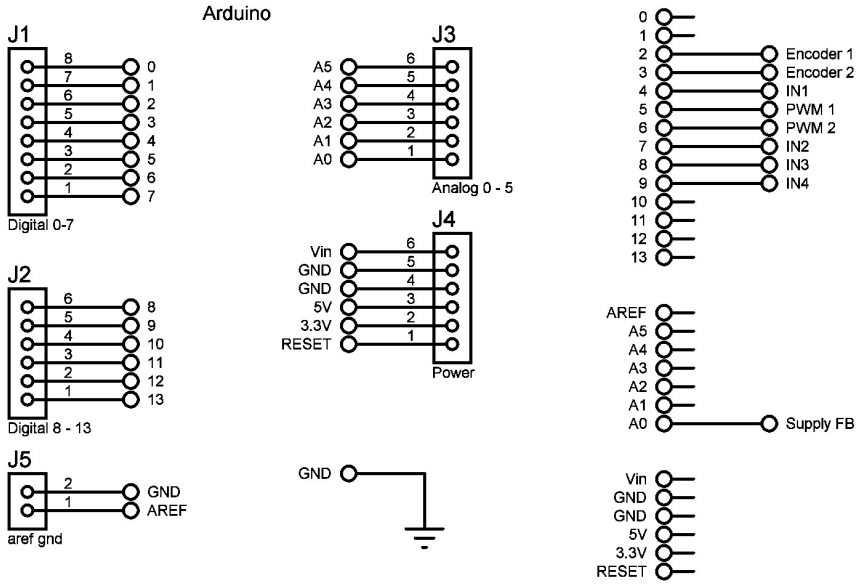

Figura 2.7: Asignación de los pines de Arduino en el control de los motores

En la figura 2.8 se observa una vista tridimensional del circuito de control de motores con todos los elementos integrados en una placa de circuito impreso diseñada según el estándar de *shields* de Arduino.

2.2.2. Procesador principal

Las diversidad y complejidad de las tareas que debe realizar el robot requieren un procesador con una capacidad de cómputo muy superior a la que ofrece el microcontrolador de Arduino. Se empleó la SBC Raspberry Pi modelo B, por sus buenas prestaciones, las facilidades que brinda el sistema operativo Linux, la inmensa cantidad de aplicaciones y bibliotecas con que cuenta, las múltiples interfaces de comunicación y su excelente relación calidad-precio.

La comunicación entre la Raspberry Pi y el circuito de control de motores

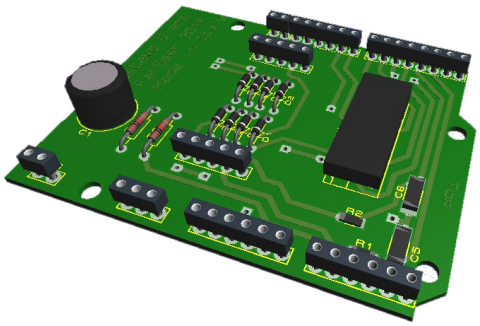

Figura 2.8: Vista 3D de la placa de control de motores

se realiza mediante el protocolo serie UART sobre USB. Esto es posible porque Arduino cuenta con un módulo hardware, que transforma los datos recibidos por USB en bits asíncronos compatibles con el protocolo UART, los cuales pueden ser interpretados por el microcontrolador interno. En el otro extremo, Raspberry Pi cuenta con un driver que simula el comportamiento de un puerto serie y envía los datos por USB hacia el Arduino.

2.2.3. Comunicaciones

Se empleó el estándar 802.11, conocido popularmente como WiFi para las comunicaciones inalámbricas. Para dotar a la SBC de una interfaz de este tipo, se empleó un adaptador WiFi-USB, compatible con los estándares 802.11b/g/n, como se muestra en la figura 2.9.

Figura 2.9: Adaptador Wifi-USB empleado

El uso de esta tecnología en lugar de otras más económicas en cuanto a costo y consumo de energía, como podrían ser Bluetooth o Zigbee, se debe a la diferencia considerable de ancho de banda, el radio de cobertura de la señal y las facilidades que brinda el protocolo IP sobre WiFi.

2.2.4. Cámara

El robot se ha diseñado con los objetivos de estudiar algoritmos de visión por computadora y realizar procesamiento de imágenes, por tanto la calidad de las imágenes juega un papel fundamental en el desarrollo de las tareas que se planifican.

Se decidió emplear en el diseño el módulo PiCamera, ver figura 2.10, por tener un desempeño muy superior a las *webcams* en cuanto a resolución máxima y cantidad de cuadros por segundo unido a sus facilidades de integración con Raspberry Pi mediante la interfaz CSI.

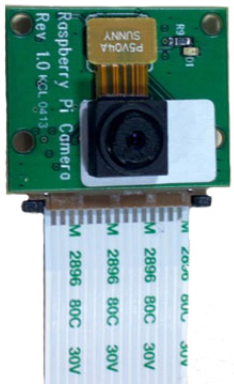

Figura 2.10: Módulo Pi Camera.

2.2.5. Fuentes de alimentación

Todos los circuitos empleados en el diseño requieren una alimentación de 5V DC y los motores requieren de 12V DC. Para garantizar la autonomía de la plataforma se deben emplear baterías que garanticen al menos 2 horas de trabajo de forma continua.

Los motores tienen un consumo máximo nominal de 0.88A. Para su alimentación se decidió emplear una batería de 12V y carga de 7000mAh como

se muestra en la figura 2.11 a.

El consumo máximo del resto de los circuitos es de aproximadamente 1.6A. En este caso se optó por la elección de un PowerBank con salida de 5V, capacidad de entregar hasta 2.2A y una carga de 5000mAh, sobre la posibilidad de construir una fuente que se alimentara de la batería de 12V. La elección se basó en aumentar la autonomía y disminuir la probabilidad de fallos. El dispositivo empleado se muestra en la figura 2.11 b.

Figura 2.11: Fuentes de alimentación del sistema.

2.2.6. Esquema de conexiones

La figura 2.12 muestra la conexión entre todos los elementos del hardware del sistema y las interfaces que se emplean en cada caso. A continuación se listan los componentes identificados en la imagen.

1. Raspberry Pi modelo B
2. Arduino Uno
3. Circuito controlador de motores
4. Power Bank
5. Módulo de cámara de Raspberry Pi
6. Adaptador Wifi-USB
7. Motor izquierdo
8. Motor derecho
9. Batería 12V

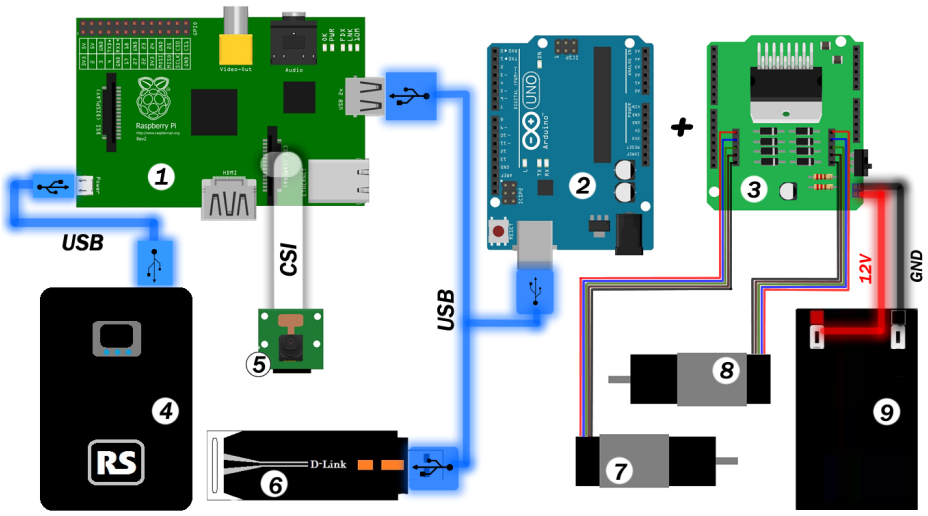

Figura 2.12: Diagrama general del sistema diseñado

2.3. Software

A continuación se describe toda la programación realizada en los controladores del robot y en aplicaciones externas que interactúan con la plataforma. En cada caso se especifica el procesador que ejecutará el programa.

2.3.1. Controlador de velocidad de motores

El control automático de la velocidad de los motores es una tarea imprescindible para garantizar el correcto funcionamiento de otros controladores. El controlador debe garantizar que la velocidad real de los motores sea la deseada, sin importar las condiciones del terreno, la inclinación del plano de desplazamiento, la carga restante de la batería, o cualquier otro evento externo.

Para medir la velocidad angular real en los motores, se procede a contar periódicamente la cantidad de pasos de los *encoders*, se ajusta el valor para expresarlo en radianes y se divide el resultado entre el período de muestreo como muestra la ecuación 2.2.

$$w = \frac{\Delta_e 2\pi}{TE_r} \tag{2.2}$$

Dónde:

w	Velocidad angular del motor	$[Rad/s]$
T	Intervalo de muestreo	$[s]$
Δ_e	Cantidad de pasos de los *encoders* en el intervalo T	$[pasos]$
E_r	Resolución de los *encoders*	$[pasos/vueltas]$

Una vez calculada la velocidad real, se procede a analizar la diferencia con la velocidad deseada. Por comodidad se define como $e(t)$ a la diferencia entre el valor deseado y el valor real de la variable de control (en este caso la velocidad de rotación de los motores) para cada instante de tiempo como muestra la ecuación 2.3.

$$e(t) = w_e(t) - w_r(t) \tag{2.3}$$

En función de $e(t)$ se toma una nueva acción de control $u(t)$, en este caso, una modificación en el ciclo útil de la señal modulada en ancho de pulso que controla la energía promedio entregada a los motores. Nótese que $e(t)$ coincidirá en su signo con la variación que debe ser aplicada sobre u.

