



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA
MECÁNICA – MECATRÓNICA

TECNOLOGÍAS MÓVILES APLICADAS AL SEGUIMIENTO DE RUTAS EN
VEHÍCULOS AUTÓNOMOS

TESIS
QUE PARA OPTAR POR EL GRADO DE:
MAESTRO EN INGENIERÍA

PRESENTA:
LUIS ANGEL CABRAL GÚZMAN

TUTOR PRINCIPAL
DR. VICTOR JAVIER GONZÁLEZ VILLELA
FACULTAD DE INGENIERÍA

MÉXICO, D. F. NOVIEMBRE 2013



Universidad Nacional
Autónoma de México



UNAM – Dirección General de Bibliotecas

Tesis Digitales

Restricciones de uso

DERECHOS RESERVADOS ©

PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

JURADO ASIGNADO:

Presidente: Dr. Ramírez Reivich Alejandro C.

Secretario: M. I. Peñuelas Rivas Ulises Martin

Vocal: Dr. González Villela Víctor Javier

1 er. Suplente: M. I. Díaz Hernández Octavio

2 d o. Suplente: M.I. Flores Medero Navarro Billy Arturo

Lugar donde se realizó la tesis:
Facultad de Ingeniería

TUTOR DE TESIS:

DR. VICTOR JAVIER GONZÁLEZ VILLELA

FIRMA

ÍNDICE GENERAL

ÍNDICE GENERAL	3
ÍNDICE DE TABLAS Y FIGURAS	5
1. INTRODUCCIÓN	8
1.1. GENERALIDADES	8
1.2. TRABAJOS PREVIOS	9
1.3. APORTACIÓN, APROVECHANDO LA TECNOLOGÍA MÓVIL Y LOS MAPAS DIGITALES	11
1.4. OBJETIVOS	11
1.5. JUSTIFICACIÓN	12
2. DESCRIPCIÓN DEL SISTEMA	13
2.1. SISTEMA GPS	13
2.1.1. Descripción General del GPS	13
2.1.2. Constitución del Sistema GPS	14
2.1.2.1. Segmento Espacial	15
2.1.2.2. Segmento de Control	15
2.1.2.3. Segmento de Usuario	15
2.1.3. Funcionamiento del Sistema GPS	16
2.1.3.1. Cálculo de la Posición del Receptor	18
2.1.4. Códigos Utilizados y Servicios de posicionamiento GPS	20
2.1.5. Fuentes de Error en el Sistema	21
2.1.5.1. Retraso de la señal debido a la ionósfera y atmósfera	21
2.1.5.2. Otras fuentes de error	24
2.1.5.3. Dilución de la Precisión Geométrica (GDOP)	24
2.1.5.4. Disponibilidad Selectiva	26
2.1.6. Elipsoide y Geoide, Sistemas de Coordenadas y Proyecciones Cartográficas	27
2.1.6.1. Elipsoide	28
2.1.6.2. Geoide	28
2.1.6.3. Sistemas de Coordenadas Geodésicas	29
2.1.6.4. El sistema WGS 84 y diversas proyecciones cartográficas	31
2.1.6.5. Conversión de las Coordenadas Geográficas a Metros	31
2.2. ROBÓTICA MÓVIL	33
2.2.1. Locomoción en Robótica Móvil	34
2.2.2. Cinemática de Robots Móviles con Llantas	34
2.2.2.1. Posición del Robot	35
2.2.2.1.1. Matriz de Rotación	36
2.2.2.2. Llanta Fija, Centrada y Descentrada Orientable, Sueca y Esférica	42
2.2.2.2.1. Llanta Fija y sus Restricciones Cinemáticas	43
2.2.2.2.2. Llanta centrada orientable y sus Restricciones Cinemáticas	47
2.2.2.2.3. Llanta descentrada orientable y sus Restricciones Cinemáticas	48
2.2.2.2.4. Llanta Sueca u omnidireccional	49
2.2.2.2.5. Rueda Esférica o de Bola	49
2.2.2.3. Restricciones Cinemáticas del robot Móvil	49
2.2.2.3.1. Maniobrabilidad de un Robot Móvil	53
2.2.2.3.1.1. Grado de Movilidad (r_m)	54

2.2.2.3.1.2.	Grado de Manejabilidad (r_s)	55
2.2.2.3.1.3.	Grado de Maniobrabilidad (r_M).....	55
2.2.3.	<i>Postura de un Robot Móvil con Configuración Ackerman</i>	56
2.2.4.	<i>Modelo Cinemático de un Robot Móvil con Configuración Ackerman</i>	60
2.3.	PLATAFORMA MÓVIL	65
2.3.1.	<i>¿Qué es un Smartphone?</i>	66
2.3.2.	<i>Plataforma Android</i>	67
2.3.3.	<i>Arquitectura de Android</i>	67
2.3.3.1.	El núcleo Linux	68
2.3.3.2.	Runtime de Android	68
2.3.3.3.	Librerías Nativas.....	69
2.3.3.4.	Entorno de Aplicación.....	69
2.3.3.5.	Aplicaciones.....	70
2.3.4.	<i>Componentes de una aplicación Android</i>	70
2.3.4.1.	Activity.....	71
2.3.4.2.	Service	71
2.3.4.3.	Content Provider.....	71
2.3.4.4.	Broadcast Receiver.....	71
2.3.5.	<i>Ciclo de vida de una Activity</i>	72
2.3.6.	<i>Introducción a la Tecnología GSM</i>	76
2.3.6.1.	La Red Celular	77
2.3.6.2.	Arquitectura de la Red GSM.....	78
3.	IMPLEMENTACIÓN DEL SISTEMA.....	81
3.1.	SMARTPHONE DEL USUARIO	81
3.2.	ROBOT MÓVIL	83
3.2.1.	<i>Smartphone del robot móvil</i>	87
3.2.1.1.	Ubicación y Orientación del Robot Móvil.....	89
3.2.1.2.	Navegación	95
3.2.2.	<i>Interfaz Electrónica</i>	99
4.	PRUEBAS Y RESULTADOS.....	112
4.1.	SENSOR GPS.....	112
4.2.	SENSOR DE ORIENTACIÓN	114
4.3.	NAVEGACIÓN	116
4.3.1.	<i>Trayectoria definida por dos puntos</i>	118
4.3.2.	<i>Trayectoria definida por más de dos puntos</i>	120
5.	CONCLUSIONES Y TRABAJO A FUTURO.....	125
6.	APÉNDICES	128
	APÉNDICE A, CÓDIGO PROGRAMADO EN EL MICROCONTROLADOR.....	128
	APÉNDICE B, INSTALACIÓN DE MOTODEV	140
	APÉNDICE C, ABRIR UN ARCHIVO DE UN PROYECTO DE ANDROID.....	144
	EN ESTA SECCIÓN SE PRESENTA EL MÉTODO PARA ABRIR UN PROYECTO PREVIAMENTE REALIZADO Y PODER VISUALIZAR TODO EL CÓDIGO QUE CONFORMA A LA APLICACIÓN.....	144
	APÉNDICE D, CÓDIGO PROGRAMADO EN EL SMARTPHONE DEL USUARIO Y DEL ROBOT MÓVIL.....	146

ARTICULO PRESENTADO EN CIICA 2013-SOMIXXVII XONGRESO DE INSTRUMENTACIÓN.....	147
BIBLIOGRAFÍA.....	156

Índice de tablas y figuras

FIGURA 2.1. DIAGRAMA DEL SISTEMA A IMPLEMENTAR.	13
FIGURA 2.2. SEGMENTOS DEL SISTEMA GPS.....	14
FIGURA 2.3. SEGMENTO DE CONTROL, ESTACIONES DE SEGUIMIENTO.	16
FIGURA 2.4. SI SE DISPONE DE DOS SATÉLITES, EL RECEPTOR ESTARÁ EN EL CONTORNO DE LA CIRCUNFERENCIA FORMADA POR LA INTERSECCIÓN DE LAS ESFERAS.....	17
FIGURA 2.5. SI SE CONOCE LA DISTANCIA A TRES SATÉLITES, ENTONCES SOLO EXISTIRÁN DOS PUNTOS EN LOS QUE PODREMOS ESTAR.	18
FIGURA 2.6. EFECTO <i>MULTIPATH</i>	24
FIGURA 2.7. UNA BUENA ALINEACIÓN GEOMÉTRICA DE DOS SATÉLITES.	25
FIGURA 2.8. UNA MALA ALINEACIÓN GEOMÉTRICA DE DOS SATÉLITES.	25
FIGURA 2.9. DISTRIBUCIÓN DE LOS SATÉLITES Y ESTADO DE LA SEÑAL, EN EL CENTRO ESTÁ EL RECEPTOR.	26
FIGURA 2.10 ERROR EN LA MEDICIÓN CON LA SA APAGADA Y ACTIVA [13].....	27
FIGURA 2.11. RELACIÓN ENTRE LA SUPERFICIE DE LA TIERRA, EL GEOIDE Y EL ELIPSOIDE.....	29
FIGURA 2.12. CONCEPTO DE COORDENADAS GEODÉSICAS.	30
FIGURA 2.13. POSICIÓN DEL ROBOT SOBRE UN MARCO DE REFERENCIA GLOBAL, [5].	35
FIGURA 2.14. MARCO DE REFERENCIA 2 ROTADO.	36
FIGURA 2.15. MARCO DE REFERENCIA ROTADO ALREDEDOR DE Z1 EN SENTIDO HORARIO.....	37
FIGURA 2.16. VISTA EN EL PLANO DE ROTACIÓN ALREDEDOR DEL EJE Z.	37
FIGURA 2.17. CÁLCULO DE LAS COORDENADAS DEL PUNTO "P" EN EL SISTEMA 2.	38
FIGURA 2.18. ROTACIÓN ALREDEDOR DEL EJE "Y" UN ÁNGULO β	40
FIGURA 2.19. ROTACIÓN ALREDEDOR DEL EJE "X" UN ÁNGULO γ	40
FIGURA 2.20. COMPOSICIÓN DE ROTACIONES. A) ROTACIÓN ALREDEDOR DEL EJE Z, b) ROTACIÓN ALREDEDOR DEL EJE Y c) ROTACIÓN ALREDEDOR DEL EJE X.....	41
FIGURA 2.21. LLANTAS FIJAS Y ORIENTABLES CENTRADAS, [16].	43
FIGURA 2.22. LLANTA FIJA Y SUS RESTRICCIONES CINEMÁTICAS.	44
FIGURA 2.23. LLANTA FIJA Y SUS RESTRICCIONES CINEMÁTICAS EN LA VELOCIDAD ANGULAR.	45
FIGURA 2.24. LLANTAS ORIENTABLES DESCENTRADAS, [16].....	48
FIGURA 2.25. A) LLANTA SUECA A 90°, b) LLANTA SUECA A 45°.....	49
FIGURA 2.26. ROBOT TIPO DIFERENCIAL.	54
FIGURA 2.27. POSICIÓN DEL ROBOT SOBRE UN MARCO DE REFERENCIA GLOBAL, [5].	57
FIGURA 2.28. MOVIMIENTO DEL ROBOT MÓVIL SOBRE EL PLANO	58
FIGURA 2.29. PROYECCIONES DE LA VELOCIDAD LINEAL Y ANGULAR DEL ROBOT.	59
FIGURA 2.30. ESQUEMA DEL ROBOT MÓVIL CON CONFIGURACIÓN ACKERMAN.	61
FIGURA 2.31. VARIABLES EN LA RUEDA DERECHA TRASERA DEL ROBOT.....	62
FIGURA 2.32 ARQUITECTURA DEL SISTEMA OPERATIVO ANDROID.....	68
FIGURA 2.33 CICLO DE VIDA DE UNA <i>ACTIVITY</i>	73
FIGURA 2.34. CÓDIGO DE LOS TRES MÉTODOS PRINCIPALES DE UNA <i>ACTIVITY</i>	76
FIGURA 2.35. CELDAS QUE CUBREN CIERTA ÁREA GEOGRÁFICA.	77

FIGURA 2.36. DISTRIBUCIÓN DE FRECUENCIAS EN CADA CELDA.	78
FIGURA 2.37. ARQUITECTURA DE LA RED GSM.	79
FIGURA 3.1 ARQUITECTURA GENERAL DEL SISTEMA.	81
FIGURA 3.2 PANTALLA PRINCIPAL DEL USUARIO.	82
FIGURA 3.3. DIAGRAMA DE SECUENCIA DE LA APLICACIÓN DEL USUARIO.	84
FIGURA 3.4. ROBOT MÓVIL AUTÓNOMO EMPLEADO.	85
FIGURA 3.5. ROBOT MÓVIL TIPO ACKERMAN.	86
FIGURA 3.6. PANTALLA PRINCIPAL DEL USUARIO PARA EL ROBOT MÓVIL.	87
FIGURA 3.7. ARQUITECTURA DE LA APLICACIÓN DEL ROBOT MÓVIL.	88
FIGURA 3.8. PROYECCIÓN DE MERCATOR DEL MUNDO.	89
FIGURA 3.9. DIAGRAMA DE SECUENCIA DE LA APLICACIÓN DEL ROBOT MÓVIL.	90
FIGURA 3.10. ÁNGULO DE AZIMUT.	91
FIGURA 3.11. CÓDIGO BÁSICO PARA EL USO DEL GPS EN DISPOSITIVOS CON SISTEMA ANDROID.	92
FIGURA 3.12. CÓDIGO BÁSICO PARA USAR ALGUNOS SENSORES EN EL SISTEMA ANDROID.	93
FIGURA 3.13. CÓDIGO BÁSICO PARA OBTENER EL VALOR DE ALGUNOS SENSORES EN EL SISTEMA ANDROID.	94
FIGURA 3.14. CÓDIGO BÁSICO PARA MANDAR MENSAJES SMS.	96
FIGURA 3.15. POSICIÓN DEL ROBOT Y LA META A ALCANZAR, ADAPTADO DE [5].	97
FIGURA 3.16. ESQUEMA ELECTRÓNICO DEL ROBOT MÓVIL.	100
FIGURA 3.17. INTERFAZ BLUETOOTH AL SERIAL.	101
FIGURA 3.18. MICROCONTROLADOR PIC16F877A.	102
FIGURA 3.19. ALGORITMO DE LA ELECTRÓNICA, MÉTODO PRINCIPAL.	103
FIGURA 3.20. ALGORITMO DE LA ELECTRÓNICA, CONTINUACIÓN DEL MÉTODO PRINCIPAL.	104
FIGURA 3.21. ALGORITMO EMPLEADO EN LA INTERRUPCIÓN DEL PIN 0 DEL PUERTO B.	105
Figura 3.22. Algoritmo empleado en la interrupción del pin 4 al 7 del puerto B.	105
FIGURA 3.23. ALGORITMO EMPLEADO PARA LA COMUNICACIÓN VÍA SERIAL RS232.	106
FIGURA 3.24. ALGORITMO EMPLEADO PARA LA COMUNICACIÓN VÍA SERIAL RS232, CONTINUACIÓN.	107
FIGURA 3.25. ALGORITMO EMPLEADO PARA EL CONTROL DE LOS SERVOMOTORES DEL ROBOT.	108
FIGURA 3.26. ALGORITMO EMPLEADO PARA EL CONTROL DE LA VELOCIDAD DE LOS MOTORES DE CORRIENTE CONTINÚA.	109
FIGURA 3.27. ALGORITMO EMPLEADO PARA EL CONTROL DE LA VELOCIDAD DE LOS MOTORES DE CORRIENTE CONTINÚA, (CONTINUACIÓN).	110
FIGURA 3.28. ENCÓDER Y LLANTA FIJA UTILIZADA EN EL ROBOT MÓVIL.	111
FIGURA 4.1. CONJUNTO DE DATOS REPORTADOS POR EL MOTOROLA MB860.	112
FIGURA 4.2. CONJUNTO DE DATOS REPORTADOS POR EL MOTOROLA XT910.	113
FIGURA 4.3. ERROR DE LA LATITUD (AZUL) Y LONGITUD (ROJO) EN METROS PARA EL MOTOROLA MB860.	113
FIGURA 4.4. ERROR DE LA LATITUD (AZUL) Y LONGITUD (ROJO) EN METROS PARA EL MOTOROLA XT910.	114
FIGURA 4.5. COMPARACIÓN ENTRE UNA BRÚJULA ANALÓGICA Y EL SENSOR DE ORIENTACIÓN DEL MODELO MOTOROLA MB860.	115
FIGURA 4.6. COMPARACIÓN ENTRE UNA BRÚJULA ANALÓGICA Y EL SENSOR DE ORIENTACIÓN DEL MODELO MOTOROLA XT910.	115
FIGURA 4.7. DIAGRAMA DEL SISTEMA A IMPLEMENTAR.	117
FIGURA 4.8. TRAYECTORIA DE REFERENCIA EN AZUL Y TRAYECTORIA REALIZADA POR EL ROBOT MÓVIL EN ROJO.	119
FIGURA 4.9. COMPARACIÓN ENTRE LA TRAYECTORIA DESEADA A SEGUIR (AZUL), CON LA REALIZADA POR EL ROBOT MÓVIL (ROJO).	119
FIGURA 4.10. ERROR DE SEGUIMIENTO DE LA RUTA DE REFERENCIA DEL ROBOT.	120
FIGURA 4.11. TRAYECTORIA DE REFERENCIA EN AZUL Y TRAYECTORIA REALIZADA POR EL ROBOT EN ROJO PARA MÁS DE DOS PUNTOS.	122

FIGURA 4.12. COMPARACIÓN ENTRE LA TRAYECTORIA DESEADA A SEGUIR (AZUL), CON LA REALIZADA POR EL ROBOT MÓVIL (ROJO) PARA MÁS DE DOS PUNTOS.	123
FIGURA 4.13. ERROR DE SEGUIMIENTO DE LA RUTA DE REFERENCIA DEL ROBOT PARA MÁS DE DOS PUNTOS.....	123
TABLA 2.1. CLASIFICACIÓN DE LAS CAPAS DE LA IONÓSFERA.	22
TABLA 2.2. CLASIFICACIÓN DEL VALOR GDOP.....	26
TABLA 2.3. ALGUNOS SISTEMAS DE REFERENCIA Y SU ELIPSOIDE ASOCIADA.	28
TABLA 2.4. DISTANCIA QUE REPRESENTA UN GRADO DE LATITUD Y DE LONGITUD EN FUNCIÓN DE LA LATITUD	33
TABLA 2.5. LOS CINCO TIPOS DE LLANTAS O RUEDAS.	42
TABLA 2.6. CONFIGURACIONES BÁSICAS DE LOS ROBOTS MÓVILES, [16].	56
TABLA 2.7. PARÁMETROS DE LA CINEMÁTICA DEL ROBOT MÓVIL.	62
TABLA 2.8 SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN EN LOS SMARTPHONE.	66
TABLA 2.9 SISTEMAS OPERATIVOS MÁS VENDIDOS EN EL 2011 Y 2012.	67
TABLA 2.10. MÉTODOS PRINCIPALES DE UNA ACTIVITY.....	75
TABLA 3.1 COMPONENTES DE LA INTERFAZ DEL USUARIO.	83
TABLA 3.2. PARÁMETROS Y SUS VALORES DE LA CINEMÁTICA DEL ROBOT MÓVIL.	86
TABLA 3.3. COMPONENTES DE LA INTERFAZ DEL USUARIO PARA EL ROBOT MÓVIL.	87
TABLA 4.1. ERROR PRESENTADO ENTRE LA BRÚJULA ANALÓGICA Y DIGITAL DE LOS MODELOS MOTOROLA MB810 Y XT910....	116

1. INTRODUCCIÓN

1.1. Generalidades

Han pasado casi 127 años, desde que la Sra. Bertha Benz, esposa del ingeniero alemán Karl Friedrich Benz, junto con sus dos hijos, tomaron sin permiso aquel vehículo en el que Karl había trabajado durante 2 años aproximadamente, para poder ir a visitar a unos familiares. El viaje fue un éxito, 106 km en 8 horas sin carreteras ni estaciones de servicio. Este hecho ha marcado un parte aguas para toda la humanidad, se había realizado el primer viaje en un automóvil [26].

Desde entonces, el automóvil se ha convertido en el objeto de estudio de muchos emprendedores a lo largo de todo el mundo, cambiando por completo nuestra concepción de la sociedad, la economía y la cultura, entre otras cosas. Revolucionó el mundo, creó las cadenas de producción, acortó distancias y su desarrollo ha contribuido al avance tecnológico, que desde entonces ha venido en acenso. Los vehículos han formado parte de nuestra vida diaria.

Por otra parte, la evolución de la tecnología alcanzado a este gran invento y con ello ha traído comodidad, mayor seguridad, eficiencia y en algún grado cierta autonomía o asistencia al conductor, de tal manera que por su configuración actual y por la íntima relación que guarda con las disciplinas de la ingeniería mecánica, electrónica y el control, los sistemas automotrices, pueden ser considerados productos inherentemente mecatrónicos.

El estudio de los vehículos autónomos es un área relativamente nueva de la investigación. Se puede considerar como una rama especializada de la robótica y sólo ha sido posible gracias a los avances tecnológicos más recientes, entre los cuales podemos hablar de sistemas de visión, procesamiento de sensores como encóders, de velocidad, etc.

Desafortunadamente, hay una serie de cuestiones que deben abordarse antes de que los vehículos autónomos se conviertan en algo común en nuestra vida diaria. Uno de los problemas más restrictivos es la navegación y el sistema de control o de procesamiento de un vehículo autónomo. Estos sistemas ofrecen a un vehículo, la inteligencia necesaria para realizar una tarea o la capacidad de moverse en su entorno. Estos sistemas tienen un gran impacto en el rendimiento de un vehículo autónomo, la robustez, la utilidad y el costo.

Los dispositivos móviles, como celulares y tabletas, traen incorporados varios sensores, los cuales podrían ser muy útiles en la robótica, además de la conexión a diferentes bases de datos como "*Google Maps*", sabiendo aprovechar estas tecnologías, no solo serían una buena herramienta, sino que también disminuiría los costos y la complejidad de la

construcción, estas cualidades combinadas con un sistema de comunicación vía GSM, abren la posibilidad de establecer un sistema de control de largo alcance con menores dificultades.

En este trabajo se presenta el desarrollo de un vehículo autónomo con configuración Ackerman, el cual es comandado por el usuario usando dispositivos y tecnologías móviles. Por medio de una interfaz gráfica de usuario, programada en un "*smartphone*", se reporta la ubicación del robot en un mapa y se le indica hacia dónde dirigirse, señalando un punto en el mismo. Las intenciones del usuario se transmiten vía SMS hasta el robot, éste crea una trayectoria hasta su objetivo con ayuda de "*Google Maps*", empleando un algoritmo de navegación basado en la información de un compás electrónico y GPS embebidos en un sistema móvil.

1.2. Trabajos Previos

La visión de los autos autónomos usados mientras sus conductores se relajan no es nueva, ya que fue presentado en 1939 en la Feria Mundial de Nueva York, en el pabellón de General Motors [1]. Sin embargo, más investigaciones y desarrollos se iniciaron a principios de los 70's, cuando la competencia en la carrera espacial entre la ex Unión Soviética y los Estados Unidos inició con respecto a la supremacía de la exploración del espacio exterior. Ya había planes para misiones espaciales en Marte, y debido a la distancia del planeta a la Tierra, simplemente no era posible controlar un robot por un operador desde la tierra viendo una imagen en vivo, era necesario un robot autónomo.

A mediados de la década de 1980, el Vehículo Terrestre Autónomo de DARPA (ALV) completó con éxito la primera demostración en carretera con radar láser.

En Alemania, el pionero en conducción autónoma, Ernst Dickmanns, llevaba a cabo sus primeras actividades en 1980 en la Universidad de Bundeswehr München, equipó un auto Mercedes Benz con cámaras y otros sensores alcanzando una velocidad máxima de 25 m/s en una carretera. El vehículo llamado VaMoR (Vehículo de Movilidad Autónoma y Visión por Computadora) fue un éxito y una inspiración para los próximos proyectos.

Los autos autónomos son un conjunto de diferentes tecnologías que van desde sistemas de censado de obstáculos, sistemas de navegación, sistemas de visión, etcétera, dependiendo de las tecnologías y algoritmos empleados, estos sistemas se vuelven más económicos y eficientes, particularmente en el sistema de navegación, se han tenido diferentes propuestas, a continuación se presenta una breve descripción de algunos de estos trabajos.

Las nuevas tecnologías aportan ciertas ventajas en el desarrollo de la robótica, en este tema, en el año 2003, en la Universidad Carlos III de Madrid, Alaa Mohamed Khamis Rashwan [2] realiza un análisis en donde se estudian las ventajas e inconvenientes de utilizar el internet como medio de comunicación en los sistemas de teleoperación con robots móviles empleando dispositivos móviles, también estudia el uso de los dispositivos conocidos como PDAs como elementos de interacción con el robot, este trabajo señala las ventajas en cuanto a movilidad y portabilidad que las nuevas tecnologías pueden aportar a la robótica.

En el año del 2008, la SOMIM presenta un artículo de un robot terrestre de cuatro ruedas, el cual es controlado a través de un asistente digital personal (PDA) [3], el cual presenta una interfaz gráfica para el usuario, este dispositivo transmite al vehículo una serie de puntos que forman el recorrido deseado. Este sistema comprueba la facilidad con la que las nuevas tecnologías pueden incorporarse a los sistemas robóticos, sin embargo, no es un sistema totalmente autónomo ya que la creación de la ruta depende totalmente del usuario.

En el 2011, Menard y Timothy [4], del departamento de computación de la Universidad de Alaska, realizan un estudio comparando la capacidad de rastreo de vehículos a través del GPS de 3 teléfonos inteligentes, midiendo su precisión. Este trabajo proporciona una referencia importante para poder apreciar la precisión que se puede lograr con estos dispositivos a través de su GPS.

Dentro de la investigación para el desarrollo de un sistema de navegación para un vehículo autónomo, en el año 2004, Víctor J. González del Centro de Investigación en Mecatrónica de la escuela de Mecánica de Wolfson y del Departamento de Ingeniería Mecatrónica de la Facultad de Ingeniería de la UNAM [5], realiza un estudio acerca de la habilidad en la generación de una ruta y la evasión de obstáculos de los robots móviles usando campos potenciales. Este trabajo permite automatizar la generación de trayectorias conociendo datos del robot, su orientación y su posición, además de la localización de los obstáculos y la meta.

En el 2012, Sivaraj, D. [6], propone un sistema de líneas dibujadas en el camino que se podían utilizar como medios de referencia para el vehículo, además de comunicación inalámbrica entre estos para informar de su velocidad, pensando en la aplicación de un sistema inteligente de transporte. El inconveniente de esta técnica es la necesidad de marcas de referencia, dentro del área de trabajo, para que el sistema de visión pueda realizar su función.

En el 2012, Saman, Abu Bakar Sayuti del Departamento de Ingeniería Eléctrica y Electrónica de la Facultad de Ingeniería de la Universidad Tecnológica PETRONAS [7], proponen una navegación autónoma basada en una máquina de estados finita. La ventaja de usar este método propuesto es que con una programación sencilla se obtiene una navegación autónoma, sin embargo, debido a la gran diversidad de formas que una ruta puede contener el número de estados puede crecer más allá que sólo adelante, atrás derecha, izquierda y alto.

1.3. Aportación, aprovechando la tecnología móvil y los mapas digitales.

La navegación autónoma tiene ya varias décadas [1], se han explorado diferentes técnicas y algoritmos [5], [6] y [7], entre otros; así mismo, el desarrollo de los robots móviles ha tenido un gran avance, desde su modelado cinemático y dinámico, hasta su control [5], de igual manera, la unidad de procesamiento (computadora, laptop) está siendo sustituida por unidades más pequeñas que ofrecen portabilidad y reducen los costos de desarrollo [2] [3]. En este trabajo se desea montar un sistema capaz de censar variables de suma importancia, como son la posición y la orientación de un prototipo de vehículo, estas variables serán medidas utilizando la tecnología contenida dentro de un teléfono inteligente, de tal forma que con esta información el algoritmo de navegación [5] sea capaz de corregir la velocidad y la orientación de las llantas del prototipo para seguir de la manera más precisa posible una trayectoria previamente obtenida de forma automática por el sistema, atendiendo las instrucciones del usuario transmitidas utilizando la tecnología GSM.

Para obtener una navegación autónoma, con la posibilidad de ser alterada dependiendo de la posición actual y el lugar de destino, se requiere conocer el área de trabajo, en este caso la ciudad, estos datos pueden provenir de mapas digitales y un algoritmo capaz de generar la ruta necesaria; la propuesta de este trabajo es aprovechar no sólo las nuevas tecnologías y sensores que los teléfonos inteligentes tienen, sino también las bases de datos existentes, en este caso "*Google Maps*", para la obtención automática de la ruta a seguir por el robot autónomo.

1.4. Objetivos

El objetivo general es desarrollar un prototipo de un vehículo terrestre autónomo el cual aproveche los avances que pueden aportar las tecnologías móviles. Como objetivos particulares se tienen:

- ❖ Implementar algoritmos de control en los dispositivos móviles aprovechando las tecnologías incorporadas en estos (GSM, sensores y capacidad de programación).
- ❖ Montar un sistema de rastreo, basado en GPS y sensores de navegación que los sistemas móviles contienen, particularmente con el sistema operativo Android, que permita obtener la posición, y orientación del robot autónomo dentro de un área urbana.
- ❖ Utilizar la información obtenida de los sensores de navegación y GPS del dispositivo móvil para alimentar el algoritmo de navegación, con la finalidad de que el prototipo del automóvil autónomo siga una trayectoria obtenida por medio de la tecnología de “*Google Maps*”.

1.5. Justificación

Los autos autónomos son productos que pueden satisfacer las necesidades de las personas o hacer la vida más cómoda y segura, su uso para transportar personas que por alguna razón no pueden usar autos convencionales y, por lo tanto se ven restringidos en su capacidad de transportarse dentro de la ciudad de forma independiente, es un ejemplo de esta carestía, así como la de evitar accidentes y hacer más eficientes los medios de transporte al automatizarlos. Considerando lo anterior, la navegación para autos autónomos es una habilidad básica que estos robots deben de tener, sin embargo, la tecnología empleada actualmente para tal fin es en muchos casos costosa, voluminosa y complicada. Se experimenta con las tecnologías móviles, debido a que, actualmente, estos dispositivos pueden ser fácilmente programados y cuentan con los sensores de navegación necesarios para la instrumentación de un robot móvil, sumado a su capacidad de establecer una conexión a internet y acceder a bancos de información y mapas digitales como “*Google Maps*”, para obtener rutas desde su ubicación hasta su destino, se considera que contienen características muy apropiadas para su uso en la robótica y de forma particular, para emplear algoritmos de navegación para robots móviles en entornos urbanos.

2. DESCRIPCIÓN DEL SISTEMA

El sistema está compuesto por cuatro elementos interconectados entre sí, se tienen dos dispositivos móviles, uno para el usuario y otro para el robot, comunicándose entre ellos por medio de mensajería SMS, la red de satélites GPS y el robot móvil, que sirve como prototipo de un auto autónomo, figura 2.1.

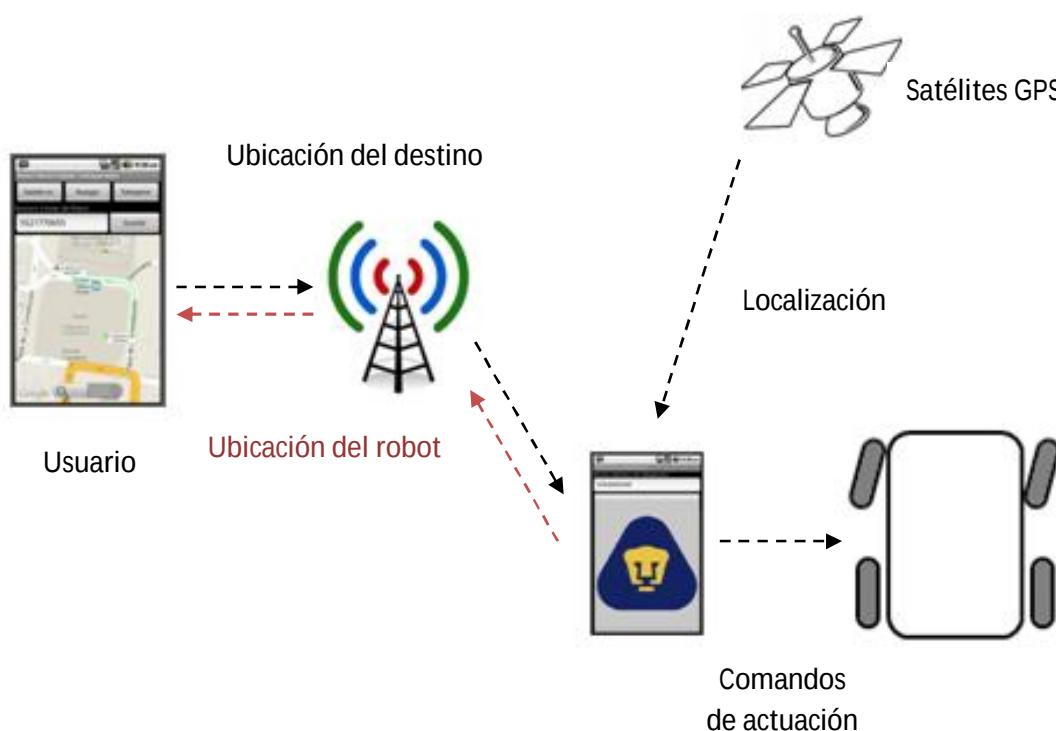


Figura 2.1. Diagrama del sistema a implementar.