La nueva acción de control es determinada en cada período por un controlador del tipo PID, y puede ser descrita en el dominio del tiempo continuo según la ecuación 2.4.

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d T_d \frac{de(t)}{dt} \tag{2.4}$$

La selección de las constantes K_p, K_i y K_d, dada la naturaleza del problema, debe garantizar un control de alta precisión y un tiempo de establecimiento mínimo.

La implementación de este controlador se realizó en la plataforma Arduino. De esta forma se redujo parte de la carga de procesamiento en el procesador principal. El funcionamiento está basado en 4 subrutinas que se explican a continuación.

Lectura de los encoders

Para la atención de los *encoders* se emplearon dos subrutinas ejecutadas por interrupción, una por cada motor. Cada vez que un motor gira el intervalo correspondiente a un paso de los *encoders*, ocurre una interrupción y se aumenta el valor de una variable de conteo. Esto permite que de forma independiente a la ejecución del programa principal, la cuenta de los pasos de los *encoders* se mantenga actualizada, garantizando la mayor precisión posible en la estimación de la velocidad de rotación.

PID

Para realizar el control de velocidad, se programó una variante discreta del controlador PID. La subrutina correspondiente se ejecuta cada 25ms mediante el empleo de un temporizador y su interrupción. En cada tiempo de ejecución y para cada motor, se toman los valores almacenados del conteo de los pasos de los *encoders*, se estima la velocidad usando la ecuación 2.2, se calcula el error usando la ecuación 2.3 y mediante una variante discreta de la ecuación del controlador PID, se obtiene el nuevo valor de ciclo útil que debe tomar la señal modulada en ancho de pulso que maneja la energía promedio suministrada al motor.

Programa principal

La subrutina del programa principal se limita a atender el arribo de información por el puerto serie, interpretar esta información y ejecutar una acción determinada en función del comando recibido. Se definió un protocolo de seis comandos, que permite que el procesador principal obtenga la información necesaria de la placa Arduino y pueda modificar parámetros del controlador.

Las funciones de los comandos implementados por el protocolo son:

- Establecer la velocidad deseada en los motores.
- Obtener los valores del conteo de los pasos de los *encoders*.
- Obtener los valores reales de velocidad de los motores.
- Obtener información del voltaje de la batería.
- Modificar parámetros del controlador como K_p, K_i y K_d.
- Reiniciar el conteo de los pasos de los *encoders*.

2.3.2. Controlador de Trayectorias

En la sección 1.5.4 se mencionó que existen tres tipos de funciones que puede realizar un sistema de control de trayectorias. A continuación se explican brevemente.

Movimiento punto a punto: Puede ser considerado como un problema de estabilización. Para su solución, se procede a minimizar la función de error, que consiste en la diferencia entre la localización actual y

la localización deseada dentro de la trayectoria planificada. La figura 2.13 muestra una trayectoria punto a punto donde X_0 y X_2 denotan el inicio y el fin de la trayectoria, X_1 representa el punto sobre la trayectoria donde el robot debería encontrarse y X_1' corresponde a la posición real del robot.

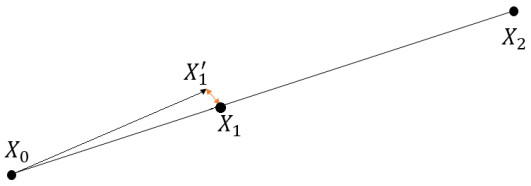

Figura 2.13: Control de trayectoria punto a punto.

Seguimiento de rutas: Se le brinda al controlador una descripción geométrica de la ruta cartesiana a seguir. Para esta tarea no existen restricciones temporales y se toma en cuenta solamente la disposición espacial de los puntos de la ruta y la posición del robot. Producto de la independencia del tiempo en la realización de la trayectoria este problema puede ser tratado como un problema de estabilización donde la función de error a minimizar es la distancia entre la trayectoria que se está realizando y la trayectoria ideal. La figura 2.14 muestra la representación de una ruta, donde X_0, X_1, X_2 y X_3 denotan los puntos por donde debe transitar el robot y X_1' corresponde a la posición real del robot.

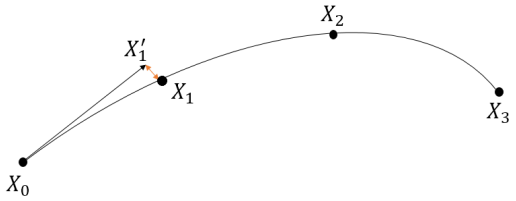

Figura 2.14: Control de trayectoria sin restricciones temporales.

Seguimiento de trayectorias: El robot debe seguir una ruta cartesiana rigiéndose por restricciones temporales, lo cual es equivalente a se-

guir a otro robot móvil. La trayectoria puede ser dividida en una ruta geométrica parametrizada y una regla de tiempo para cada parámetro. Esta división no es estrictamente necesaria pero en muchos casos puede representar una simplificación considerable del problema. La solución consiste en la minimización del error bidimensional resultante de las diferencias de tiempo y distancia entre la trayectoria actual y la trayectoria ideal [11]. La figura 2.15 muestra la representación de una trayectoria con restricciones temporales, donde X_0, X_1, X_2 y X_3 denotan los puntos por donde debe transitar el robot, t_0, t_1, t_2 y t_3 describen los instantes de tiempo en los cuales el robot debe encontrarse en las posiciones asociadas y X_1' corresponde a la posición real del robot en el instante t_1'.

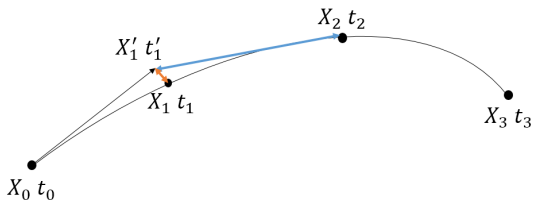

Figura 2.15: Control de trayectoria con restricciones temporales.

Se empleó un controlador de trayectorias realizado como parte de una investigación precedente para realizar esta tarea en la plataforma diseñada. Los detalles de la implementación y algunos resultados experimentales pueden ser comprobados en los trabajos de diploma de Jorge Silvio Delgado [18] y Abel Domínguez [8].

Para adecuar este controlador al sistema diseñado fue necesario programar una interfaz con las funciones de accionamiento de motores. Estas funciones se encargan de mandar los comandos necesarios a través del puerto serie empleando el protocolo descrito en el epígrafe anterior.

2.3.3. Localización mediante odometría

La localización por odometría permite estimar la posición dada una condición inicial. En un robot móvil con ruedas de accionamiento diferencial, se deben registrar las distancias recorridas por las ruedas controladas. A través de los sensores (*encoders*) se obtiene la cantidad de pulsos registrados en el tiempo de muestreo y se estiman los ángulos de rotación de las ruedas. Con

estos ángulos se puede estimar el desplazamiento del robot y el ángulo que giró su estructura al aplicar las siguientes ecuaciones [17]:

$$\Delta_S = \frac{r}{2}(\Delta\phi_R + \Delta\phi_L) \tag{2.5}$$

$$\Delta_\Theta = \frac{r}{d}(\Delta\phi_R + \Delta\phi_L) \tag{2.6}$$

Donde:

Δ_S	Desplazamiento lineal del robot	$[m]$
Δ_Θ	Desplazamiento angular del robot	$[rad]$
r	Radio de las ruedas	$[m]$
d	Distancia entre los ejes de las ruedas	$[m]$
$\Delta\phi_R$	Desplazamiento angular de la rueda derecha	$[rad]$
$\Delta\phi_L$	Desplazamiento angular de la rueda izquierda	$[rad]$

A través de estas cantidades incrementales y un estado inicial se puede obtener una estimación del estado actual, para lo cual se utiliza el método de integración Runge-Kutta de segundo orden:

$$X_k = X_{k-1} + \Delta S cos(\Theta_{k-1} + \frac{\Delta\Theta}{2}) \tag{2.7}$$

$$Y_k = Y_{k-1} + \Delta S sen(\Theta_{k-1} + \frac{\Delta\Theta}{2}) \tag{2.8}$$

$$\Theta_k = \Theta_{k-1} + \Delta\Theta \tag{2.9}$$

Donde:

X_k	Posición actual en el eje de las abscisas del plano que se quiere estimar
X_{k-1}	Posición anterior conocida en el eje de las abscisas del plano
Y_k	Posición actual en el eje de las ordenadas del plano que se quiere estimar
Y_{k-1}	Posición anterior conocida en el eje de las ordenadas del plano
Θ_k	Ángulo actual de orientación de la estructura respecto al eje de las abscisas del plano que se quiere estimar
Θ_{k-1}	Ángulo anterior conocido de orientación de la estructura respecto al eje de las abscisas del plano

Hay que destacar que esta integración está sujeta a un error que crece con el tiempo y se hace significativo cuando las distancias recorridas son grandes. Las causas de este error son varias:

- Deslizamiento de las ruedas.
- Inexactitud en la calibración de los parámetros cinemáticos (por ejemplo, radio de las ruedas y distancia entre sus ejes).
- Error numérico del método.

Este método de navegación relativo se suele complementar con otros métodos absolutos, que permiten corregir la estimación del estado del robot. En el epígrafe 1.5.1 se hace referencia a algunos métodos absolutos de localización.

2.3.4. Sistema de monitorización y control remoto

Se desarrolló un software para realizar el control y supervisión de las tareas programadas y comprobar su efectividad.