2.1. Sistema GPS

El Sistema de Posicionamiento Global (GPS) es un sistema de navegación basado en satélites que fue desarrollado por el Departamento de Defensa de los Estados Unidos a principios de 1970, proporciona un posicionamiento continuo, en cualquier parte del mundo. Inicialmente, el GPS fue desarrollado como un sistema militar para satisfacer las necesidades bélicas de Estados Unidos. Sin embargo, más tarde se hizo disponible para todos, y ahora es un sistema de uso al que se pueden acceder tanto los militares como civiles [8].

2.1.1. Descripción General del GPS

El GPS consiste en una constelación de 27 satélites, de los cuales 3 son de reserva, y es conocido como NAVSTAR (Navegación por satélite con medición de tiempo y distancia).

Para asegurar la cobertura continua en todo el mundo, los satélites GPS están dispuestos de tal manera que, por lo menos, cuatro satélites puedan ser accesibles por un receptor, en realidad un receptor tiene acceso de cuatro a diez satélites GPS en cualquier parte del mundo.

Cada satélite GPS transmite continuamente una señal de radio. Cuando un receptor GPS está encendido, se recoge la señal GPS a través de la antena del receptor. Una vez que el receptor adquiere la señal GPS, la procesará utilizando su software incorporado. El resultado parcial del procesamiento de la señal se compone de las distancias a los satélites GPS a través de los códigos digitales (conocidos como los pseudorrangos) [9] y las coordenadas del satélite a través de un mensaje de navegación que transmite este mismo.

Las órbitas de los satélites GPS son casi circulares (en realidad se tiene una forma elíptica con una excentricidad máxima de aproximadamente 0.01), con una inclinación aproximada de 55° con respecto al ecuador. La altitud del satélite es de unos 20,200 kilómetros por encima de la superficie de Tierra [10].

2.1.2. Constitución del Sistema GPS

El Sistema de Posicionamiento Global está formado por tres segmentos (figura 2.2):

- Segmento Espacial.
- Segmento de Control.
- Segmento de Usuario.

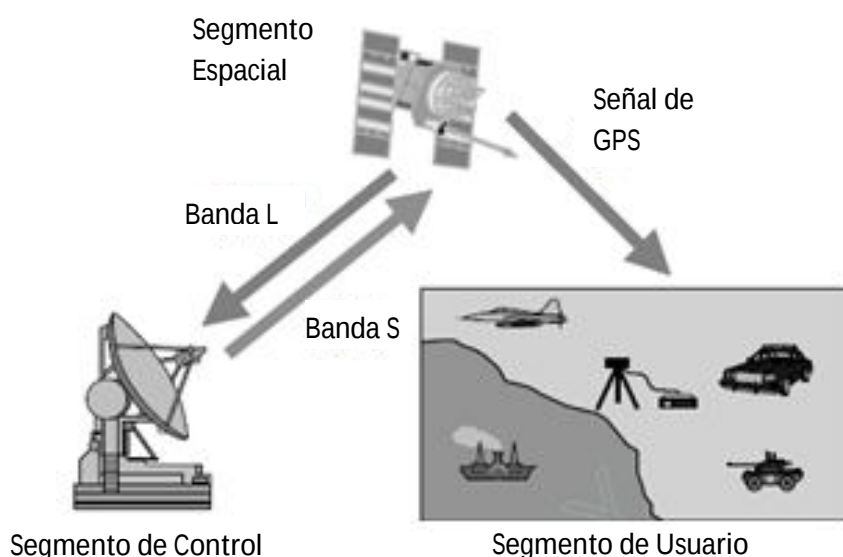


Figura 2.2. Segmentos del sistema GPS.

2.1.2.1. Segmento Espacial

Este segmento se encuentra formado por los satélites de la constelación NAVSTAR. Como ya se ha mencionado, en la actualidad esta constelación se encuentra formada por 27 satélites, de los cuales 3 son de reserva y los otros 24 están distribuidos en 6 planos orbitales, se nombran como A, B, C, D, E y F (cada plano con 4 satélites en una órbita casi circular). Cada plano tiene una separación de 60° y tienen una inclinación de 55° con el plano del ecuador [9].

La constelación NAVSTAR, con esta configuración, posibilita la localización con un receptor de manera continua durante las 24 horas del día, en cualquier lugar de la tierra. Cada satélite efectúa una revolución completa alrededor de la tierra cada 12 horas aproximadamente [8] y puede ser identificado de diversos modos:

- ❖ Por su número NAVSTAR (SVN).
- ❖ Por su número orbital. Un ejemplo seria el satélite 3D, que corresponde al satélite número tres del plano orbital D.

2.1.2.2. Segmento de Control

El segmento de control del sistema GPS consta de una red mundial de estaciones de seguimiento, con una estación maestra de control (MCS) situado en los Estados Unidos en Colorado Springs, Colorado, otras cuatro estaciones secundarias en Hawaii, Isla Ascensión, Diego García y Kwajalein, figura 2.3.

Las estaciones de seguimiento se encuentran espaciadas de igual forma en longitud y sus coordenadas de ubicación son conocidas con una gran precisión (figura 2.3). Los datos recogidos por las estaciones secundarias son enviados a la MCS, donde son procesados. Su misión es dar seguimiento continuo de todos los satélites de la constelación NAVSTAR para los siguientes fines:

- ❖ Establecer la órbita de cada satélite, así como determinar el estado de sus osciladores.
- ❖ Hallados los parámetros anteriores, estos son empaquetados y cargados en los satélites GPS a través del enlace con la banda S [8] para después difundirlos a los usuarios.

2.1.2.3. Segmento de Usuario

Se encuentra formado por los instrumentos que todos los usuarios necesitamos para utilizar el sistema GPS, tanto por civiles como por militares.



Figura 2.3. Segmento de control, estaciones de seguimiento.

Un equipo GPS se integra básicamente por un receptor o sensor con antena que se comunica con los satélites, la antena puede ser externa o integrada en el sensor, este sensor recibe señales RF que posteriormente son amplificadas y digitalizadas, a continuación se procesan para calcular la posición del usuario [9].

Los GPS modernos son muy ligeros, tanto que hasta el momento están siendo integrados dentro de otros equipos como es el caso de los celulares [4], sin embargo dependiendo de su precisión, los hay de diferentes precios y tamaños.

2.1.3. Funcionamiento del Sistema GPS

El funcionamiento del GPS es bastante sencillo, aunque encierra un sin fin de complejidades técnicas y una gran diversidad de subsistemas, propios de una red satelital, sin embargo, los modernos receptores permiten su utilización por personas sin ninguna capacitación.

La base de todo el sistema es la medición constante de la distancia que separa al receptor de los satélites. Conociendo que los satélites están a 20,200 Km de la superficie terrestre, podemos pensar en una esfera que tiene en su centro a un satélite y nosotros estamos colocados en la superficie de esta esfera.

Como se mencionó antes, cada satélite GPS transmite continuamente una señal de radio. Cuando un receptor GPS está encendido, se recoge la señal GPS a través de la antena del receptor. Una vez que el receptor adquiere la señal GPS, la procesará utilizando su

software incorporado. El resultado parcial del procesamiento de la señal se compone de las distancias a los satélites GPS a través de los códigos digitales (conocidos como los pseudorrangos) [9] y las coordenadas de satélite a través de un mensaje de navegación que transmite este mismo.

En teoría, sólo tres distancias a tres satélites rastreados simultáneamente son necesarias, sin embargo, un cuarto satélite es necesario para tener en cuenta el retraso del reloj del receptor.

Si se dispone de la distancia a dos satélites, nos encontraremos dentro del contorno de la circunferencia en donde se interceptan las dos esferas, cada esfera tiene en su centro las coordenadas de cada satélite y de radio la distancia a cada uno de ellos respectivamente (figura 2.4). Si además disponemos de la distancia de un tercer satélite, la posición del receptor estará determinada por la intersección de la tercera esfera con la circunferencia anterior (figura 2.5).

La intersección de la última esfera con la circunferencia da como resultado dos puntos en el espacio, en los cuales podremos encontrarnos. Para conocer en cuál de los dos puntos nos encontramos, el sistema puede descartar uno de los dos por ser absurdo o realizar una cuarta medición a otro satélite.

El funcionamiento del sistema GPS es bastante fácil de entender, solo se necesitan conocer las distancias a los satélites y su localización, todo lo demás que se mencione serán detalles técnicos utilizados para llevar a cabo estas mediciones.

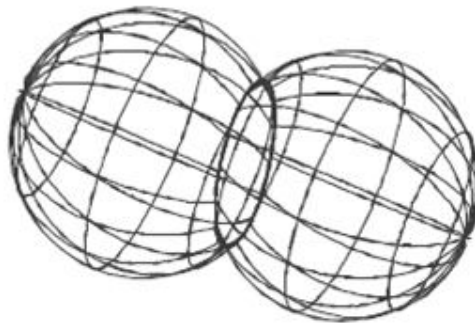


Figura 2.4. Si se dispone de dos satélites, el receptor estará en el contorno de la circunferencia formada por la intersección de las esferas.

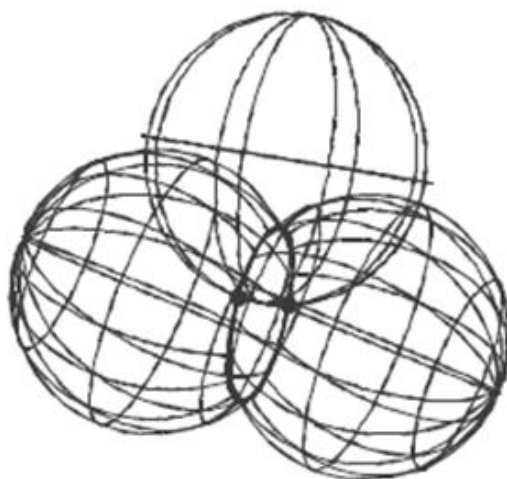


Figura 2.5. Si se conoce la distancia a tres satélites, entonces solo existirán dos puntos en los que podremos estar.

La precisión obtenida con el método descrito anteriormente era hasta hace poco limitada a 100 metros para el componente horizontal, 156 metros para la componente vertical, y 340 ns para el componente de tiempo aproximadamente. Este nivel de baja precisión se debió al efecto conocido como disponibilidad selectiva destinado a los receptores para la población civil. **Con la decisión presidencial de los Estados Unidos de poner fin a la disponibilidad selectiva en el año 2000, la precisión obtenida mejoró bastante, se tiene acceso a la precisión manejada por los militares autorizados, sin embargo, esta decisión puede cambiar si el departamento de defensa de los Estados Unidos lo considera necesario por cuestiones de seguridad [11].** Para mejorar aún más la precisión del posicionamiento, el método diferencial, que emplea dos receptores simultáneamente, puede ser utilizado. En este caso, el nivel de precisión del posicionamiento puede mejorar hasta en algunos casos a unos pocos centímetros.

2.1.3.1. Cálculo de la Posición del Receptor.

El concepto es fácil, se basa en que los satélites GPS emiten señales de radio que viajan a la velocidad de la luz (aproximadamente 300 000 Km/s en el vacío). Conociendo el momento en que un satélite emite una señal y el momento en el que la recibimos en nuestro receptor, es fácil conocer la distancia, de ahí su nombre de la constelación de satélites GPS NAVSTAR (Navegación por satélite con medición de tiempo y distancia), el cálculo de la distancia se resume en:

$$S = C * \Delta t \dots \dots \dots 2.1$$

Donde:

- C = Velocidad de la luz.
- Δt = Tiempo transcurrido durante el viaje de la señal desde el satélite hasta el receptor.

La complejidad está en la capacidad tecnológica en medir con gran exactitud intervalos de tiempo muy cortos. Para tal motivo, se necesitan relojes de gran precisión, tanto en los receptores como en los satélites. Los satélites GPS poseen relojes atómicos que oscilan entre 10^{-11} y 10^{-14} segundos, los receptores normalmente poseen relojes más económicos hechos de cuarzo que oscilan entre 10^{-9} segundos.

La diferencia en la precisión es grande, tan sólo la luz recorre 30 cm en 10^{-9} segundos y 0.003 mm en 10^{-14} segundos; una solución al problema sería instalar relojes atómicos a los receptores, sin embargo, esto no es posible debido a su alto costo. Una solución más sencilla y económica es efectuar una cuarta medición a otro satélite estableciendo así un sistema de cuatro ecuaciones con cuatro incógnitas (tres coordenadas de la localización del punto del receptor y una desviación del reloj del receptor).

Recordando que la ecuación de una esfera con centro en (X_0, Y_0, Z_0) y radio R está dada por la ecuación número 2.2:

$$(X - X_0)^2 + (Y - Y_0)^2 + (Z - Z_0)^2 = R^2 \dots \dots \dots 2.2$$

Sí tenemos cuatro esferas que corresponden a las formadas por los cuatro satélites consultados, entonces tenemos una ecuación para cada esfera, el centro de cada una está dado por la localización del satélite respectivo, que son puntos conocidos. El valor de los radios R_1, R_2, R_3 y R_4 de cada esfera estaría dada por la distancia del satélite al receptor, es decir:

$$\begin{aligned} R_1 &= C * \Delta t_1 \\ R_2 &= C * \Delta t_2 \\ R_3 &= C * \Delta t_3 \\ R_4 &= C * \Delta t_4 \end{aligned}$$

2.3

Donde Δt_n = Tiempo transcurrido durante el viaje de la señal desde el satélite “n” hasta el receptor.

Tomando en cuenta que existen errores debido a la imprecisión de los relojes del receptor, se debe de corregir el tiempo añadiendo una nueva incógnita conocida como *Clock Bias* (CB) o Sesgo de Reloj, que hace referencia al error en el resultado debido a

factores presentes cuando fueron tomadas las mediciones. De esta manera las distancias o los radios de las esferas serían:

$$\begin{aligned}R_1' &= C * \Delta t_1 - CB \\R_2' &= C * \Delta t_2 - CB \\R_3' &= C * \Delta t_3 - CB \\R_4' &= C * \Delta t_4 - CB\end{aligned}$$

2.4

En resumen, conocemos el centro de cada esfera, dado por las coordenadas de cada satélite, su radio, con su factor de corrección CB y, se sabe que todas estas esferas comparten un punto en común (X, Y, Z) , este punto es precisamente la ubicación del receptor y la incógnita, por lo tanto, el sistema de ecuaciones quedaría establecido de la siguiente forma:

$$\begin{aligned}(X - X_1)^2 + (Y - Y_1)^2 + (Z - Z_1)^2 &= (R_1 + CB)^2 \\(X - X_2)^2 + (Y - Y_2)^2 + (Z - Z_2)^2 &= (R_2 + CB)^2 \\(X - X_3)^2 + (Y - Y_3)^2 + (Z - Z_3)^2 &= (R_3 + CB)^2 \\(X - X_4)^2 + (Y - Y_4)^2 + (Z - Z_4)^2 &= (R_4 + CB)^2\end{aligned}$$

2.5

Donde (X, Y, Z) son las coordenadas del punto que queremos determinar, CB es la desviación del reloj que también es una incógnita y (X_n, Y_n, Z_n) son las coordenadas de los centros de cada una de las cuatro esferas (conocidas, pues coinciden con la posición de cada satélite).

La unidad de control que acompaña al receptor GPS es el encargado de resolver este sistema de ecuaciones [9]. El hecho de que necesitemos al menos cuatro satélites, implica que el receptor GPS debe de ser capaz de recoger cuatro señales como mínimo de forma simultánea, esto se logra al implementar canales independientes para la recepción en los sensores.

2.1.4. Códigos Utilizados y Servicios de posicionamiento GPS

Como se mencionó en el apartado anterior, el saber en qué momento los satélites emitieron las señales recibidas es muy importante para poder determinar una posición. Tanto los receptores y los satélites GPS son diseñados para que el oscilador presente en ellos genere el mismo código a la misma hora, de esta manera al emitir algún satélite su señal, ésta tardará un tiempo en llegar hasta el receptor, entonces si se compara con el código generado por el receptor, es posible determinar el desfase de tiempo.

Cada satélite transmite señales de radio de bajo poder en frecuencias designadas como L1 y L2, estas frecuencias son conocidas como *frecuencias portadoras*. La frecuencia central de L1 es 1575.42 MHz y la de L2 es 1227.6 MHz.

Sobre estas dos frecuencias portadoras se transmiten a su vez dos códigos: el C/A y el P.

El código P (Preciso) se transmite a la frecuencia de 10.23 MHz, ofrece mayor precisión y se utiliza en el denominado “*Servicio de Posicionamiento Preciso (Precise Positioning Service) PPS*” y se transmite sobre la portadora L2; en la portadora L1 se transmite un código denominado P(Y) el cual es el código P, pero encriptado o protegido.

El código C/A (*Course/Acquisition*) o también denominado S (*Standard*), es de una frecuencia de 1.023 MHz. Esta señal ofrece menor precisión y está disponible para cualquier persona alrededor del mundo y se utiliza en el llamado *Servicio de Posicionamiento Estándar (Standard Positioning Service) SPS* y se transmite sobre la portadora L1. Originalmente el SPS proveía una precisión en el orden de 100 metros en el eje horizontal y 156 metros en el eje vertical, en el 95% de las muestras. Con la reciente decisión presidencial de los Estados Unidos de suspender la disponibilidad selectiva, la precisión de posicionamiento SPS está actualmente en un nivel comparable a la de los PPS [11].

Junto con estos dos códigos, se envía un mensaje de navegación, éste contiene la información de la órbita del satélite y del reloj, en general contiene los mensajes de estado del sistema y un modelo de retardo de la ionósfera. Las señales de los satélites están sincronizados con los relojes atómicos de alta precisión [9].

2.1.5. Fuentes de Error en el Sistema

Un receptor basa su localización midiendo su distancia hasta los satélites, sin embargo, diversos errores afectan esta medición y por lo tanto se afecta el cálculo.

La precisión en el posicionamiento que un usuario puede alcanzar con un receptor, depende principalmente de cómo sus sistemas de hardware y software recojan y procesen los diversos errores que afectan a la medición.

2.1.5.1. Retraso de la señal debido a la ionósfera y atmósfera

El GPS utiliza ondas de radio en las transmisiones que deben atravesar sucesivamente el vacío, la ionósfera y la tropósfera.

Las ondas de radio viajan a la velocidad de la luz, que normalmente se supone un parámetro inmutable y fijo de valor igual a 300 000 Km/s. Sin embargo, esta es la

velocidad de propagación de la luz en el vacío, cuando se propaga en otro medio existe un cierto retardo que, aunque pudiera parecer muy pequeño, afecta de una manera importante a las precisas mediciones efectuadas por los sistemas GPS [12].

En la parte más alta de la atmósfera terrestre, las radiaciones ultravioleta y de rayos X procedentes del sol interactúan con las moléculas de gas y los átomos. Estas interacciones producen la ionización del gas: un gran número de electrones libres cargados negativamente y átomos y moléculas con carga positiva [12]. Tal región de la atmósfera, donde la ionización del gas tiene lugar se llama la ionósfera, esta se extiende desde una altura aproximada de 50 km a 1000 km ó incluso más. De hecho, el límite superior de la región de la ionósfera no está claramente definido.

La densidad de electrones en la región de la ionósfera no es constante, sino que cambia con la altitud, pudiéndose clasificar en capas, ver tabla número 2.1. La altura y el grosor de esas capas varían con el tiempo, como resultado de los cambios de la radiación del Sol y del campo magnético de la Tierra. Por ejemplo, la capa conocida como F1 desaparece durante la noche y es más pronunciada en el verano que en el invierno.

Capa de la ionósfera	Altura aproximada desde la superficie terrestre en Km.
D	50 - 90
E	90 - 140
F1	140 - 210
F2	210 - 1000

Tabla 2.1. Clasificación de las capas de la Ionósfera.

La ionósfera dobla la señal de radio del GPS y cambia su velocidad, a medida que pasa a través de las capas de ésta para llegar a un receptor GPS. Doblar el camino de la señal GPS causa un error en la distancia insignificante, particularmente si el ángulo de elevación del satélite es mayor que 5°. Es el cambio en la velocidad de propagación lo que causa un error más significativo en la distancia del receptor al satélite, y por lo tanto, debe tenerse en cuenta. El retraso debido a la ionósfera es proporcional al número de electrones libres a lo largo de la trayectoria de la señal GPS, sin embargo, este depende de una serie de factores:

1. **La hora del día**, el nivel de densidad de electrones llega a un máximo en la tarde y un mínimo alrededor de la medianoche en hora local.
2. **Las estaciones del año**, los niveles de densidad electrónica son más altos en invierno que en verano.

3. **El ciclo solar de 11 años**, los niveles de densidad electrónica alcanzan un valor máximo aproximadamente cada 11 años, lo que corresponde a un pico en las actividades de erupción solar conocidos como el pico del ciclo solar.
4. **La ubicación geográfica**, los niveles de densidad electrónica son mínimos en las regiones de latitudes medias y muy irregulares en las regiones polares y ecuatoriales.

El problema del retraso tiene una solución bastante efectiva, todo se basa en el conocimiento de que el retraso de las señales GPS es inversamente proporcional al cuadrado de las frecuencias de las mismas, es decir, cuanto menor sea la frecuencia (y por tanto mayor la longitud de onda), mayor será el retraso.

Teniendo esto en cuenta, si se emiten desde el satélite dos señales con frecuencias distintas en un instante determinado, podemos examinar, una vez que llegan al receptor, el desfase existente entre dos partes de dichas señales que, teóricamente, deberían haber llegado al mismo tiempo. De esta forma puede calcularse con bastante aproximación el retraso debido a la ionósfera (partiendo de las frecuencias de ambas señales y del desfase medido entre ambas en la recepción) [9].

Por esta razón, se diseñó el sistema con las portadoras L1 y L2 anteriormente mencionadas.

Para poder aplicar estas correcciones necesitaremos un receptor que sea capaz de leer las portadoras L1 y L2. A este tipo de receptores se les llama de “*doble frecuencia*”, y su precio es bastante superior a los de una sola. Una solución para los usuarios con una sola frecuencia de GPS, es utilizar correcciones de redes regionales. Tales correcciones pueden ser recibidas en tiempo real a través de diversas redes de comunicación.

Una vez atravesada la ionósfera, queda todavía la tropósfera, en la cual las fuentes de error más importantes son la variación de temperatura del aire seco y la presencia de vapor de agua. La primera tiene mucha mayor influencia (alrededor del 90%), pero puede ser predicha con un alto grado de exactitud usando modelos matemáticos [9], con lo que se eliminaría el error cometido por este factor. Aunque la influencia del vapor de agua es mucho menor, es muy difícil determinar la distribución del mismo en la tropósfera, y por tanto corregir esta fuente de imprecisión.

Existen unos aparatos que pueden medir la cantidad de vapor de agua en una determinada dirección. Se llaman radiómetros de vapor de agua y se utilizan solamente en circunstancias excepcionales en las que se requiera máxima precisión.

2.1.5.2. Otras fuentes de error

Podríamos destacar la imprecisión de los relojes, tanto en los satélites (muy pequeña, pero existente), como en los receptores. Ya hemos visto que la imprecisión del reloj de tierra puede eliminarse utilizando una cuarta determinación. Los relojes atómicos de los satélites son controlados regularmente por el Departamento de Defensa de los Estados Unidos, efectuando las correcciones oportunas.

Otro error de cierta importancia es el conocido como “*multipath*” o “*multisenda*”, que se produce cuando las señales no van directamente del satélite al receptor, sino que se desvían y sufren varios rebotes antes de alcanzar su objetivo. Los receptores GPS actuales incorporan diversos mecanismos para evitar este fenómeno y minimizar su influencia. Este efecto puede ser considerablemente reducido eligiendo puntos de estación protegidos de reflexiones (edificios, vehículos, árboles, etc.), es decir, evitar las superficies reflectantes en las proximidades del receptor y un apropiado diseño de la antena, en la figura 2.6 se puede apreciar una ilustración que describe este efecto.

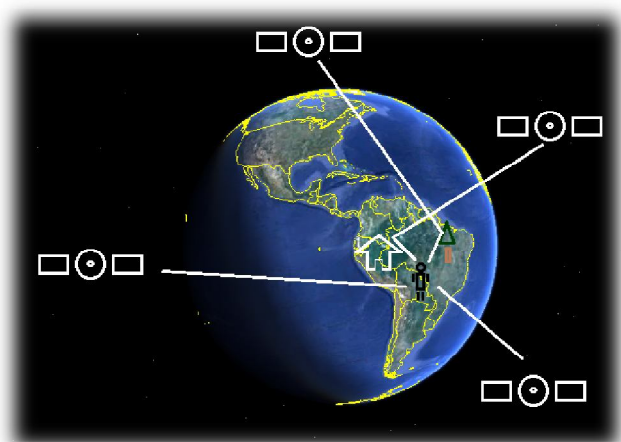


Figura 2.6. Efecto *multipath*.

2.1.5.3. Dilución de la Precisión Geométrica (GDOP)

La precisión final obtenida no solamente depende de que tengamos a nuestra disposición un cierto número de satélites, sino también de que estos tengan una situación espacial favorable en cuanto a determinaciones geométricas. La figura 2.7 y 2.8 muestran esto para el caso de dos dimensiones.

Si los dos satélites están en una posición ventajosa (figura 2.7), tomando al receptor como vértice y trazando una línea imaginaria hacia los satélites, desde el punto de vista del receptor se pueden ver en un ángulo de aproximadamente 90° entre sí. Debido a que la medición de la distancia a los satélites no se puede determinar absolutamente con

precisión, como se explicó anteriormente, las posiciones posibles para ello están marcadas por los círculos grises. El punto de intersección A de los dos círculos es pequeña, el cuadro azul es el área en donde puede ubicarse el receptor, por lo tanto la posición determinada será bastante precisa.

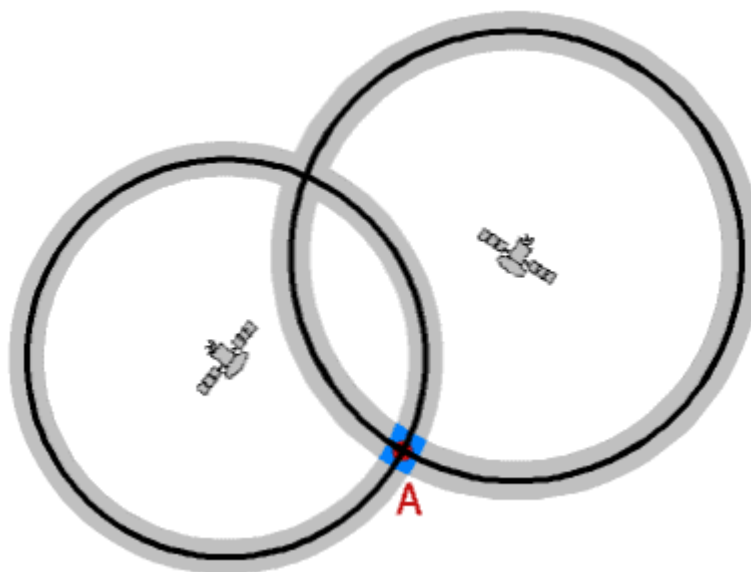


Figura 2.7. Una buena alineación geométrica de dos satélites.

Si los satélites son más o menos posicionados en una línea desde el punto de vista del receptor (figura 2.8), el plano de intersección de las posiciones posibles es considerablemente más grande y alargado, en consecuencia, la determinación de la posición es menos precisa.

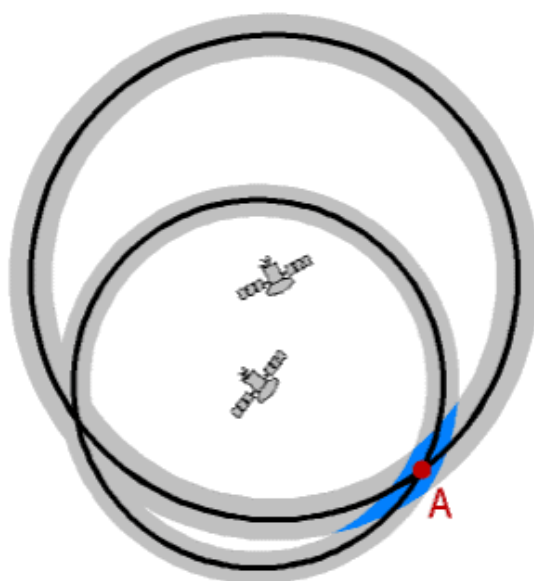


Figura 2.8. Una mala alineación geométrica de dos satélites.

Valor de Parámetro GDOP	Clasificación
< 1	Ideal
1 - 2	Excelente
2 - 5	Buena
5 - 10	Moderada
10 - 20	Razonable
> 20	Pobre

Tabla 2.2. Clasificación del valor GDOP.

Tabla 2.2. Clasificación del valor GDOP.

A circular diagram representing a celestial map. It features a central red dot, four concentric black circles, and a horizontal line with 'W' on the left and 'E' on the right. A vertical line passes through the center, with 'S' at the bottom. There are 10 colored dots: 8 green and 2 yellow. The dots are distributed across the concentric circles and quadrants.



nica mediante la cual el Departamento de degradar la precisión del método S provocar errores de 100 y 156 metros mente.

A esta técnica se la ha denominado SA (*Selective Availability*, o Disponibilidad Selectiva) y se aplicó por primera vez en Marzo de 1990, como consecuencia del descubrimiento de

que el SPS ofrecía precisiones mucho mayores de lo que se esperaba (casi tan grandes como las del PPS).

La SA introduce 2 tipos de error [13]: El primero, llamado "*error delta*" el cual consiste en variar el tiempo real del satélite, y con ello no obtener una buena medición de distancia desde el receptor hasta el satélite. El segundo, llamado "*error épsilon*", es un error adicional que consiste en variar la información de la órbita real del satélite. Con la SA activada, los errores nominales en la horizontal y vertical pueden ser de hasta 100 metros y 156 metros, respectivamente, en el 95% de las muestras.

Después de extensos estudios, los EE.UU. suspendió la SA el 1 de mayo de 2000, resultando en una mejor precisión del GPS [11]. Con la SA desactivada, el GPS obtiene una precisión nominal en la horizontal y vertical del orden de 22 metros y 33 metros o menos en el 95% de las muestras respectivamente (figura 2.10).

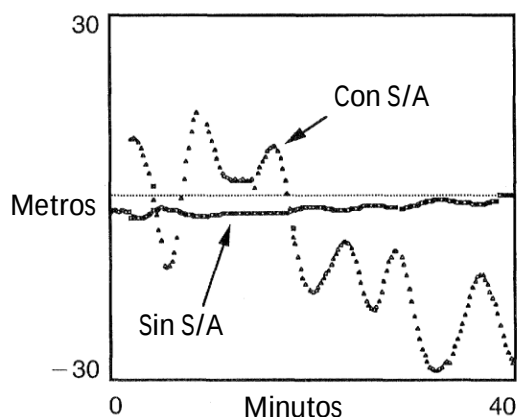


Figura 2.10 Error en la medición con la SA apagada y activa [13].

Actualmente, la SA puede ser aplicada sin previo aviso, lo que podría originar errores de gran importancia en las mediciones.

2.1.6. Elipsoide y Geoide, Sistemas de Coordenadas y Proyecciones Cartográficas

La capacidad de GPS para determinar la ubicación precisa de un usuario en cualquier lugar, ha atraído a millones de usuarios de todo el mundo de diversos campos. Un problema común es entender la información que nos arroja el GPS o la lectura del mapa en donde nos ubica. A continuación se trata el problema de los puntos de referencia y los sistemas de coordenadas.

2.1.6.1. Elipsoide

Debido a que la superficie de la Tierra es muy irregular los cálculos para determinar una posición son complicados. Para mantener los cálculos simples y obtener una alta precisión en el sistema GPS, un elipsoide es la superficie matemática que más se adapta a la forma real de la Tierra.