Entre las características principales de esta aplicación se encuentran:

- Visión remota.
- Navegación en mapas.
- Control remoto.
- Supervisión del estado general del robot y sus sensores.
- Posibilidad de ejecución de algoritmos de mayor costo computacional utilizando capacidades externas.

A continuación se describe cada módulo implementado y se explica su funcionalidad.

Servidor de Control

A diferencia de los restantes módulos, este se ejecuta en el robot que se desea supervisar, lo que posibilita que el microprocesador del mismo no tenga necesariamente que computar los algoritmos de mayor costo.

Este servidor emplea un protocolo diseñado específicamente para la aplicación. Es fácilmente extensible para añadir funcionalidades y comprende comandos específicos para cada una de las funciones que el robot tiene programadas. Para su implementación se empleó el protocolo UDP sobre una red IP.

Visión remota

El software cuenta con la posibilidad de mostrar las imágenes tomadas por una cámara a bordo del robot a través de una red inalámbrica. El procesamiento de las mismas se realiza desde una computadora personal, cuyas prestaciones son generalmente superiores a las de un robot móvil. Este módulo facilita el control remoto a grandes distancias. La figura 2.16 muestra una sección de una captura de la aplicación donde se puede observar una imagen tomada de la cámara a bordo del robot.

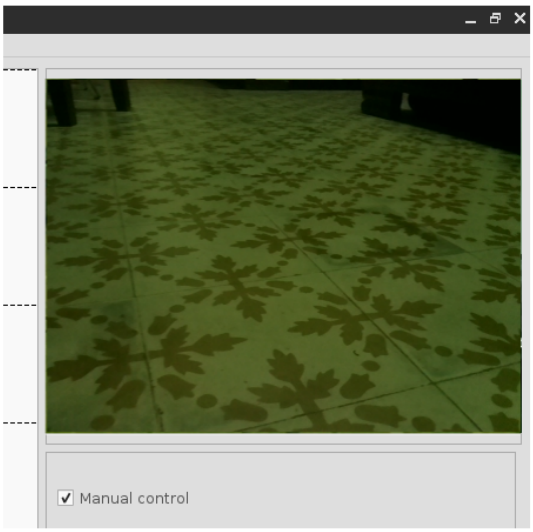

Figura 2.16: Captura de video remota.

Control remoto a lazo abierto

La aplicación también provee un soporte para control manual, empleando las teclas WASD de un teclado QWERTY. El algoritmo empleado es también compatible para el empleo de un *joystick*.

Navegación en mapas

Para modelar un mapa se definió un tipo de archivo basado en formato JSON en donde se describen las dimensiones de una o varias zonas. La aplicación construye un gráfico con estas descripciones y lo proyecta en un

visor. En la figura 2.17 se observa una captura de la aplicación donde se muestra un ejemplo de mapa y la localización del robot.

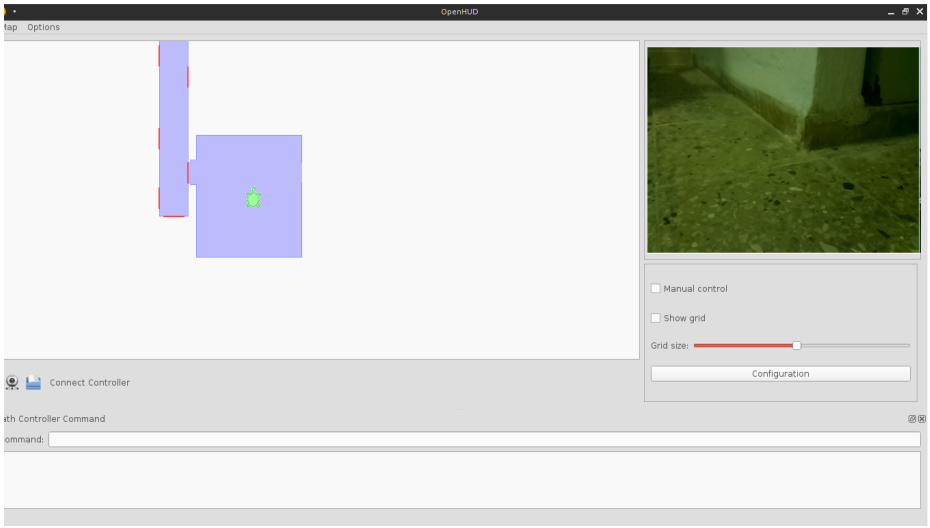

Figura 2.17: Localización y navegación en el mapa de una habitación.

Para realizar la navegación, se puede controlar el movimiento de la plataforma robótica desde la aplicación de forma manual o a través del controlador de trayectoria. El servidor envía entonces las actualizaciones de la posición del robot, y la aplicación se encarga de cambiar la ubicación del mismo en el visor.

Control de trayectorias

El programa brinda la posibilidad, una vez conectado el robot al servidor, de acceder a todas las funcionalidades del controlador de trayectorias. Usando el protocolo descrito al inicio de este epígrafe, el usuario de esta aplicación puede definir las tareas que debe realizar el controlador. Estas pueden ser el movimiento directo a una posición en el terreno o una serie de puntos definiendo una ruta.

Lectura de sensores

La lectura de los sensores permite al robot conocer la situación de su entorno. A este software se pudieran acoplar sensores como brújulas, acelerómetros y giroscopios. Las lecturas de los mismos se pueden consultar con

la previa implementación de una interfaz con esta finalidad. De esta forma se asegura la inclusión de muchos tipos de sensores con diversas finalidades.

2.4. Análisis de costo

A continuación se analiza el costo de los materiales y medios empleados para la fabricación del prototipo. La tabla 2.2 muestra los costos de cada uno de los componentes empleados en el diseño.

Componente	**Precio (USD)**
Raspberry-pi	35.00
Arduino Uno	20.00
Power Bank	20.50
Adaptador WiFi	11.00
Batería 12V	20.00
Memoria SD	8.00
Motores y Ruedas	74.00
Cámaras	30.00
Otros	15.00
Total	**233.50**

Tabla 2.2: Costo de los medios y materiales empleados.

Nota: Otros, se refiere a los condensadores, resistencias, conectores y otros componentes utilizados en la fabricación de la placa de circuito impreso.

Capítulo 3

Detección de obstáculos

En el campo de la robótica móvil autónoma se han desarrollado varias técnicas para estimar la posición y dimensiones de los obstáculos que puedan existir en el ambiente donde opera el robot. Dependiendo del propósito específico y el presupuesto disponible, existen en el mercado varios sensores para este fin. Entre ellos se encuentran, los radares de ondas milimétricas, los lásers, los sensores ultrasónicos y las cámaras. En la sección 1.6 se abordan de forma más amplia las características de cada sensor y algunos detalles de su funcionamiento.

3.1. Visión computacional

La visión computacional surgió a raíz del desarrollo del procesamiento digital de imágenes. Su campo comprende el diseño y programación de algoritmos que permitan a las computadoras extraer información de las imágenes [19].

Este capítulo se centra en el análisis de varios algoritmos de visión computacional, que permiten identificar los obstáculos que puedan existir en capturas tomadas del ambiente donde opera un robot, usando una sola cámara. Con este propósito investigadores del tema han empleado técnicas basadas en filtrado por histogramas de apariencia, flujo óptico y procesos estocásticos como las cadenas de Markov y campos ocultos de Markov para desarrollar sus técnicas de detección.

3.2. Imágenes en la computadora

En este epígrafe se explican conceptos relacionados con el almacenamiento y el procesamiento de imágenes digitales.

3.2.1. Espacios de colores

Son modelos matemáticos abstractos que describen posibles representaciones de colores mediante vectores. Un espacio de color puede ser descrito de forma eficiente a partir de combinaciones lineales de vectores linealmente independientes pertenecientes a dicho espacio. Existen varios espacios de colores empleados con finalidades técnicas. A continuación se analizan algunos casos particulares.

RGB

Se emplea usualmente para definir los colores mostrados en el monitor de una computadora. Cada color queda descrito en función de sus componentes de Rojo, Verde y Azul.

La figura 3.1 muestra una representación del espacio de color RGB donde se puede observar como cada componente se asocia a un eje de coordenadas y según las combinaciones de estos se obtienen el resto de los colores.

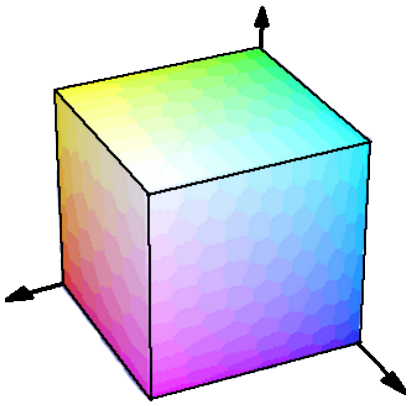

Figura 3.1: Representación gráfica del espacio de color RGB.

HSV

A diferencia de RGB, este espacio de color se representa en coordenadas cilíndricas como muestra la figura 3.2. Las componentes que determinan este espacio de color son:

Matiz: Describe un color de referencia asociado a una longitud de onda. Esta componente contiene la información de croma en esta representación.

Saturación: Representa cuán alejado está el color del blanco de referencia.

Valor: Contiene la información del brillo de la imagen.

El nombre del espacio se debe a las iniciales de estos componentes en el idioma inglés: *Hue*, *Saturation* y *Value* respectivamente.

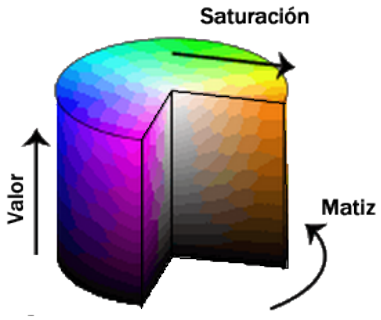

Figura 3.2: Representación gráfica del espacio de color HSV.

Esta representación es muy empleada por artistas debido a que resulta mucho más natural pensar en los colores en términos de su matiz y su brillo, que en términos de componentes de colores primarios. HSV es una transformación del espacio de color RGB, sus componentes y colorimetría son relativas a dicho espacio.

HSL

Es una representación bastante similar a HSV como se puede observar en la figura 3.3. Aunque la definición de Matiz es exactamente la misma, el

significado de la Saturación cambia considerablemente y se introduce como tercera componente la Iluminación para este espacio de color. La principal diferencia entre ambas representaciones es que el brillo de un color puro es igual al brillo del blanco, mientras que la iluminación de un color puro es igual a la iluminación de un gris.

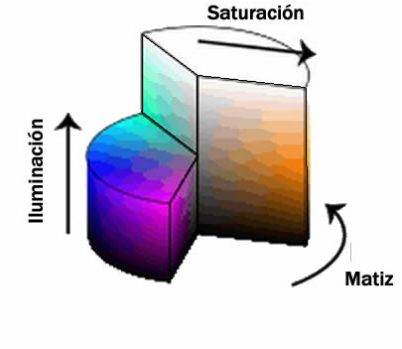

Figura 3.3: Representación gráfica del espacio de color HSL.

3.2.2. Imágenes digitales

Las imágenes almacenadas en una computadora se describen mediante una cantidad finita de elementos llamados píxeles. Los píxeles se disponen en filas y columnas de tamaños fijos. Cada pixel contiene información de un color y puede estar expresada en cualquier espacio de color.

De forma general una imagen puede ser descrita mediante una matriz de vectores de la forma:

$$\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1j} \\ p_{21} & p_{22} & \cdots & p_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ p_{i1} & p_{i2} & \cdots & p_{ij} \end{bmatrix}$$

Donde cada elemento p_{ij} representa un vector con la información de color de un pixel descrita mediante los componentes del espacio de color utilizado.

La figura 3.4 muestra la imagen correspondiente a la matriz:

$$\begin{bmatrix} (255,255,255) & (255,255,204) & (204,0,102) & (255,0,0) \\ (255,204,255) & (153,153,153) & (0,255,0) & (255,255,0) \\ (0,255,255) & (153,0,51) & (51,51,51) & (51,0,0) \\ (0,0,255) & (0,102,255) & (153,0,0) & (0,0,0) \end{bmatrix}$$

Donde el color de cada pixel está descrito por un vector del espacio RGB.

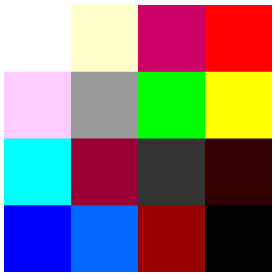

Figura 3.4: Imagen de ejemplo.

3.3. Algoritmos de detección de Obstáculos

3.3.1. Detección basada en flujo óptico

El flujo óptico es el patrón del movimiento aparente de los objetos, superficies y bordes en una escena causado por el movimiento relativo entre un observador (un ojo o una cámara) y la escena. Existen múltiples variantes de detección de obstáculos usando flujo óptico.

De forma general las técnicas de detección usando flujo óptico se basan en comparar dos imágenes tomadas de forma consecutiva de una misma escena, y analizar patrones que puedan brindar información útil sobre los obstáculos. La efectividad de la detección está influida en gran medida por la frecuencia de muestreo de los cuadros. La figura 3.5 muestra dos imágenes tomadas consecutivamente por una plataforma robótica móvil.

De manera general la detección de obstáculos empleando flujo óptico, puede ser descrita de forma breve en 3 pasos:

1. Localizar la mayor cantidad de píxeles posibles de la imagen anterior en la nueva imagen.

Figura 3.5: Imágenes tomadas de forma continua por una plataforma robótica móvil.

2. Crear una matriz de vectores a partir de la diferencia en la posición de cada píxel.

3. Seleccionar bajo algún criterio los vectores que describen obstáculos.

Extracción de la matriz de flujo óptico

Para localizar secciones de una imagen en otra, existen varios algoritmos que brindan buenos resultados, aunque de manera general poseen un costo computacional considerable. Los detalles en la implementación de estas técnicas no serán abordados en este trabajo. Mediante el uso de bibliotecas especializadas en visión por computadora, como *OpenCV*, se puede obtener la información de flujo óptico entre dos imágenes. En la figura 3.6 se puede observar una muestra de los vectores de flujo óptico extraídos de las imágenes mostradas en la figura 3.5.

Selección de vectores

Una vez obtenidos los vectores que indican el desplazamiento de cada píxel respecto a la imagen anterior, existen diversas variantes de técnicas que se pueden emplear para extraer información útil referente al ambiente.

En este caso se empleó un criterio muy simple, basado en la idea intuitiva de que si el observador se encuentra en movimiento y el ambiente se mantiene invariable, la velocidad con que se desplazan los píxeles correspondientes a obstáculos es diferente a la velocidad correspondiente a los píxeles del suelo. Por esta razón, estableciendo un umbral en la norma de los vectores se puede

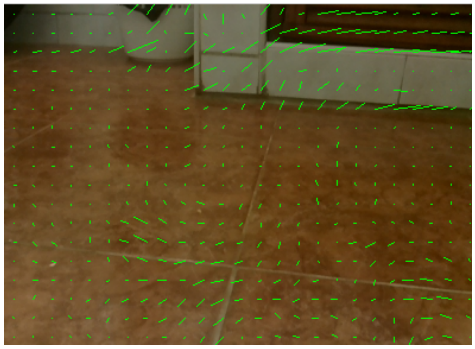

Figura 3.6: Representación de los vectores de flujo óptico.

filtrar según este criterio y obtener una aproximación de las regiones de la imagen que pueden ser consideradas regiones de suelo o de obstáculos.

En la figura 3.7 se pueden observar las áreas correspondientes a los vectores de flujo óptico de mayor norma encontrados en el análisis de las imágenes mostradas en la figura 3.5.

Figura 3.7: Detección de obstáculos mediante análisis de flujo óptico.

Los métodos basados en flujo óptico se han empleado en detección de obstáculos para robots móviles, pero de forma general se obtienen mejores resultados cuando el observador mantiene su posición.

3.3.2. Detección basada en apariencia

Este algoritmo fue presentado en el año 2000 por Iwan Ulrich [23], y se diseñó para la detección de obstáculos basándose en su apariencia, y en la

diferencia entre estos y el resto de la imagen.

El método asume tres condiciones que se cumplen para una variedad considerable de entornos, tanto en interiores como en exteriores:

- Los obstáculos difieren en su apariencia con el suelo.
- El suelo es relativamente plano.
- No hay obstáculos suspendidos en el aire.

De forma simplificada puede ser descrito mediante cuatro pasos:

1. Filtrar las componentes de alta frecuencia en la imagen capturada.
2. Transformación al espacio de color HSL.
3. Realización de un histograma a un área de referencia.
4. Comparación de cada píxel de la imagen con el histograma de referencia.

Aplicación de un filtro gaussiano

Filtrando las componentes de altas frecuencias, se disminuye el ruido en la imagen, lo que permite centrar el análisis en las regiones más amplias, obviando detalles pequeños que pudieran introducir errores. Este es un procedimiento muy común en los algoritmos de este tipo.

En este caso se emplea un filtro gaussiano, cuya respuesta al impulso se expresa mediante:

$$g(x,y) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Para eliminar las componentes de altas frecuencias, se efectúa un producto de convolución entre la imagen y la función de respuesta al impulso que define el filtro gaussiano, evaluada en los valores discretos equivalentes a las dimensiones de la imagen.

$$I_f(i,j) = I(i,j) * g(i,j) \tag{3.1}$$

Donde:

$I_f(i,j)$	Imagen filtrada
$I(i,j)$	Imagen original
$g(i,j)$	Respuesta al impulso del filtro

Nótese que la frecuencia de corte del filtro está determinada por la desviación estándar σ de la función gaussiana empleada.

El resultado de aplicar esta operación puede ser comprobado en la figura 3.8, donde se muestra a la izquierda la imagen tomada de la cámara a bordo de la plataforma robótica diseñada, y a la derecha, la misma imagen luego de aplicarle el procesamiento descrito.

Figura 3.8: Resultado de filtrar componentes de altas frecuencias.

Transformación de espacio de color

El espacio de color HSV, brinda facilidades para encontrar similitudes entre píxeles de una imagen. Esto se debe a la forma en que se describen los colores en este espacio. Debido a que la información del color es muy ruidosa para bajos valores de intensidad, solo se tendrán en cuentan los componentes de *hue* y *saturation* si el componente *value* asociado está sobre un valor mínimo. De igual forma, para bajos valores de *saturation* el componente *hue* no posee información alguna, por eso solo serán tenidos en cuenta los valores de *hue* cuya *saturation* asociada exceda un valor mínimo.

Selección de un área de referencia

Para la selección de un área de referencia, que se asume sea un área libre de obstáculos, se selecciona un trapecio en la zona inferior central de la imagen, (como se muestra en la figura 3.9). Se seleccionó esta región, porque de forma estadística se comprobó que es la sección donde existe mayor probabilidad de no encontrar obstáculos [23]. Las dimensiones del trapecio que delimita la región tomada como referencia, son parámetros del

algoritmo que pueden ser variados en dependencia del ambiente donde debe operar el robot.