Existen diferentes modelos de elipsoides utilizados en geodesia, denominados elipsoides de referencia. Las diferencias entre éstos vienen dadas por los valores asignados a sus parámetros más importantes:

- ❖ **Semieje ecuatorial (a) o Semieje mayor:** Longitud del semieje correspondiente al ecuador, desde el centro de masa de la Tierra hasta la superficie terrestre.
- ❖ **Semieje polar (b) o Semieje menor:** Longitud del semieje desde el centro de masa de la Tierra hasta uno de los polos. Alrededor de este eje se realiza la rotación de la elipse base.

Un elipsoide se obtiene por rotación de una elipse alrededor de su eje menor “b”. Similar a la elipse, la elipsoide de referencia puede ser definido por sus semiejes mayor y menor (a, b) o por el semieje mayor y el aplanamiento (a, f), donde $f = 1 - \left(\frac{b}{a}\right)$, este último representa que tan diferentes son los semiejes de una elipse entre sí.

Existen numerosos elipsoides que sirven de referencia a diversos sistemas geodésicos diferentes. La tabla 2.3 muestra algunos ejemplos de tres sistemas de referencia comunes y sus elipsoides asociadas [8].

Sistema de referencia	Elipsoide	Semieje mayor (a) en metros	Factor de aplanamiento (f)
WGS 84	WGS 84	6 378 137	298.257 223 563
NAD 83	GRS 80	6 378 137	298.257 222 101
NAD 27	Clarke 1866	6 378 206.4	294.978 698 2

Tabla 2.3. Algunos sistemas de referencia y su elipsoide asociada.

2.1.6.2. Geoide

Dado que la mayor parte de la Tierra está cubierta por mares y océanos, la superficie de referencia por excelencia para medir altitudes es el nivel medio del mar.

El nivel medio del mar depende de las irregularidades en el campo gravitatorio de la Tierra, que se forma debido a que en el planeta existen zonas con diferente densidad,

estas variaciones alteran su posición. El agua de los océanos busca estar en equilibrio y, por ello, tienden a seguir una superficie gravitatoria equipotencial.

Es por esto que se introduce una nueva figura llamada Geoide, que se define como la superficie equipotencial del campo gravitatorio de la Tierra que mejor se ajusta al nivel medio global del mar (figura 2.11).

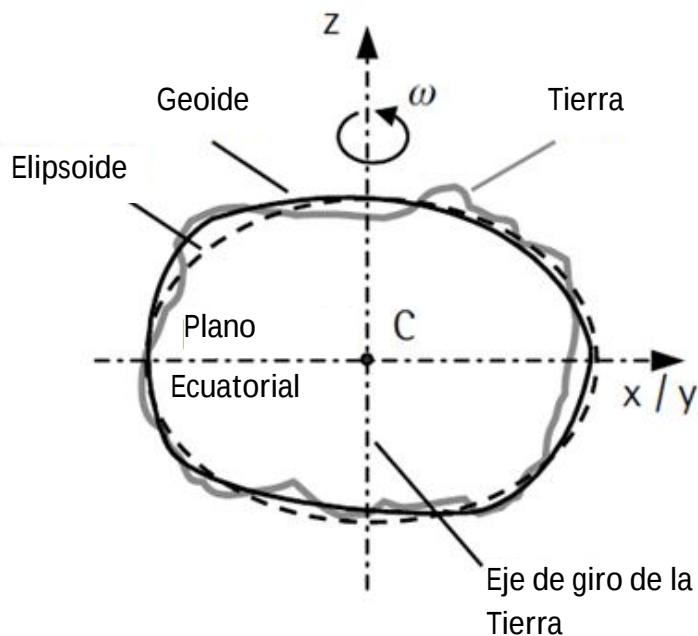


Figura 2.11. Relación entre la superficie de la Tierra, el geoide y el elipsoide.

2.1.6.3. Sistemas de Coordenadas Geodésicas

Un sistema de coordenadas se define como un conjunto de reglas para la especificación de la localización de puntos (también llamadas coordenadas). Esto generalmente implica especificar un origen, así como un conjunto de líneas de referencia (llamado ejes) con una orientación conocida.

Los sistemas de coordenadas geodésicos pueden clasificarse de acuerdo a la superficie de referencia, la orientación de los ejes y el origen. En el caso de un sistema de coordenadas 3D geodésico (también conocido como geográfico), la superficie de referencia se selecciona para que sea el elipsoide. La orientación de los ejes y el origen se especifican por dos planos: el plano meridiano a través del eje z (un meridiano es un plano que pasa a través de los polos norte y sur) y el plano ecuatorial del elipsoide (véase la Figura 2.12).

En un sistema 3D geodésico, las coordenadas de un punto se identifican por la latitud geodésica (ϕ), la longitud geodésica (λ), y la altura sobre la superficie de referencia (h). La

Figura 2.12 muestra estos parámetros. Es posible transformar las coordenadas geodésicas (ϕ, λ) en una rejilla rectangular de coordenadas para fines de mapeo.

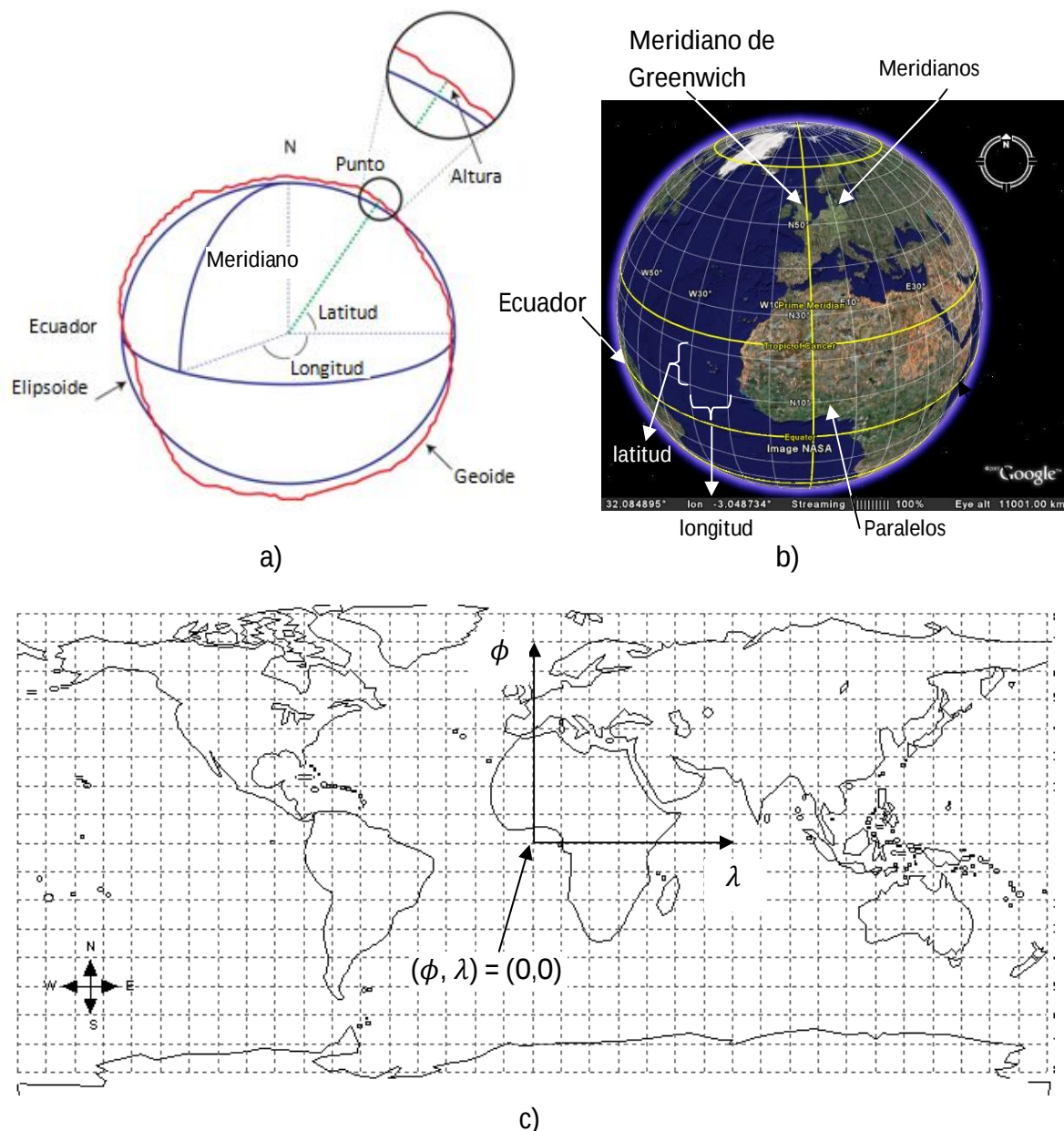


Figura 2.12. Concepto de coordenadas geodésicas a) latitud y longitud, b) meridianos y paralelos terrestres, c) Mapeo en base a la latitud y longitud terrestre.

El sistema de referencia terrestre convencional es un sistema de coordenadas geocéntricas en tercera dimensión, su origen coincide con el centro de la Tierra.

La orientación de los ejes del sistema de referencia terrestre convencional se define como: El eje "z" hacia el polo norte terrestre convencional (CTP), que se define como el promedio de posición del polo durante el período 1900 - 1905 [14]. El eje "x" está definido por la

intersección del plano ecuatorial terrestre y el plano meridional que contiene la ubicación media del observatorio de Greenwich. El eje "y" se selecciona de acuerdo al sistema de coordenadas de la mano derecha. Los tres ejes se cortan en el centro de la Tierra.

2.1.6.4. El sistema WGS 84 y diversas proyecciones cartográficas

El Sistema de Posicionamiento Global (GPS), necesita disponer de un sistema que ayude a situar una posición geográfica con referencia a un elipsoide universal. Para ello fue creado el Sistema Geodésico Mundial WGS (*World Geodetic System*), denominado WGS 74. El sistema de referencia que actualmente se utiliza es el WGS 84.

El WGS 84 es un sistema de referencia terrestre convencional, se trata de un sistema de referencia geocéntrico fijo con la Tierra y orientado positivamente, desarrollado por la antigua Agencia de Cartografía de Defensa de EE.UU. ahora incorporada a una nueva agencia: la Agencia Nacional de Imágenes y Cartografía (NIMA). Es el sistema oficial de referencia GPS, en otras palabras, un usuario de GPS obtendrá sus coordenadas en el sistema WGS 84, sin embargo un usuario puede ajustar su sistema GPS, de tal manera que los datos sean presentados en otro sistema, realizándose las transformaciones pertinentes [8]. Los parámetros que utiliza el WGS 84 se describen en la tabla 2.3.

La proyección hacia un mapa se define, desde el punto de vista geométrico, como la transformación de las características físicas de la superficie terrestre curvada hacia una superficie plana llamada mapa. Sin embargo, desde el punto de vista matemático, se define, como la transformación de coordenadas geodésicas (ϕ , λ) obtenido a partir de, por ejemplo, GPS, en una rejilla de coordenadas rectangulares, esto se conoce como la proyección del mapa en directa.

Desafortunadamente, debido a la diferencia entre la forma de la Tierra y la superficie de proyección plana, las características proyectadas sufren una distorsión [15]. Un gran número de tipos de proyección han sido desarrolladas para minimizar estas distorsiones. Las proyecciones más populares son "*Transversal de Mercator*", "*Transversal de Mercator Universal*" (UTM) y "*La proyección cónica de Lambert*", cada tipo de proyección debe de ser acompañado con el sistema de referencia, por ejemplo el sistema de referencia WGS 84 [15].

2.1.6.5. Conversión de las Coordenadas Geográficas a Metros

En el apartado 2.1.6.3 se definieron lo que se conocen como "coordenadas geodésicas" y en el apartado 2.1.6.4 se indicó que la superficie de referencia para el sistema GPS es la elipsoide adoptada en el sistema WGS 84, la intención es, conociendo las coordenadas de latitud y longitud de un punto sobre la tierra, proporcionada por un receptor GPS,

averiguar cuál es la distancia desde el origen hasta este punto en metros, considerando el origen a las coordenadas de latitud igual a cero y longitud igual a cero, figura 2.12.

Existen dos formatos en los cuales un GPS nos puede presentar una ubicación de un punto sobre la tierra, las coordenadas geográficas (GPS) se suelen expresar en forma de grados con decimales y en forma de grados minutos y segundos.

A continuación se presenta el método para pasar de grados con decimales a grados con minutos y segundos.

Teniendo 0.123456° , ¿Cuál es su representación en grados con minutos y segundos?

- ❖ Grados: Se obtiene directamente tomando la parte entera de la coordenada decimal. De 0.123456° , los grados serían 0° .
- ❖ Minutos: Son la parte entera del resultado de multiplicar la parte decimal por 60. De 0.123456° , la parte decimal es 0.123456, entonces multiplicamos 0.123456 por 60, lo que resulta en 7.40736, por tanto la parte entera serían los minutos, en este caso 7'.
- ❖ Segundos: Se toma la parte decimal del resultado de la multiplicación anterior y se multiplica de nuevo por 60, el resultado son los segundos, en este caso se toma 0.40736 y se multiplica por 60, resultando en 24.4416, estos serían los minutos.

Con el procedimiento anterior tenemos que 0.123456° es igual a $0^\circ 7' 24''$, que se lee; 0 grados, 7 minutos y 24 segundos.

Para el procedimiento contrario, es decir, pasar de grados con minutos y segundos a grados con decimales, se procede de la siguiente manera:

- ❖ Grados: Se obtiene tomando los grados directamente. De $0^\circ 7' 24''$, los grados serían 0.
- ❖ Decimales: Se toman los minutos y se dividen entre 60, el resultado se suma a lo que resulte de tomar los segundos y dividirlos entre 3600. De $0^\circ 7' 24''$, se toma 7 y se divide entre 60, lo que es igual a 0.11666, posteriormente se toma el 24 y se divide entre 3600, lo que resulta en 0.00666, entonces los decimales estarían formados por la suma de 0.11666 más 0.00666, lo que da 0.12332.

Con el procedimiento anterior tenemos que $0^\circ 7' 24''$ es igual a 0.12332° .

Con los dos procedimientos anteriormente descritos, es fácil pasar de un formato de coordenadas a otro, sin embargo, en ocasiones es necesario conocer las coordenadas en de un punto sobre la tierra en metros o kilómetros, por ejemplo, **conociendo el valor de**

un grado de latitud y longitud en metros, basta con multiplicar dichos valores por la coordenada, respectivamente, en formato de grados con decimales para conocer la distancia desde el origen del sistema hasta el punto ubicado con el GPS.

En la figura 2.12, inciso "b", se puede apreciar que dependiendo de la latitud o de que tan alejado se ubique un punto del ecuador, las distancias de un grado de latitud y longitud van cambiando, debido a la forma elipsoidal de la tierra. Si se desea saber la distancia que representa un grado de latitud, distancias verticales, se debe considerar que los grados de latitud están espaciados regularmente, sin embargo, el ligero achatamiento de la Tierra en los polos causa que un grado de latitud varíe de 110,57 km en el ecuador hasta 111,70 km en los polos. Se suele redondear un grado de latitud a 111,12 km. Para la longitud, distancia horizontal, en la figura 2.12, inciso "b", es posible observar que mientras más alejado se encuentre del ecuador, un grado de longitud se hace cada vez más pequeño, [8].

La tabla 2.4 muestra la distancia que representa un grado de latitud y longitud, en función de la latitud positiva, se considera una elipse simétrica con respecto al ecuador, es posible notar que la distancia de la longitud es la que sufre una mayor diferencia dependiendo de qué tan alejado se encuentre del ecuador, sin embargo, la distancia que representa cada grado de latitud tiene una menor variación [8].

Latitud	Distancia en metros por grado de:	
	Latitud	longitud
90	111693.9173	6.839
80	111659.8799	19393.477
70	111561.9698	38186.525
60	111412.2402	55799.979
50	111229.0264	71695.7261
40	111034.6052	85393.826
30	110852.4248	96486.2475
20	110704.2781	104647.0531
10	110607.7602	109639.331
0	110574.2727	111319.458
-50	111229.0264	71695.7261
-90	111693.9173	6.839

Tabla 2.4. Distancia que representa un grado de latitud y de longitud en función de la latitud

2.2. Robótica Móvil

La palabra "robot", del checo "robota", que significa "trabajo forzado", fue utilizada por primera vez en una comedia en el año de 1921, llamada *RUR (Robots Universales Rossum)*, una obra teatral de ciencia ficción, escrita por Karel Čapek.