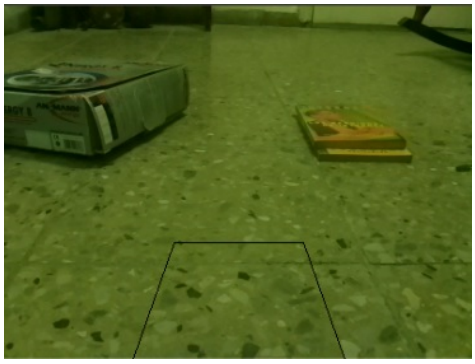

Figura 3.9: Área de referencia para el análisis del suelo.

Histograma del área de referencia

Una vez delimitada la región de referencia, se realiza un histograma para los valores de *hue* y *value* que cumplan con los requerimientos explicados anteriormente. A continuación se combinan estos histogramas mediante la aplicación de un promedio. Las ventajas fundamentales del empleo de histogramas para esta aplicación son los bajos requerimientos en cuanto a la cantidad de memoria y el tiempo de cómputo necesario para su creación. La figura 3.10 muestra el histograma de la región señalada en la figura 3.9.

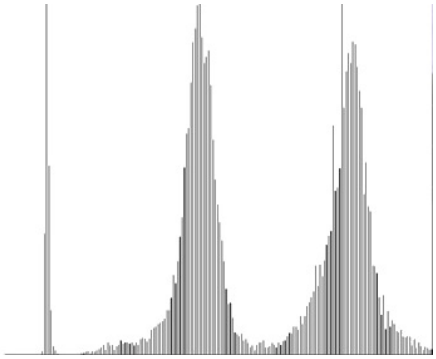

Figura 3.10: Histograma de la región de referencia.

Análisis de toda la imagen

Para la determinación de los obstáculos en la escena, se procede a analizar cada elemento de la matriz que describe la foto original, una vez aplicado el filtro gaussiano y habiendo hecho la transformación de espacio de color. Un píxel es clasificado como un obstáculo si al menos una de las siguientes condiciones se satisface:

- El valor del histograma correspondiente a la componente *hue* del píxel se encuentra por debajo de un umbral definido.
- El valor del histograma correspondiente a la componente *value* del píxel se encuentra por debajo de un umbral definido.

En la figura 3.11, a la izquierda se puede observar un ejemplo de una imagen tomada por un robot y la derecha la misma imagen con los obstáculos detectados empleando el método explicado.

Figura 3.11: Detección de obstáculos mediante filtro de histogramas.

3.4. Optimización de parámetros

Como se pudo comprobar con los algoritmos explicados anteriormente, existen varios parámetros que pueden ser ajustados en cada uno de ellos. A continuación se propone un método para la optimización de estos parámetros, que resulta independiente del algoritmo que se emplee para la detección de los obstáculos.

El método diseñado puede ser descrito en 4 pasos:

1. Capturar una serie de imágenes de prueba con la que serán evaluados los algoritmos de detección.

2. Indicar los obstáculos en cada imagen de forma manual, usando un programa de edición de imágenes.

3. Comparar el resultado del algoritmo que se quiere optimizar con la imagen detectada por un humano.

4. Encontrar de forma automática y en un tiempo razonable, los valores de los parámetros que hacen que la detección realizada por el algoritmo que se está probando sea lo más similar posible a la realizada por un humano.

3.4.1. Formulación del problema

Como en todo problema de optimización, se debe modelar el problema mediante una función objetivo. Dicha función puede ser procesada o evaluada varias veces para obtener su extremos.

Función objetivo

La implementación de este método se basó en encontrar un mínimo en la función del error entre las imágenes correspondientes a los obstáculos detectados por el algoritmo y por un humano. La figura 3.12 muestra un esquema general de como se obtiene el valor de calidad para un conjunto de n parámetros. Esta función varía su valor de calidad en dependencia de los parámetros que sean fijados para el algoritmo, de está forma, el problema se reduce a encontrar los parámetros correspondientes al mejor valor de calidad.

En el bloque comparador se realiza una comparación píxel a píxel entre las dos imágenes con los obstáculos detectados. En caso que ambas imágenes tengan el mismo valor del píxel, se procede a analizar el siguiente, de lo contrario se incrementa primero un contador de píxeles erróneos. Al finalizar el procedimiento en la totalidad de la imagen se divide la cantidad de píxeles diferentes entre la cantidad total de la imagen original, lo que pudiera ser interpretado como un coeficiente de error cuyo valor está acotado entre 0 y 1. Minimizando el coeficiente de error se obtiene una detección de obstáculos con mayor calidad.

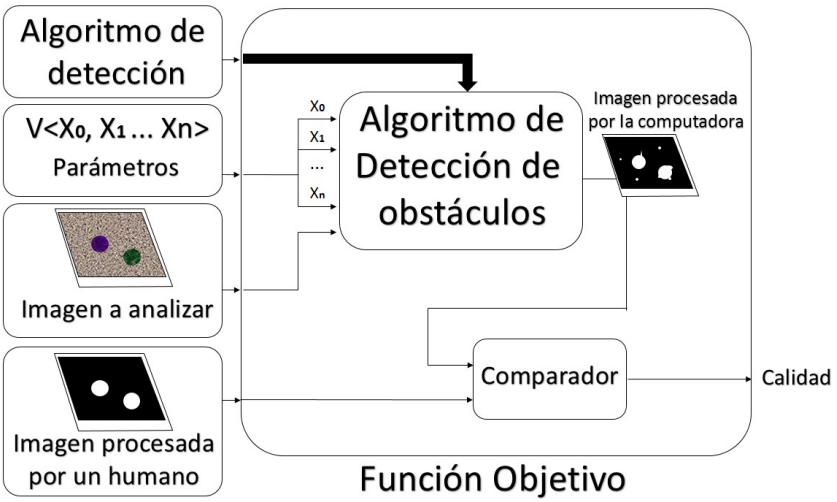

Figura 3.12: Función objetivo para la optimización.

3.4.2. Solución mediante métodos aproximados

Si se contara con la expresión analítica de la función de error, y las primeras derivadas parciales de esta fueran continuas, el problema se reduciría a realizar un diferencial y encontrar los ceros. Desafortunadamente esto no es posible en ningún caso, y el único recurso disponible es la evaluación de la función, que dados los requerimientos de cómputo de estos algoritmos, no resulta un proceso poco costoso, y se debe tratar de minimizar la cantidad de evaluaciones que se realicen en la función.

Se propone utilizar algoritmos metaheurísticos para encontrar los parámetros óptimos en la detección de obstáculos dado un ambiente determinado. En general constituyen una buena alternativa para solucionar problemas donde la función objetivo es multimodal, y una búsqueda exhaustiva no resulta factible debido al alto costo de la función objetivo [20].

Una metaheurística es un método heurístico para resolver un tipo de problema computacional general, usando los parámetros dados por el usuario sobre unos procedimientos genéricos y abstractos de una manera que se espera eficiente. Usando una metaheurística diseñada para optimizar espacios continuos es posible recorrer de forma eficiente el dominio de la función, determinado por la cantidad de parámetros del algoritmo que se quieran optimizar, en búsqueda de óptimos, obteniendo resultados aproximados en

tiempos razonables.

Existen diversos algoritmos metaheurísticos que se basan en ideas muy diferentes, pero de forma general, una metaheurística puede ser vista como un algoritmo de caja negra, que recibe como parámetros para su funcionamiento:

- Función que se desea optimizar
- Número de dimensiones de la función
- Dominio de la función
- Cantidad de evaluaciones posibles

Y como resultado devuelve la mejor combinación de variables encontradas que corresponden al menor valor de la función, obtenido durante las evaluaciones.

Búsqueda de Población Mínima

Es una metaheurística diseñada para optimizar problemas continuos multimodales con funciones objetivo costosas. La idea de MPS consiste en proveer fuertes mecanismos de exploración y diversificación que le permiten cubrir de manera efectiva el espacio de búsqueda, manteniendo un tamaño de población relativamente pequeño. En Búsqueda de Población Mínima (o MPS, del inglés Minimum Population Search) [6], el tamaño de la población recomendado es la dimensionalidad del problema ($n = d$), y se utiliza un mecanismo secundario para escapar el hiperplano $(n-1)$-dimensional definido por los n miembros de la población. Para generar nuevas soluciones, se utilizan segmentos de línea para buscar en dicho hiperplano, y un posterior paso ortogonal al mismo posibilita abarcar todas las dimensiones del espacio de búsqueda.

MPS ha demostrado ser una técnica efectiva para solucionar problemas de optimización continuos multimodales, razón por la cual se decidió utilizarla en este trabajo.

3.4.3. Implementación de la optimización

Usando los recursos anteriormente expuestos se desarrolló un software que utiliza:

1. Imágenes de prueba tomadas en el ambiente donde debe operar el robot.

2. Las imágenes anteriores con los obstáculos señalados por un humano.
3. Un algoritmo de detección de obstáculos.
4. Los parámetros del algoritmo que se desean optimizar, y el rango de valores que pueden tomar.
5. Una metaheurística para espacios continuos.

Dados estos elementos el software permite encontrar los valores de los parámetros del algoritmo que proporcionan mejores resultados en la detección de obstáculos, para las imágenes de prueba dadas.

Capítulo 4

Resultados experimentales

Este capítulo explica el marco de experimentación desarrollado para comprobar los resultados de la investigación. Se detallan las características de los experimentos realizados, los objetivos perseguidos en cada caso y se exponen los resultados de los mismos.

4.1. Determinación del tiempo de establecimiento en el control de la velocidad de los motores

El controlador de velocidad empleado se explica en el epígrafe 2.3.1. El control de velocidad de los motores posee tres parámetros que han de ser ajustados de forma tal que minimicen el tiempo de establecimiento del sistema.

En la figura 4.1 se puede observar un gráfico de velocidad angular en función del tiempo. A partir del instante t_0, señalado en el borde inferior, se modificó la velocidad deseada en el motor y se puede observar como el sistema tardó 323 milisegundos en estabilizarse a la nueva velocidad.

La tabla 4.1 muestra otros valores de tiempo de establecimiento para diferentes cambios de velocidad angular.

Analizada esta información se puede concluir que el tiempo establecimiento promedio del controlador es de 341.25 milisegundos.

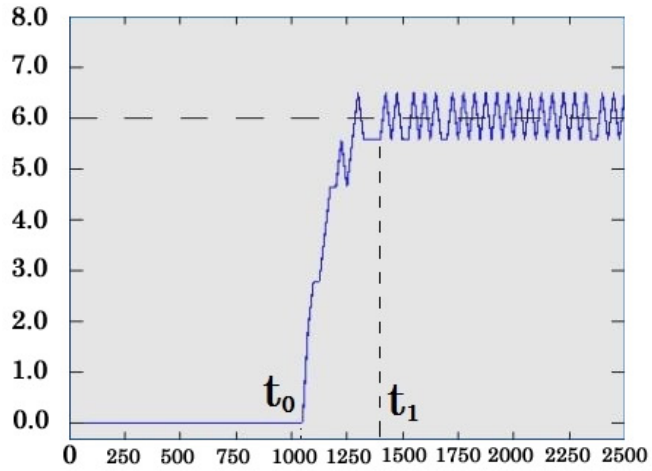

Figura 4.1: Gráfico de estabilización del controlador PID

Velocidad inicial (rad/s)	**Velocidad final** (rad/s)	**Tiempo de establecimiento** (ms)
0	3	103
0	-3	101
0	7.6	251
0	-7.6	256
7.6	0	714
-7.6	0	398
0	6	323
0	-7	405
6	7.5	160
-7	-4	315
7.5	0	697
-4	0	372

Tabla 4.1: Tiempos de establecimiento del controlador PID.

4.2. Evaluación de la precisión en el sistema de localización basado en Odometría

Para el sistema de localización se empleó la técnica odométrica descrita en el epígrafe 2.3.3. Como se puede observar, la precisión de esta técnica depende solamente de la precisión con que se pueda medir el ángulo de rotación de cada rueda, la distancia entre las ruedas y el radio de las mismas.

La precisión en la medición del ángulo está determinada por la resolución de los *encoders* empleados. Por tanto, para obtener los mejores resultados se deben estimar con la mejor precisión posible los valores de radio y distancia entre las ruedas. Para ello, se realizaron dos experimentos que permiten el ajuste de estos parámetros.

El ajuste del radio puede realizarse moviendo al robot en línea recta una distancia fija, varias veces. Se registran los valores reales del movimiento, se promedian y se puede obtener el error en el radio de las ruedas.

Para ajustar la distancia entre las ruedas se realiza un experimento basado en un principio similar. El robot deberá rotar sobre su eje central una cantidad fija de vueltas, mientras se registran periódicamente el ángulo de pose obtenido por la Odometría y el ángulo medido con una brújula. Como los errores característicos de estos métodos son diferentes, si se analiza el desfasaje promedio luego de varias vueltas, se puede obtener el error en la distancia entre las ruedas.

La figura 4.2 muestra una gráfica con los valores obtenidos de ángulo de pose mediante odometría (rojo) y brújula (azul) tomados periódicamente, cada 100ms, durante el experimento de rotación descrito anteriormente.

La figura 4.3 muestra una gráfica con la diferencia entre los valores obtenidos de ángulo de pose mediante odometría y brújula mostrados anteriormente. En esta figura se pueden comprobar las siguientes observaciones:

- Existe un error periódico que coincide con el error característico de la brújula.
- A medida que aumenta la cantidad de vueltas el error promedio aumenta.

De la segunda observación se puede concluir que el valor del parámetro distancia entre las ruedas posee un error, en este caso por exceso, que ha de ser corregido. Para su corrección se pueden emplear métodos numéricos de estimación de mínimos realizando el experimento descrito varias veces hasta encontrar el valor del parámetro que provea un error promedio aceptable.

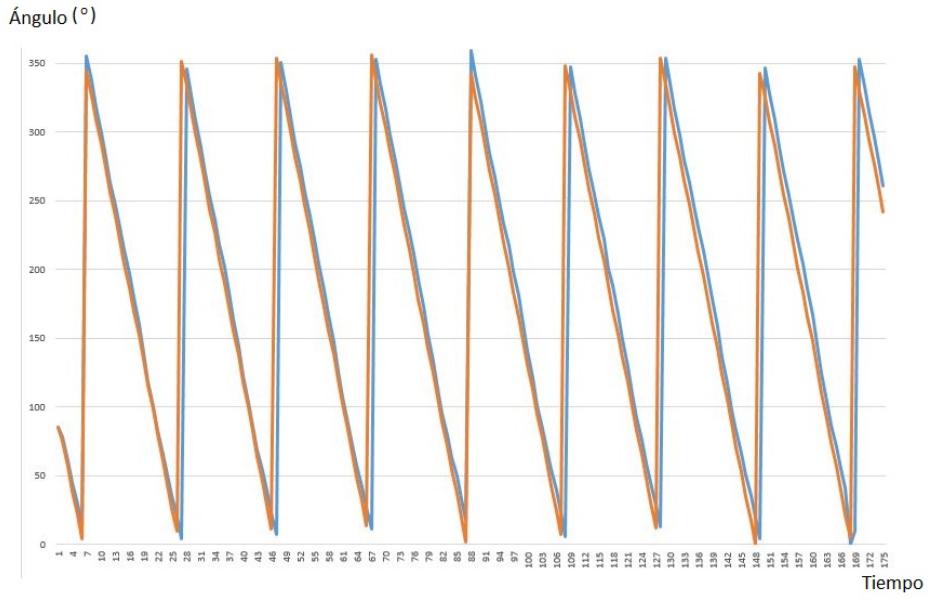

Figura 4.2: Ángulo de pose obtenido mediante odometría y brújula

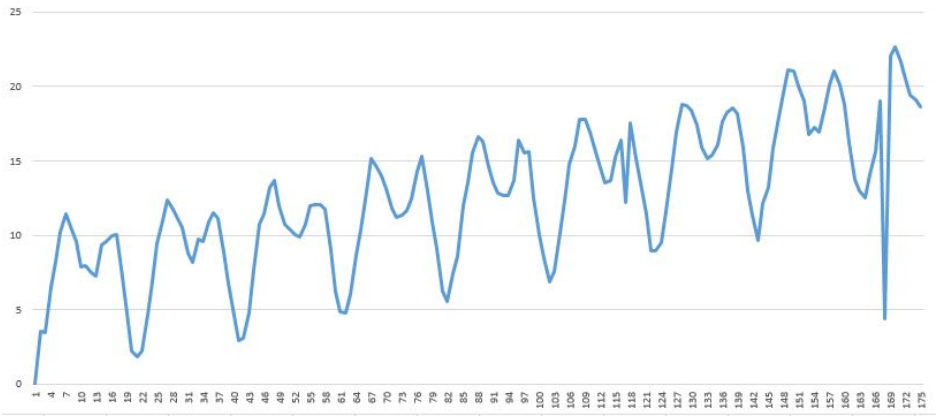

Figura 4.3: Error entre los ángulos de pose obtenidos mediante odometría y brújula

4.3. Análisis del rango de cobertura de las comunicaciones inalámbricas

Como se explicó en el epígrafe 2.2.3, el sistema de comunicaciones empleado se basó en la tecnología WiFi. Para facilitar el acceso al robot, se implementó un punto de acceso inalámbrico en la computadora a bordo. De esta forma no es necesario configurar al robot para conectarse a una red específica.

La tabla 4.2 muestra la potencia recibida para diferentes valores de distancia y diferentes ángulos. Los ángulos referidos anteriormente, denotados por θ_0, θ_1, θ_2 y θ_3 se muestran en la figura 4.4. Las mediciones fueron realizadas con un terminal móvil Android del fabricante HTC, modelo Desire X.

Distancia (m)	**Potencia**(θ_0) (dbm)	**Potencia**(θ_1) (dbm)	**Potencia**(θ_2) (dbm)	**Potencia**(θ_3) (dbm)
1	-42	-42	-41	-48
2	-52	-47	-43	-51
3	-52	-49	-48	-58
5	-56	-56	-50	-62
10	-61	-63	-59	-72
15	-71	-73	-68	-77
20	-75	-78	-69	-78
30	-78	-80	-78	-81
50	-83	-83	-80	-86

Tabla 4.2: Mediciones de intensidad de la señal.

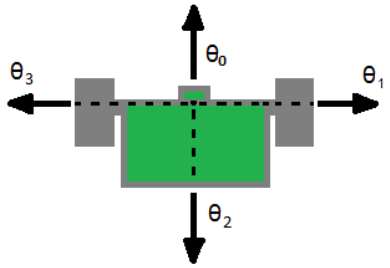

Figura 4.4: Ángulos analizados en la medición de potencia

4.4. Detección de obstáculos usando flujo óptico

En el epígrafe 3.3.1 se explicó el funcionamiento de este método de detección de obstáculos. A continuación se muestra como influye la variación de parámetros en una misma captura empleando como criterio de calidad, la función objetivo descrita en el epígrafe 3.4.1, se muestran varios ejemplos de la aplicación de este algoritmo a capturas realizadas por el robot y en cada caso se da a conocer el valor de calidad asociado.