La robótica se ocupa del estudio de las máquinas que pueden reemplazar a los seres humanos en la ejecución de una tarea, tanto en la actividad física como en la toma de decisiones.

Los robots más utilizados hoy en día son manipuladores, que opera en un espacio de trabajo limitado y no pueden moverse. Un robot móvil es diferente, este puede cambiar su posición a través de la locomoción.

2.2.1. Locomoción en Robótica Móvil

La característica principal de los robots móviles es la presencia de una base móvil que permite que el robot se mueva libremente en el medio ambiente. A diferencia de los manipuladores, estos tipos de robots se utilizan principalmente en aplicaciones de servicio, donde tener capacidad de movimiento y en algunos casos una gran autonomía es necesario. Desde un punto de vista mecánico, un robot móvil se compone de uno o más cuerpos rígidos equipados con un sistema de locomoción. Esta descripción incluye las siguientes dos clases principales de robots móviles:

- ❖ **Robots móviles con extremidades;** están hechos de múltiples cuerpos rígidos, conectados a través de articulaciones. Algunos de estos cuerpos rígidos forman las extremidades inferiores, útiles como medio de locomoción. Hay una gran variedad de estructuras mecánicas en esta clase, cuya frecuencia de diseño se inspira en el estudio de los organismos vivos, van desde hexápodos, como insectos, o bípedos, como el hombre.
- ❖ **Robots móviles con llantas;** estos consisten en un cuerpo rígido (base o chasis) y un sistema de llantas que proporcionan movimiento con respecto al suelo. Otros cuerpos rígidos (remolques), también equipados con llantas, pueden estar conectado a la base por medio de articulaciones.

Algunas configuraciones de los robots móviles con llantas o ruedas pueden ser consideradas como una simplificación de los automóviles usados actualmente, es por ello que se usan como prototipo para así poder probar el sistema de navegación que se desea desarrollar.

2.2.2. Cinemática de Robots Móviles con Llantas

Calcular la cinemática del robot móvil es el primer paso para entender su comportamiento mecánico, gracias a este análisis, es posible diseñar de forma apropiada al robot, tomando en cuenta las tareas que debe de realizar y también nos permite entender cómo crear el software de control, tomando en cuenta su hardware.

2.2.2.1. Posición del Robot

Para poder modelar el robot se considera que es un cuerpo rígido y que su movimiento se realiza sobre un plano horizontal. Con el fin de especificar la posición del robot, definimos dos planos de referencia, uno global y uno local sobre el mismo robot, como se muestra en la figura 2.13.

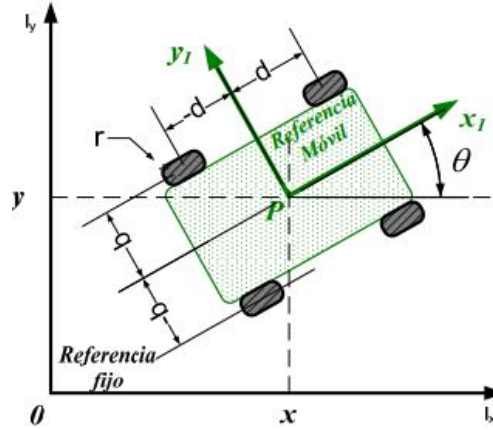


Figura 2.13. Posición del robot sobre un marco de referencia global, [5].

El marco de referencia global está definido por los ejes " l_x " y " l_y " ortogonales entre ellos mismos. Para especificar la posición del robot sobre el plano de referencia global, se elige un punto P , sobre el chasis del robot, como punto de referencia de su posición. Sobre el punto P se define dos ejes " x_1 " y " y_1 ", ortogonales entre ellos, formando el eje de referencia local. La posición de P en el marco de referencia global está especificado por las coordenadas " x " e " y ", y la diferencia angular entre el marco de referencia global y local está dada por θ . Podemos describir la postura del robot con estos tres elementos dentro del vector ε_I . Observe el uso del subíndice " I " para aclarar que esta postura esta daba desde el marco de referencia global [5].

$$\varepsilon_I = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \dots \dots \dots 2.6$$

En la ecuación 2.6 se puede observar el vector ε_I , donde " x " y " y " son las coordenadas del punto " P " sobre el marco de referencia global y θ es el ángulo formado por el eje longitudinal del robot, que coincide con el e eje " x_1 " del marco de referencia local, y el eje l_x del sistema de referencia global o fijo. La derivada de ε_I con respecto del tiempo es $\dot{\varepsilon}_I$.

$$\dot{\varepsilon}_I = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \dots \dots \dots 2.7$$

Donde " $\dot{x}$ " y " $\dot{y}$ " son las componentes de la velocidad lineal y " $\dot{\theta}$ " es la velocidad angular del robot.

El vector ε_I contiene la posición y orientación del punto "P", es decir, contiene la posición y la orientación que se desea que el robot tenga, midiendo, dichos componentes, desde el marco global o fijo de referencia. Para describir el movimiento del robot en términos del marco de referencia local, o viceversa, será necesario realizar una transformación de una base a otra, esto se logra empleando la matriz de rotación.

2.2.2.1.1. Matriz de Rotación

En la figura 2.14 podemos observar dos marcos de referencia, el marco 1, definido por los ejes " X_1 ", " Y_1 " y " Z_1 ", y el marco de referencia 2, definido por los ejes " X_2 ", " Y_2 " y " Z_2 ". Supongamos que se tiene localizado un punto en términos del marco de referencia 1, la matriz de rotación permite localizar a este mismo punto en términos del marco de referencia 2, realizando una simple multiplicación matricial como se puede apreciar en la ecuación 2.8, a continuación se definirá esta matriz de rotación $R(\alpha, \beta, \gamma)$.

$$\begin{bmatrix} P_{X2} \\ P_{Y2} \\ P_{Z2} \end{bmatrix} = R(\alpha, \beta, \gamma) \begin{bmatrix} P_{X1} \\ P_{Y1} \\ P_{Z1} \end{bmatrix} \dots \dots \dots 2.8$$

Ecuación 1. Localización de un punto del marco de referencia 1 al marco de referencia 2.

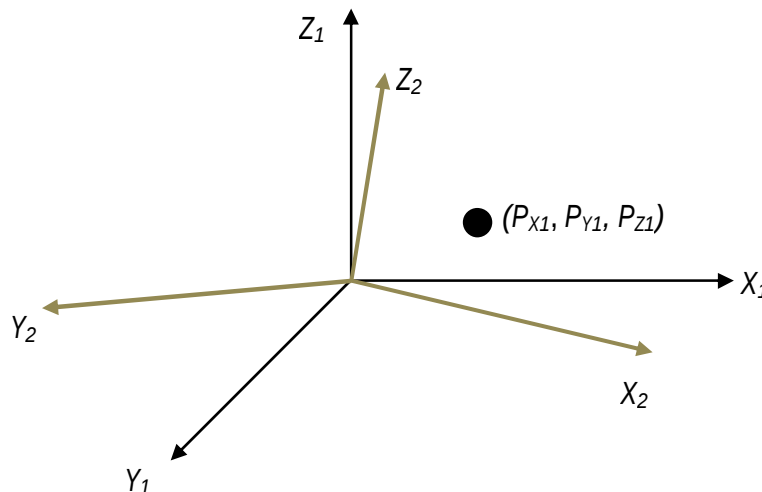


Figura 2.14. Marco de referencia 2 rotado.

En la figura 2.14, se puede ver como el sistema de referencia 2 se encuentra rotado con respecto al marco de referencia 1, para encontrar la matriz de rotación que asocia a estos dos marcos de referencia, realizaremos una rotación con un ángulo θ alrededor del eje " Z_1 ", como se puede apreciar en la figura 2.15, delimitando un nuevo marco de referencia; pero recordando que se encuentra rotado en un ángulo θ alrededor del eje " Z_1 ".

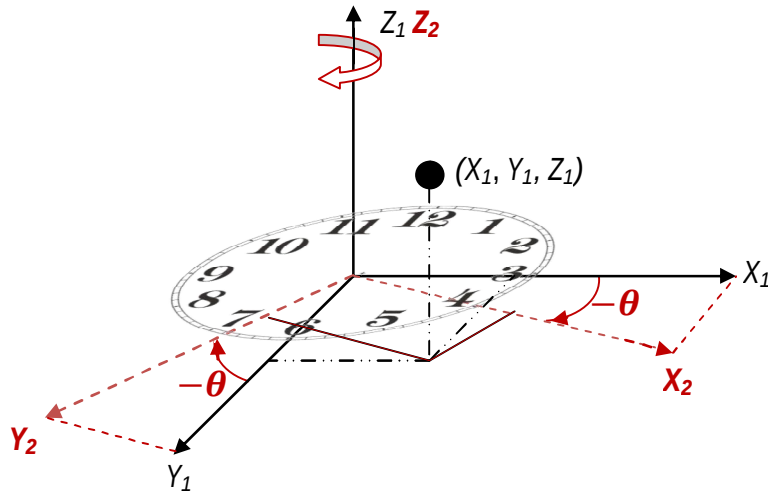


Figura 2.15. Marco de referencia rotado alrededor de Z1 en sentido horario.

En la figura 2.15 podemos ver que se aplicó una rotación alrededor de "Z1" en sentido horario, esto ocasiona que "Z2" se encuentre alineado con "Z1", el punto "P" puede localizarse en los dos sistemas de referencia, pero existe una matriz, llamada matriz de rotación, con la que se puede pasar de un sistema a otro. Para la matriz de rotación asociada a la figura 2.15, podemos trasladarnos al plano, de tal manera que sea más sencillo encontrarla.

En la figura 2.16, podemos ver la rotación en el plano, visto desde este punto de vista el ángulo "θ" debería ser positivo, sin embargo, debemos de recordar que el sistema fue rotado considerando el eje "z" y por tal motivo es negativo, [18].

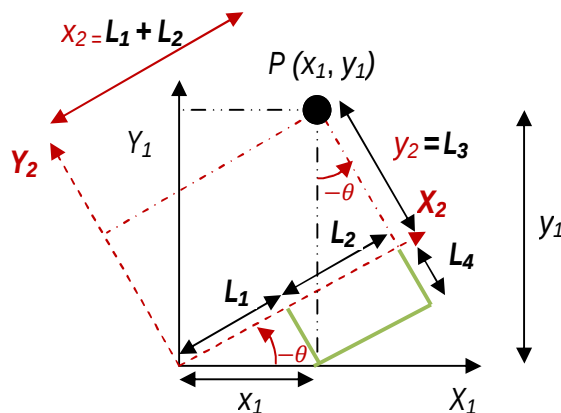


Figura 2.16. Vista en el plano de rotación alrededor del eje Z.

El punto "P" está referenciado al marco formado por los ejes "X1" y "Y1", la intención es encontrar la matriz para hallar las coordenadas del punto "P" en el sistema de referencia formado por los ejes "X2" y "Y2". En la figura 2.16 se puede apreciar las siguientes relaciones:

$$x_2 = L_1 + L_2 \dots \dots \dots 2.9$$

$$y_2 = L_3 \dots \dots \dots 2.10$$

Para encontrar L_1 nos apoyamos con el triángulo de líneas continuas resaltado en la figura 2.17, y para encontrar L_2 y L_3 , usamos el triángulo resaltado con líneas discontinúas en la misma figura.

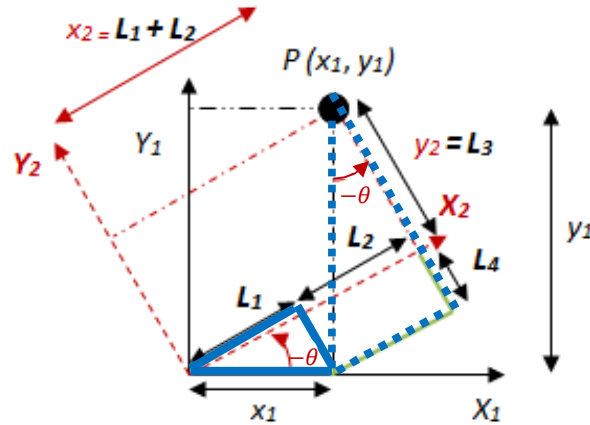


Figura 2.17. Cálculo de las coordenadas del punto "P" en el sistema 2.

En el triángulo con líneas continuas:

$$\cos(-\theta) = \cos(\theta) = \frac{L_1}{x_1}$$

$$L_1 = x_1 \cos(\theta) \dots \dots \dots 2.11$$

En el triángulo con líneas discontinúas:

$$\sen(-\theta) = -\sen(\theta) = \frac{L_2}{y_1}$$

$$L_2 = -y_1 \sen(\theta) \dots \dots \dots 2.12$$

$$\cos(-\theta) = \cos(\theta) = \frac{L_3 + L_4}{y_1}$$

$$L_3 = y_1 \cos(\theta) - L_4 \dots \dots \dots 2.13$$

Para L_4 , de nuevo nos fijamos en el triángulo con líneas continuas:

$$\sen(-\theta) = -\sen(\theta) = \frac{L_4}{x_1}$$

$$L_4 = -x_1 \text{sen}(\theta) \dots \dots \dots 2.14$$

Sustituimos 2.14 en 2.13:

$$L_3 = y_1 \cos(\theta) - (-x_1 \text{sen}(\theta))$$

$$L_3 = y_1 \cos(\theta) + x_1 \text{sen}(\theta) \dots \dots \dots 2.15$$

Sustituimos 2.11 y 2.12 en 2.9 y 2.15 en 2.10

$$x_2 = x_1 \cos(\theta) - y_1 \text{sen}(\theta) \dots \dots \dots 2.16$$

$$y_2 = x_1 \text{sen}(\theta) + y_1 \cos(\theta) \dots \dots \dots 2.17$$

Para el eje Z, tenemos:

$$z_2 = x_1(0) + y_1(0) + z_1 \dots \dots \dots 2.15$$

Tomando en cuenta lo anterior podemos modificar las ecuaciones 2.16 y 2.17, para que tomen en cuenta el eje z:

$$x_2 = x_1 \cos(\theta) - y_1 \text{sen}(\theta) + z_1(0)$$

$$y_2 = x_1 \text{sen}(\theta) + y_1 \cos(\theta) + z_1(0)$$

$$z_2 = x_1(0) + y_1(0) + z_1$$

Reordenando estas últimas ecuaciones en una matriz:

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\text{sen}(\theta) & 0 \\ \text{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$$

$$r_2 = R(\theta) r_1 \dots \dots \dots 2.16$$

Donde:

$$R(\theta) = \begin{bmatrix} \cos(\theta) & -\text{sen}(\theta) & 0 \\ \text{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \dots \dots \dots 2.17$$

La matriz $R(\theta)$ es la **matriz de rotación alrededor del eje Z** y, la ecuación 2.16 es la operación de rotación que transforma al vector del sistema de coordenadas 1 al sistema rotado de coordenadas 2 [18].