4.4.1. Optimización de parámetros

Los parámetros fundamentales de este algoritmo son la frecuencia de adquisición de las imágenes y el criterio de selección de los vectores. En este caso los vectores son seleccionados según su norma. La figura 4.5 muestra una misma captura, procesada usando este algoritmo con diferentes parámetros. A la izquierda de cada imagen se puede observar el por ciento de efectividad del método calculado usando el coeficiente de error expuesto en el epígrafe 3.4.1.

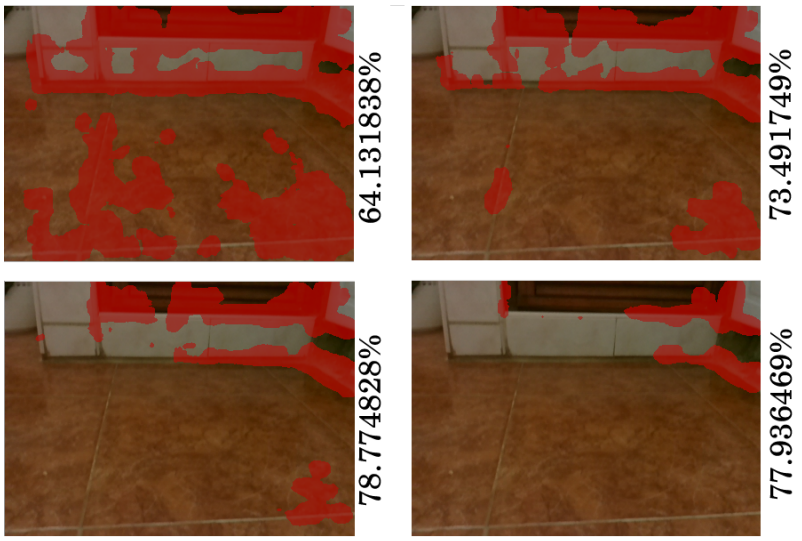

Figura 4.5: Impacto de los parámetros en la calidad de la detección mediante flujo óptico

4.4.2. Aplicación del algoritmo en varios ambientes

Para comprobar la efectividad de este método se realizaron pruebas en ambientes considerablemente diferentes. La figura 4.8 muestra en la columna de la izquierda varias capturas realizadas por el robot y a la derecha el resultado de la detección realizada con este método.

4.5. Detección de obstáculos basada en apariencia

En el epígrafe 3.3.2 se explicó el funcionamiento de este método de detección de obstáculos. A continuación se muestra como influye la variación de parámetros en una misma captura empleando como criterio de calidad, la función objetivo descrita en el epígrafe 3.4.1, se muestran varios ejemplos de la aplicación de este algoritmo a capturas realizadas por el robot y en cada caso se da a conocer el valor de calidad asociado.

4.5.1. Optimización de parámetros

En el caso de este algoritmo, existen varios parámetros a ajustar. Según pruebas realizadas los de mayor impacto en el resultado de la detección, son las dimensiones de la región de referencia. La figura 4.7 muestra una misma captura, procesada usando este algoritmo con diferentes parámetros. La columna de la izquierda muestra la imagen original con la región de referencia señalada, la columna central muestra el resultado de la detección y a la derecha se puede observar el por ciento de efectividad del método calculado usando el coeficiente de error expuesto en el epígrafe 3.4.1.

4.5.2. Aplicación del algoritmo en varios ambientes

Para comprobar la efectividad de este método se realizaron pruebas en ambientes considerablemente diferentes. La figura 4.8 muestra en la columna de la izquierda varias capturas realizadas por el robot y a la derecha el resultado de la detección realizada con este método.

Figura 4.6: Detección mediante flujo óptico en diferentes ambientes

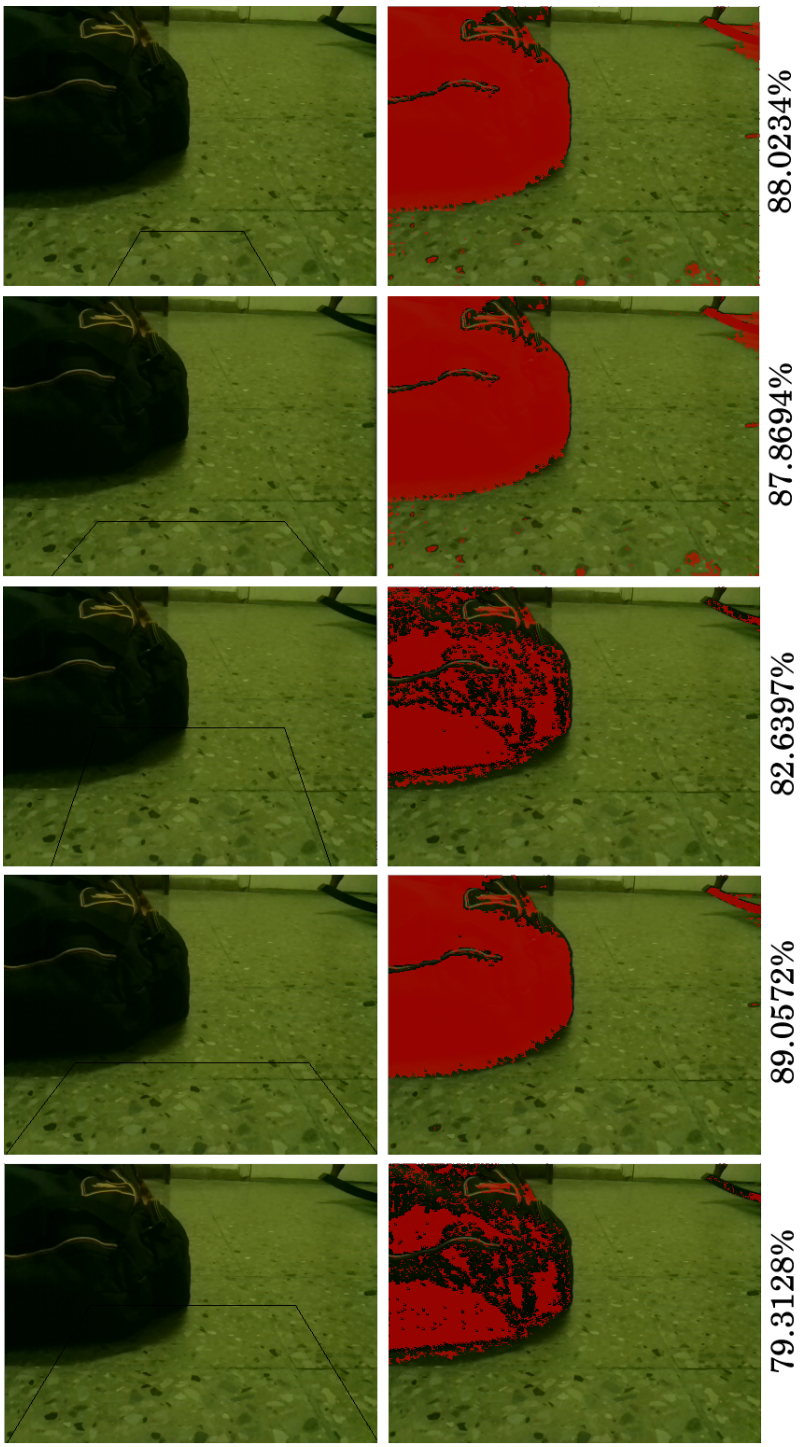

Figura 4.7: Impacto de los parámetros en la calidad de la detección basada en apariencia

Figura 4.8: Detección basada en apariencia en diferentes ambientes

Conclusiones

Se diseñó una plataforma robótica móvil con las capacidades necesarias para ejecutar tareas que requieran de procesamiento digital de imágenes.

Durante el desarrollo del proyecto se realizó una búsqueda de información actualizada sobre la robótica móvil: la estructura mecánica, la localización de los robots móviles, las tecnologías de comunicación inalámbrica y los sistemas de detección de obstáculos, lo que permitió seleccionar adecuadamente los componentes a integrar en la plataforma robótica.

Se desarrolló un robot móvil con ruedas de accionamiento diferencial. Este robot es capaz de planificar trayectorias, realizar el control de estas y puede estimar la posición a través de navegación por odometría. El trabajo realizado puede servir de base para investigaciones futuras sobre temas de robótica móvil.

Se implementó un sistema de comunicación inalámbrica con tecnología WiFi. A través de este se pudo realizar el intercambio de datos en tiempo real entre el robot y una PC, y otros terminales que implementan dicho estándar.

Se integraron los sistemas de control de velocidad y trayectorias desarrollados en investigaciones precedentes.

Se programó un software que permite el control y la supervisión remota en tiempo real de la plataforma.

Se incluyó en el diseño del robot una cámara a bordo con las características adecuadas para el estudio de algoritmos de visión computacional orientados a la robótica.

Se implementaron dos algoritmos de detección de obstáculos, con principios diferentes, usando la cámara a bordo del robot.

Se propuso un método de optimización de parámetros para algoritmos de detección de obstáculos.

Mediante una serie de pruebas se pudo comprobar el correcto funcionamiento de la plataforma móvil experimental.

El proyecto realizado cumple con todos los objetivos propuestos.

Recomendaciones

Entre las principales recomendaciones para futuros trabajos se encuentran:

- Incluir sensores ultrasónicos para obtener información redundante de la distancia de los obstáculos
- Añadir un sistema de iluminación frontal para garantizar una mejor adquisición de las imágenes
- Mejorar la capacidad de cómputo de la plataforma mediante el empleo de una SBC más potente como Raspberry Pi 2
- Comprobar resultados de otros algoritmos de detección de obstáculos
- Realizar una implementación usando aceleración por hardware de los algoritmos analizados

Glosario de términos

- A

  ARM (Advanced RISC Machine) Es una arquitectura RISC de 32 bits desarrollada por ARM Holdings.