En este caso sólo hemos rotado el sistema de coordenadas 2 alrededor del eje "Z" un ángulo θ , pero la figura 2.14, hace referencia a un sistema rotado también alrededor del

eje "Y" un ángulo β y alrededor del eje "X" un γ , estas dos últimas rotaciones se pueden apreciar en las figuras 2.18 y 2.19, respectivamente y sus matrices de rotación correspondientes son las 2.18 y 2.19, [18].

$$R(\beta) = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \dots\dots\dots 2.18$$

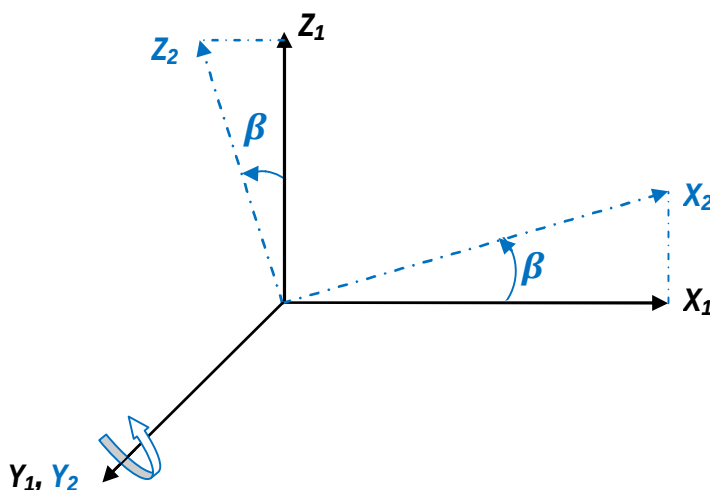


Figura 2.18. Rotación alrededor del eje "Y" un ángulo β .

$$R(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) \\ 0 & \sin(\gamma) & \cos(\gamma) \end{bmatrix} \dots\dots\dots 2.19$$

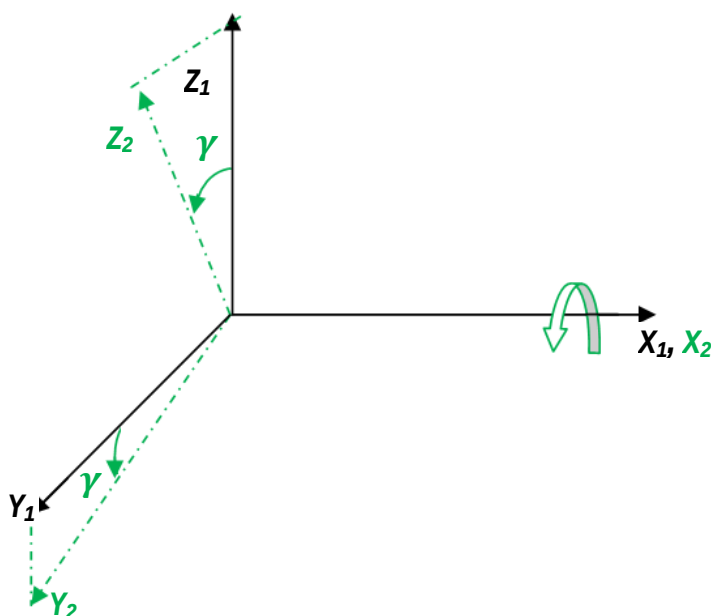


Figura 2.19. Rotación alrededor del eje "X" un ángulo γ .

Para tener una matriz de rotación que determine una rotación en cada eje, basta con realizar una composición de rotaciones, multiplicando las matrices dadas por las ecuaciones 2.17, 2.18 y 2.19, [18].

$$R(\theta, \beta, \gamma) = R(\gamma)R(\beta)R(\theta) \dots \dots \dots 2.20$$

La ecuación 2.20 quiere decir que se realizó una rotación primero alrededor del eje "z" un ángulo " θ ", posteriormente alrededor del eje "y" un ángulo " β " y por último alrededor del eje "x" un ángulo " γ ".

$$R(\theta, \beta, \gamma) = \begin{bmatrix} C(\beta)C(\theta) & -C(\beta)S(\theta) & S(\beta) \\ S(\gamma)S(\beta)C(\theta) + C(\gamma)S(\alpha) & S(\theta)S(\gamma)S(\beta) + C(\gamma)C(\theta) & -S(\gamma)C(\beta) \\ -C(\gamma)S(\beta)C(\theta) + S(\gamma)S(\theta) & C(\gamma)S(\beta)S(\theta) + C(\theta)S(\gamma) & C(\gamma)C(\beta) \end{bmatrix} \quad 2.21$$

Donde:

$$S(\theta) = \text{sen}(\theta)$$

$$S(\beta) = \text{sen}(\beta)$$

$$S(\gamma) = \text{sen}(\gamma)$$

$$C(\alpha) = \text{cos}(\alpha)$$

$$C(\beta) = \text{cos}(\beta)$$

$$C(\gamma) = \text{cos}(\gamma)$$

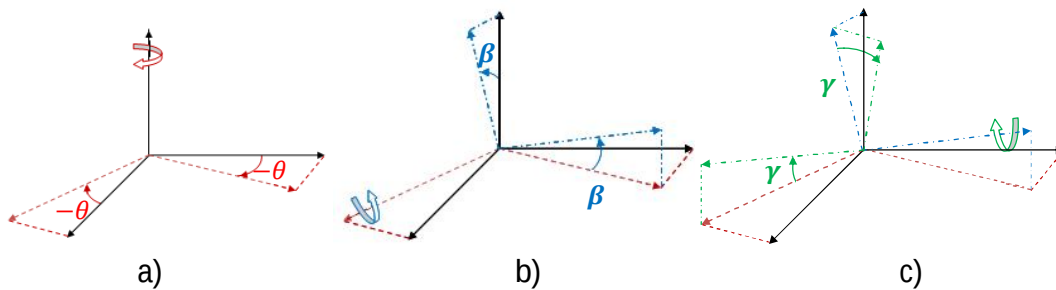


Figura 2.20. Composición de rotaciones. a) Rotación alrededor del eje Z, b) Rotación alrededor del eje Y c) Rotación alrededor del eje x.

Las matrices 2.17, 2.18 y 2.19, en el caso de una sola rotación alrededor del eje "z", "y" o "x" respectivamente o la matriz 2.21, en el proceso de que la primer rotación sea alrededor del eje "z", posteriormente alrededor del eje "y" y, por último alrededor del eje "x", pueden ser utilizadas para mapear los movimientos del robot del marco de referencia global al marco de referencia local. Considerando sólo una rotación alrededor del eje "z", esta operación es denotada por la ecuación 2.22.

$$\dot{\epsilon}_R = R(\theta)\dot{\epsilon}_I \dots \dots \dots 2.22$$

Para el proceso contrario, ubicar al robot del marco de referencia local al marco de referencia global, sólo es necesario encontrar la inversa de la matriz de rotación a utilizar, considerando sólo una rotación alrededor del eje "z", esta operación es denotada por la ecuación 2.23.

$$R(\theta)^{-1}\dot{\varepsilon}_R = R(\theta)^{-1}R(\theta)\dot{\varepsilon}_I$$

$$\dot{\varepsilon}_I = R(\theta)^{-1}\dot{\varepsilon}_R \dots \dots \dots 2.23$$

Considerando que las matrices de rotación son matrices ortogonales, la matriz inversa $R(\theta)^{-1}$ es igual a la matriz transpuesta de $R(\theta)$, es decir $R(\theta)^{-1} = R(\theta)^T$, entonces la ecuación 2.23 la podemos escribir como la ecuación 2.24.

$$\dot{\varepsilon}_I = R(\theta)^T \dot{\varepsilon}_R \dots \dots \dots 2.24$$

Donde:

$$R(\theta)^T = R(\theta)^{-1} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

La ecuación 2.24 ubica al robot en todo momento en el sistema de referencia global o fija, [18].

2.2.2.2. Llanta Fija, Centrada y Descentrada Orientable, Sueca y Esférica

El elemento mecánico básico de estos robots son las llantas y, el proceso de entender los movimientos de un robot comienza con el proceso de describir la contribución y restricción que proporciona cada llanta para el movimiento. Cada tipo de llanta impone limitaciones sobre el movimiento del robot, por ejemplo, impidiendo el derrape lateral.

Hay cinco clases principales de llantas, como se muestra en la tabla 2.5. Estas difieren en su cinemática y, por lo tanto la elección del tipo de llanta tiene un gran efecto sobre la cinemática general del robot móvil, [16].

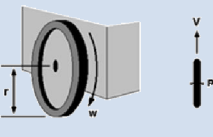
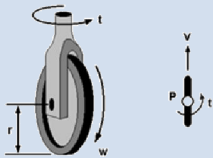
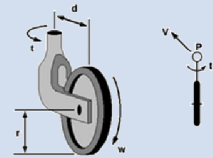
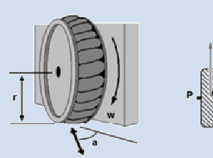
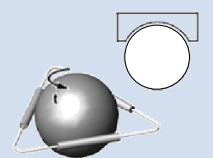
Llantas o ruedas				
Fija	Centrada Orientable	Descentrada Orientable (castor)	Sueca o omnidireccional	Esférica o de bola
				

Tabla 2.5. Los cinco tipos de llantas o ruedas.

Para efectos del presente trabajo, no se profundizará en las llantas suecas ni esféricas, para mayor información se puede consultar el documento [17] en la bibliografía. En el caso de las llantas fijas, centradas y descentradas orientables tenemos la consideración de que el contacto entre la llanta y el suelo es en un solo punto, existe rodamiento puro y sin deslizamiento.

2.2.2.2.1. Llanta Fija y sus Restricciones Cinemáticas

Las llantas fijas son aquellas que no cambian su posición vertical angular con respecto al marco de referencia del robot, es decir, solamente tiene libertad para girar sobre su eje horizontal. En la figura 2.21 se aprecia al punto "P" del que parte un sistema de referencia móvil, el punto "A" está definido por las coordenadas (d,b) , el ángulo " α " es la orientación con respecto al eje paralelo a " x_1 ", en este caso es constante. El movimiento angular de la llanta sobre su eje horizontal es " φ ", su velocidad angular es " $\dot{\varphi}$ " y su radio es " r ", con las últimas dos variables es posible determinar la velocidad lineal de la llanta, [16].

$$v_{Ll} = \dot{\varphi} * r \dots \dots \dots 2.25$$

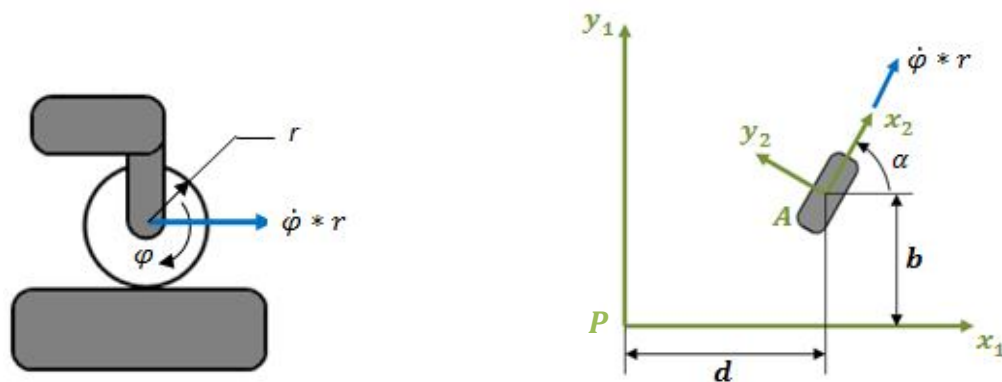


Figura 2.21. Llantas fijas y orientables centradas, [16].

Su construcción impone que todo movimiento fuera del eje de la rueda sea cero, debido a que no permite deslizamiento hacia los lados, en cambio, siempre que el movimiento tenga la misma dirección " α " de la rueda esta rodará sin fricción, idealmente, en la figura 2.22, se ve que se aplica una velocidad en el eje " x_1 " y en el eje " y_1 ", estas velocidades son las dos primeras componentes del vector 2.22, presentado anteriormente. Para propósito de análisis, estas velocidades se trasladan a la rueda colocándolas sobre el sistema de referencia ortonormal local de la rueda, formado por los ejes " x_2 " y " y_2 ", posteriormente se determinan sus componentes dentro de este marco de referencia para poder observar cómo actúa la rueda.

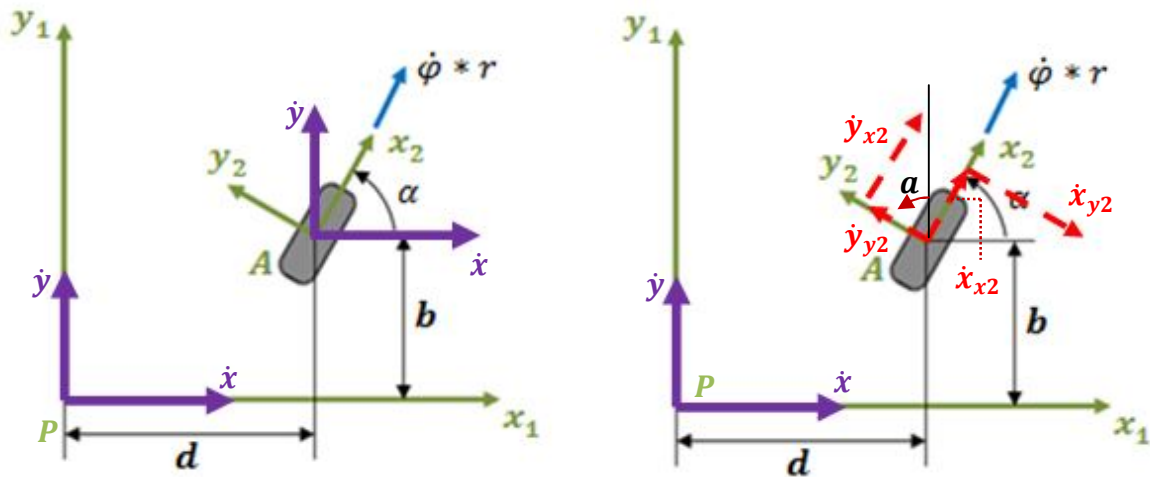


Figura 2.22. Llanta fija y sus restricciones cinemáticas.

A continuación se analizan las velocidades involucradas en la rueda. De la segunda imagen de la figura 2.22, podemos obtener las siguientes relaciones:

$$\cos(\alpha) = \frac{\dot{x}_{x2}}{\dot{x}}$$

$$\dot{x}_{x2} = \dot{x} \cos(\alpha) \dots \dots \dots 2.26$$

$$\cos(\alpha) = \frac{\dot{y}_{y2}}{\dot{y}}$$

$$\dot{y}_{y2} = \dot{y} \cos(\alpha) \dots \dots \dots 2.27$$

$$\sin(\alpha) = \frac{-\dot{x}_{y2}}{\dot{x}}$$

$$\dot{x}_{y2} = -\dot{x} \sin(\alpha) \dots \dots \dots 2.28$$

$$\sin(\alpha) = \frac{\dot{y}_{x2}}{\dot{y}}$$

$$\dot{y}_{x2} = \dot{y} \sin(\alpha) \dots \dots \dots 2.29$$

Calculando las velocidades de la llanta, debido a la velocidad lineal del robot:

Velocidades sobre el eje "x₂":

$$v_x = \dot{x}_{x2} + \dot{y}_{x2} \dots \dots \dots 2.30$$

Velocidades sobre el eje "y₂":

$$v_y = \dot{y}_{y2} + \dot{x}_{y2} \dots \dots \dots 2.31$$

Sustituyendo 2.26 y 2.29 en 2.30, asimismo 2.27 y 2.28 en 2.31.

$$v_x = \dot{x}\cos(\alpha) + \dot{y}\sin(\alpha) \dots \dots \dots 2.32$$

$$v_y = \dot{y}\cos(\alpha) + -\dot{x}\sin(\alpha) \dots \dots \dots 2.33$$

En la figura 2.23, se observa una velocidad angular dada por $\dot{\theta}$, la velocidad lineal producida en la llanta está dada por la multiplicación de esta velocidad angular, por su brazo de palanca. La figura 2.23, se utiliza para poder analizar la velocidad que posee la rueda debido a la velocidad angular " $\dot{\theta}$ " del robot.

En la figura 2.23, se puede observar que los brazos de palanca se encuentran formados por " d " y " b ", facilitando mucho el análisis, conociendo que la rueda no permite deslizamiento lateral, sólo las componentes sobre el eje " x_2 " se utilizarán para que la llanta ruede.

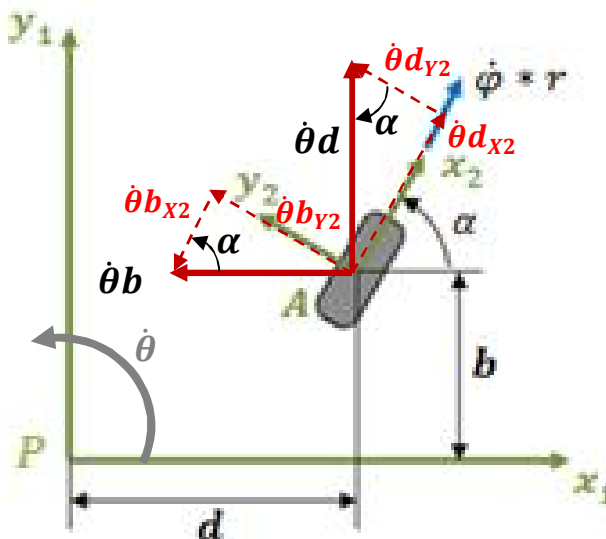


Figura 2.23. Llanta fija y sus restricciones cinemáticas en la velocidad angular.