- C

  CPU (Central Processing Unit) Unidad Central de Procesamiento. Es el componente principal del ordenador y otros dispositivos programables, que interpreta las instrucciones contenidas en los programas y procesa los datos.

- G

  GPIO (General Purpose Input-Output) Pines de propósito general de un dispositivo electrónico digital.

  GPS (Global Position System) Sistema de Posicionamiento Global.

- H

  HDMI (High-Definition Multimedia Interface) Interfaz multimedia de alta definición. Permite la transmisión de audio y video digital a alta resolución

- I

  IEEE (Institute of Electrical and Electronics Engineers) Instituto de Ingenieros Eléctricos y Electrónicos. Asociación técnico-profesional mundial dedicada a la estandarización.

  IP (Internet Protocol) Protocolo de Internet. Protocolo de comunicación de datos digitales clasificado funcionalmente en la Capa de Red según el modelo internacional OSI.

- J

  JSON (JavaScript Object Notation) Notación de objetos de Javascript. Es un formato para codificar información de objetos derivados del lenguaje Javascript, aunque actualmente es ampliamente usado por múltiples lenguajes.

- L

  LCD (Liquid crystal display) Pantalla de cristal líquido. Es una pantalla delgada y plana formada por un número de píxeles en color o monocromos colocados delante de una fuente de luz o reflectora.

  LED (Light emisor diode) Diodo emisor de luz.

- M

  MPS (Minimum Population Search) Búsqueda de Población Mínima. Algoritmo metaheurístico diseñado para optimizar problemas continuos multimodales con funciones objetivo costosas.

- N

  NTSC (National Television System Committee) Comisión Nacional de Sistema de Televisión. Sistema de televisión analógico que se ha empleado en América del Norte, América Central, la mayor parte de América del Sur y Japón entre otros.

- O

  OLED (Organic Light-Emitting Diode) Diodo orgánico de emisión de luz. Diodo que se basa en una capa electroluminiscente formada por una película de componentes orgánicos que reaccionan, a una determinada estimulación eléctrica, generando y emitiendo luz por sí mismos.

- P

  PAL (Phase Alternating Line) Línea de fase alternada. Sistema de codificación utilizado en la transmisión de señales de televisión analógica en color en la mayor parte del mundo. Se utiliza en la mayoría de los países africanos, asiáticos y europeos, además de Australia y algunos países americanos.

PID (Proportional-Integrative-Derivative) Mecanismo de control por realimentación que calcula la desviación o error entre un valor medido y el valor que se quiere obtener, para aplicar una acción correctora que ajuste el proceso.

PWM (Pulse Width Modulation) Modulación por ancho de pulsos. Consiste en modificar el ciclo útil de trabajo de una señal periódica cuadrada.

- R

RAM (Random-access memory) Memoria de acceso aleatorio. se utiliza como memoria de trabajo para el sistema operativo, los programas y la mayoría del software. Es allí donde se cargan todas las instrucciones que ejecutan el procesador y otras unidades de cómputo.

RCA Conector eléctrico común en el mercado audiovisual. El nombre proviene del nombre de la antigua compañía de electrónica estadounidense Radio Corporation of America, que fue la que introdujo el diseño en 1940.

RISC (Reduced Instruction Set Computer) Arquitectura de CPU generalmente utilizado en microprocesadores o microcontroladores. Cuenta con instrucciones de tamaño fijo y presentadas en un reducido número de formatos y sólo las instrucciones de carga y almacenamiento acceden a la memoria de datos. Además estos procesadores suelen disponer de muchos registros de propósito general.

ROM Read-only memory. Memoria de solo lectura. Medio de almacenamiento utilizado en ordenadores y dispositivos electrónicos, que permite sólo la lectura de la información y no su escritura, independientemente de la presencia o no de una fuente de energía.

RPM Revoluciones por minuto. Unidad de frecuencia que se usa también para expresar velocidad angular. En este contexto, indica el número de rotaciones completas realizadas durante un minuto por un cuerpo que gira alrededor de un eje.

- S

SBC (Single Board Computer) Computadora completa en un solo circuito. El diseño se centra generalmente en un solo microprocesador con la RAM, puertos de E/S y todas las demás carac-

terísticas de una computadora funcional en una sola tarjeta de tamaño reducido.

SD (Secure Digital) Formato de tarjeta de memoria para dispositivos portátiles.

SLAM (Simultaneous Localization And Mapping) Localización Y Mapeado Simultáneos o también Localización y Modelado Simultáneos. Es una técnica usada por robots y vehículos autónomos para construir un mapa de un entorno desconocido en el que se encuentra, a la vez que estima su trayectoria al desplazarse dentro de este entorno.

SMD (Surface Mount Device) Método de construcción de dispositivos electrónicos basado en el montaje de los componentes sobre la superficie del circuito impreso.

- T

TCP (Transmission Control Protocol) Protocolo de Control de Transmisión. Se emplea para intercambio de datos, orientado a conexión clasificado funcionalmente en la Capa de Transporte según el modelo internacional OSI.

- U

UART (Universal Asynchronous Receiver-Transmitter) Interfaz electrónica estándar dedicada a la transmisión y recepción de datos utilizando comunicación serie asincrónica.

USB (Universal Serial Bus) Estándar industrial desarrollado a mediados de los años 1990 que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica a ordenadores, periféricos y dispositivos electrónicos.

- W

WiFi: Mecanismo de conexión de dispositivos electrónicos de forma inalámbrica. Su funcionamiento se rige por el estándar IEEE 802.11.

Bibliografía

[1] *NASA Mars Science Laboratoty*. `http://www.nasa.gov/mission_pages/msl/index.html`. Consultado: 2015-03-29. (Citado en la página XIV).

[2] *PR2 WillowGarage*. `https://www.willowgarage.com/pages/pr2/overview`. Consultado: 2015-03-29. (Citado en la página XIV).

[3] *Arduino Uno datasheet*. Informe técnico, 2012. (Citado en la página 12).

[4] Ada, Lady: *Introducing the Raspberry Pi 2 - Model B*. 2015. (Citado en la página 18).

[5] Bambino, Il: *Una Introducción a los Robots Móviles*. 2008. (Citado en las páginas 2, 4, 5 y 6).

[6] Bolufé-Röhler, Antonio y Stephen Chen: *Minimum Population Search– A Scalable Metaheuristic for Multi-Modal Problems*. (Citado en la página 66).

[7] Chatterjee, Amitava, Anjan Rakshit y N Nirmal Singh: *Vision Based Autonomous Robot Navigation*. 2013, ISBN 9783642339646. (Citado en las páginas 21 y 22).

[8] Domínguez, Abel: *Implementación de una plataforma para la robótica móvil autónoma*, 2014. (Citado en la página 47).

[9] Gray, Keith W y Kay Baker: *Obstacle detection and avoidance for an autonomus farm tractor*. Tesis de Doctorado, 2000. (Citado en las páginas 29, 30, 31 y 32).

[10] Herrador, Rafael Enríquez: *Guía de Usuario de Arduino*. 2009. (Citado en las páginas 12 y 13).

[11] Laumond, Jean paul: *Robot Motion Planning and Control*. 1998, ISBN 3540762191. (Citado en las páginas 22, 23, 27, 28 y 47).

[12] Lee, Jung suk, Chanki Kim y Wan Kyun Chung: *Robust RBPF-SLAM using Sonar Sensors in Non-Static Environments*. (Citado en la página 29).

[13] Peterson, Larry L. y Bruce S. Davie: *Computer Networks : A Systems Approach*. 5th edición, 2012, ISBN 9780123704801. (Citado en las páginas 20 y 21).

[14] Richardson, Matt y Shawn Wallance: *Getting Started with Raspberry Pi*. 2012, ISBN 9781449344214. (Citado en las páginas 14 y 15).

[15] Scherz, Paul: *Practical Electronics for Inventors*, 2000. (Citado en las páginas 7 y 8).

[16] Siciliano, Bruno, Oussama Khatib y Frans Groen: *Robotic Mapping and Exploration*, volumen 55. 2009, ISBN 9783642010965. (Citado en la página 27).

[17] Siciliano, Bruno, Lorenzo Sciavicco, Luigi Villiani y Giuseppe Oriolo: *Robotics Modeling, Planning and Control*. 2008, ISBN 9781846286414. (Citado en las páginas XIII, XIV, 2, 3, 5 y 48).

[18] Silvio, Jorge y Delgado Morales: *Implementación de un Robot Móvil con Ruedas de Accionamiento Diferencial*, 2013. (Citado en las páginas 8, 9, 23, 24, 25, 26 y 47).

[19] Solem, Jan Erik: *Programming Computer Vision with Python*. 2012, ISBN 0636920022923. (Citado en la página 53).

[20] Talbi, El Ghazali: *Metaheuristics: from design to implementation*, volumen 74. John Wiley & Sons, 2009. (Citado en la página 65).

[21] Thrun, Sebastian: *Artificial Intelligence for Robotics CS373*. `https://www.udacity.com/`. Consultado: 2015-03-29. (Citado en la página XIV).

[22] Thrun, Sebastian: *Probabilistic Robotics*. páginas 1–2, 2000. (Citado en la página XIII).

[23] Ulrich, Iwan y Illah Nourbakhsh: *Appearance-Based Obstacle Detection with Monocular Color Vision*. (August), 2000. (Citado en las páginas 59 y 61).

[24] Wilmshurst, Tim: *Designing Embedded Systems with PIC Microcontrollers*. 2007, ISBN 9780750667555. (Citado en las páginas 10 y 11).