Definiendo proyecciones:

$$\cos(\alpha) = \frac{\dot{\theta}d_{y2}}{\dot{\theta}d}$$

$$\dot{\theta}d_{y2} = \dot{\theta}d\cos(\alpha) \dots \dots \dots 2.34$$

$$\text{sen}(\alpha) = \frac{\dot{\theta} d_{x2}}{\dot{\theta} d}$$

$$\dot{\theta} d_{x2} = \dot{\theta} d \text{sen}(\alpha) \dots \dots \dots 2.35$$

$$\text{cos}(\alpha) = \frac{\dot{\theta} b_{x2}}{\dot{\theta} b}$$

$$\dot{\theta} b_{x2} = \dot{\theta} b \text{cos}(\alpha) \dots \dots \dots 2.36$$

$$\text{sen}(\alpha) = \frac{\dot{\theta} b_{y2}}{\dot{\theta} b}$$

$$\dot{\theta} b_{y2} = \dot{\theta} b \text{sen}(\alpha) \dots \dots \dots 2.37$$

Calculando las velocidades de la llanta debido a la velocidad angular:

Velocidades sobre el eje "x₂":

$$v_{x_rot} = \dot{\theta} d_{x2} - \dot{\theta} b_{x2} \dots \dots \dots 2.38$$

Velocidades sobre el eje "y₂":

$$v_{y_rot} = \dot{\theta} d_{y2} + \dot{\theta} b_{y2} \dots \dots \dots 2.39$$

Sustituyendo 2.35 y 2.36 en 2.38, asimismo 2.34 y 2.37 en 2.39:

$$v_{x_rot} = \dot{\theta} d \text{sen}(\alpha) - \dot{\theta} b \text{cos}(\alpha) \dots \dots \dots 2.40$$

$$v_{y_rot} = \dot{\theta} d \text{cos}(\alpha) + \dot{\theta} b \text{sen}(\alpha) \dots \dots \dots 2.41$$

Realizando simplificaciones.

$$v_{x_rot} = \dot{\theta} (d * \text{sen}(\alpha) - b * \text{cos}(\alpha)) \dots \dots \dots 2.42$$

$$v_{y_rot} = \dot{\theta} (d * \text{cos}(\alpha) + b * \text{sen}(\alpha)) \dots \dots \dots 2.43$$

La velocidad total de llanta está compuesta por su velocidad en el eje "x₂" y "y₂". Sobre el eje "x₂" la rueda no opone resistencia:

$$v_x + v_{x_rot} = \dot{\varphi} * r$$

$$\dot{x} \text{cos}(\alpha) + \dot{y} \text{sen}(\alpha) + \dot{\theta} (d * \text{sen}(\alpha) - b * \text{cos}(\alpha)) = \dot{\varphi} * r \dots \dots \dots 2.44$$

Sobre el eje "y₂" la rueda no desliza ni gira, por lo tanto su velocidad es cero:

$$v_y + v_{y_rot} = 0$$

$$\dot{y}\cos(\alpha) + -\dot{x}\sin(\alpha) + \dot{\theta}(d * \cos(\alpha) + b * \sin(\alpha)) = 0 \dots \dots \dots 2.45$$

Las ecuaciones 2.44 y 2.45 representan las restricciones cinemáticas de la llanta en su plano de giro horizontal y en el plano ortogonal de la rueda respectivamente, la ecuación 2.45 dice que su velocidad es cero debido a que no está permitido el deslizamiento de la llanta. Estas ecuaciones pueden escribirse de la siguiente manera correspondientemente:

$$[\cos(\alpha) \quad \sin(\alpha) \quad d\sin(\alpha) - b\cos(\alpha)] \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} - \dot{\phi} * r = 0$$

$$[\cos(\alpha) \quad \sin(\alpha) \quad d\sin(\alpha) - b\cos(\alpha)]\dot{\epsilon}_R - \dot{\phi} * r = 0 \dots \dots \dots 2.46$$

$$[-\sin(\alpha) \quad \cos(\alpha) \quad d\cos(\alpha) + b\sin(\alpha)] \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = 0$$

$$[-\sin(\alpha) \quad \cos(\alpha) \quad d\cos(\alpha) + b\sin(\alpha)]\dot{\epsilon}_R = 0 \dots \dots \dots 2.47$$

Sustituyendo 2.22 en las ecuaciones 2.46 y 2.47.

$$[\cos(\alpha) \quad \sin(\alpha) \quad d\sin(\alpha) - b\cos(\alpha)]R(\theta)\dot{\epsilon}_I - \dot{\phi} * r = 0 \dots \dots \dots 2.48$$

$$[-\sin(\alpha) \quad \cos(\alpha) \quad d\cos(\alpha) + b\sin(\alpha)]R(\theta)\dot{\epsilon}_I = 0 \dots \dots \dots 2.49$$

Donde "b", "d", "r" y el ángulo "α" son constantes, el término "φ" es variante en el tiempo. También es importante recordar que ε_I esta en términos de la base global o fija.

2.2.2.2.2. Llanta centrada orientable y sus Restricciones Cinemáticas

Las llantas centradas orientables tienen un comportamiento similar a las fijas, sólo que estas contienen un eje vertical de rotación y son altamente direccionales. Así, para moverse en una dirección diferente, la llanta debe ser dirigida a lo largo de un eje vertical. La ventaja es que, debido a que el eje de rotación vertical pasa a través del punto de contacto con el suelo, la rotación en su eje vertical no causa efectos secundarios en el robot.

En la figura 2.21 se puede apreciar esta rueda, el ángulo "α" es la orientación con respecto al eje paralelo a "x₁", en este caso es variable. Su velocidad lineal está dado por la ecuación 2.25.

Las restricciones cinemáticas son muy similares a las de la llanta fija. Las ecuaciones 2.50 y 2.51 representan las restricciones cinemáticas de la llanta en su plano de giro horizontal y en el plano ortogonal de la rueda respectivamente.

$$[\cos(\alpha) \quad \sin(\alpha) \quad d\sin(\alpha) - b\cos(\alpha)]R(\theta)\dot{\epsilon}_I - \dot{\varphi} * r = 0 \dots \dots \dots 2.50$$

$$[-\sin(\alpha) \quad \cos(\alpha) \quad d\cos(\alpha) + b\sin(\alpha)]R(\theta)\dot{\epsilon}_I = 0 \dots \dots \dots 2.51$$

Donde "b", "d", "r" son constantes, los términos " $\dot{\varphi}$ " y " α " son ángulos variantes en el tiempo. También es importante recordar que $\dot{\epsilon}_I$ esta en términos de la base global o fija.

2.2.2.2.3. Llanta descentrada orientable y sus Restricciones Cinemáticas

Las llantas descentradas orientables o castor, también conocidas como ruedas locas, contienen las mismas características que las llantas centradas orientables, sólo que en este caso, el eje de giro vertical no cruza con el eje vertical de la llanta, teniendo la desventaja que, al ser direccionadas, causan una fuerza que puede tener impacto sobre el chasis del robot. En la figura 2.24, se aprecia al punto "P" del que parte un sistema de referencia móvil, el punto "A" está definido por las coordenadas (d, b), al ángulo " α " es la orientación con respecto al eje paralelo a " x_1 ", en este caso es variable. A diferencia de las anteriores, existe una distancia "e" entre los puntos "A" y "B". El movimiento angular de la llanta sobre su eje horizontal es " φ ", su velocidad angular es " $\dot{\varphi}$ " y su radio es "r", con las últimas dos variables es posible determinar la velocidad lineal de la llanta [16].

$$v_{LL} = \dot{\varphi} * r \dots \dots \dots 2.52$$

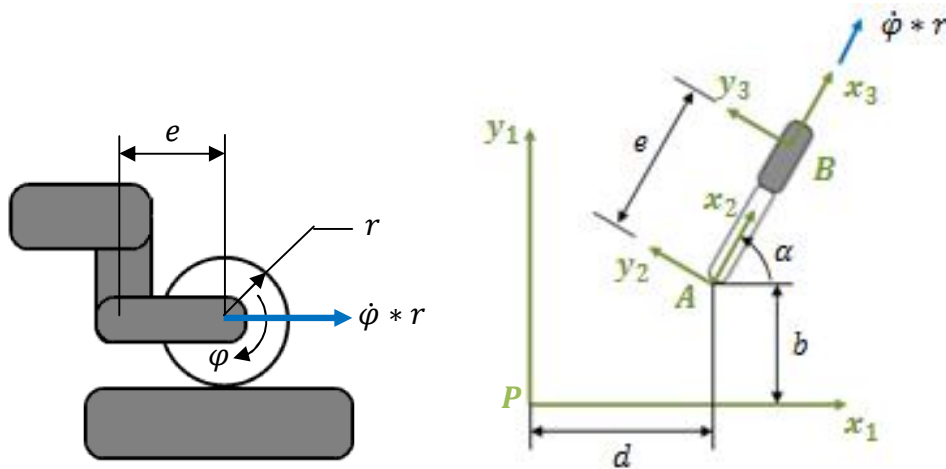


Figura 2.24. Llantas orientables descentradas, [16].

Las restricciones cinemáticas son muy similares a las de la llanta centrada orientable. Las ecuaciones 2.53 y 2.54 representan las restricciones cinemáticas de la llanta en su plano de giro horizontal y en el plano ortogonal de la rueda respectivamente.

$$[\cos(\alpha) \quad \sin(\alpha) \quad d\sin(\alpha) - b\cos(\alpha)]R(\theta)\dot{\epsilon}_I - \dot{\varphi} * r = 0 \dots \dots \dots 2.53$$

$$[-\sin(\alpha) \quad \cos(\alpha) \quad e + d\cos(\alpha) + b\sin(\alpha)]R(\theta)\dot{\epsilon}_I - e\dot{\alpha} = 0 \dots \dots \dots 2.54$$

Donde " b ", " d ", " r " y " e " son constantes, los términos " $\dot{\varphi}$ " y " α " son ángulos variantes en el tiempo, el término " $e\dot{\alpha}$ " permite la característica omnidireccional de la llanta. También es importante recordar que $\dot{\epsilon}_l$ esta en términos de la base global o fija.

2.2.2.2.4. Llanta Sueca u omnidireccional

Llanta sueca u omnidireccional; están menos limitadas en su direccionalidad, al proporcionar una baja resistencia en otra dirección, siendo esta variable dependiente de la arquitectura de la llanta, por ejemplo la llanta sueca a 45° proporciona una resistencia baja en un ángulo de 45°, de manera similar una llanta sueca a 90°, estas se pueden observar en la figura 2.25. Los rodillos pequeños fijados alrededor de la circunferencia de la llanta son pasivos y el eje primario de la llanta sirve como la articulación activa o motorizada. La principal ventaja de este diseño es que, aunque la rotación de la llanta es alimentada sólo a lo largo del eje principal, la rueda puede moverse, con muy poca fricción, a lo largo de muchas posibles trayectorias, no sólo hacia adelante y hacia atrás.

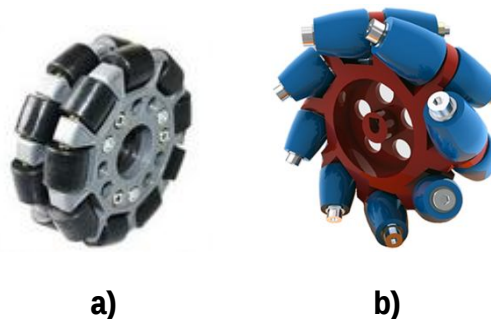


Figura 2.25. a) Llanta sueca a 90°, b) Llanta sueca a 45°.

2.2.2.2.5. Rueda Esférica o de Bola

La rueda esférica o de bola también es una rueda omnidireccional, diseñada de modo que puede ser impulsada y girar en cualquier dirección, con un mecanismo similar al usado en el "mouse" de la computadora, a través de rodillos accionados que descansan en la superficie superior de la esfera, también es utilizada, en el caso contrario, como punto de apoyo, otorgando estabilidad al robot, en este sentido la rueda sería pasiva.

2.2.2.3. Restricciones Cinemáticas del robot Móvil

Conociendo las restricciones cinemáticas de las ruedas, ahora es posible calcular las restricciones cinemáticas del chasis del robot, debido a que cada rueda impone cero o más restricciones en el movimiento del robot. Para esto se necesita colocar cada restricción cinemática de cada llanta que tiene el robot en un arreglo matricial.

Se han clasificado las llantas en fijas, centradas orientables, descentradas orientables, suecas y esféricas, sin embargo, sólo las dos primeras tienen impacto en la cinemática del robot móvil ya que las otras tres son omnidireccionales.

Suponiendo que el robot tiene un total de N ruedas, de las cuales N_f son llantas fijas y N_s llantas centradas orientables, es decir $N = N_s + N_f$, cada llanta fija tiene un ángulo α_f y cada llanta centrada orientable tiene un ángulo variable α_s , además cada una tendrá una velocidad angular, para las ruedas fijas, estas velocidades angulares estarán asignadas por $\dot{\varphi}_f(t)$ y para las centradas orientables, sus velocidades angulares estarán representadas por $\dot{\varphi}_s(t)$, ver figura 2.21.

Entonces, se tiene el siguiente arreglo de ecuaciones para las restricciones cinemáticas de las llantas en su plano de giro horizontal:

Para las Llantas fijas:

$$\begin{aligned} [\cos(\alpha_{f1}) \quad \sin(\alpha_{f1}) \quad d\sin(\alpha_{f1}) - b\cos(\alpha_{f1})] R(\theta) \dot{\epsilon}_l - r_1 * \dot{\varphi}_{f1}(t) &= 0 \dots \text{Llanta fija 1} \\ [\cos(\alpha_{f2}) \quad \sin(\alpha_{f2}) \quad d\sin(\alpha_{f2}) - b\cos(\alpha_{f2})] R(\theta) \dot{\epsilon}_l - r_2 * \dot{\varphi}_{f2}(t) &= 0 \dots \text{Llanta fija 2} \\ [\cos(\alpha_{f3}) \quad \sin(\alpha_{f3}) \quad d\sin(\alpha_{f3}) - b\cos(\alpha_{f3})] R(\theta) \dot{\epsilon}_l - r_3 * \dot{\varphi}_{f3}(t) &= 0 \dots \text{Llanta fija 3} \\ &\vdots \\ [\cos(\alpha_{fNf}) \quad \sin(\alpha_{fNf}) \quad d\sin(\alpha_{fNf}) - b\cos(\alpha_{fNf})] R(\theta) \dot{\epsilon}_l - r_{Nf} * \dot{\varphi}_f(t) &= 0 \dots \text{Llanta fija } N_f \end{aligned}$$

En matrices:

$$\begin{bmatrix} \cos(\alpha_{f1}) & \sin(\alpha_{f1}) & d\sin(\alpha_{f1}) - b\cos(\alpha_{f1}) \\ \cos(\alpha_{f2}) & \sin(\alpha_{f2}) & d\sin(\alpha_{f2}) - b\cos(\alpha_{f2}) \\ \vdots & \ddots & \vdots \\ \cos(\alpha_{fNf}) & \sin(\alpha_{fNf}) & d\sin(\alpha_{fNf}) - b\cos(\alpha_{fNf}) \end{bmatrix} R(\theta) \dot{\epsilon}_l - \begin{bmatrix} r_1 & 0 & 0 \\ 0 & r_2 & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & r_{Nf} \end{bmatrix} \begin{bmatrix} \dot{\varphi}_{f1}(t) \\ \dot{\varphi}_{f2}(t) \\ \vdots \\ \dot{\varphi}_{fNf}(t) \end{bmatrix} = 0$$

$$\mathbf{J}_{1f} \mathbf{R}(\theta) \dot{\epsilon}_l - \mathbf{J}_{2f} \dot{\varphi}_f(t) = \mathbf{0} \dots \dots \dots \mathbf{2.55}$$

Donde:

$$\mathbf{J}_{1f} = \begin{bmatrix} \cos(\alpha_{f1}) & \sin(\alpha_{f1}) & d\sin(\alpha_{f1}) - b\cos(\alpha_{f1}) \\ \cos(\alpha_{f2}) & \sin(\alpha_{f2}) & d\sin(\alpha_{f2}) - b\cos(\alpha_{f2}) \\ \vdots & \ddots & \vdots \\ \cos(\alpha_{fNf}) & \sin(\alpha_{fNf}) & d\sin(\alpha_{fNf}) - b\cos(\alpha_{fNf}) \end{bmatrix} \quad \mathbf{J}_{2f} = \begin{bmatrix} r_1 & 0 & 0 \\ 0 & r_2 & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & r_{Nf} \end{bmatrix}$$

$$\dot{\varphi}_f(t) = \begin{bmatrix} \dot{\varphi}_{f1}(t) \\ \dot{\varphi}_{f2}(t) \\ \vdots \\ \dot{\varphi}_{fNf}(t) \end{bmatrix}$$

Para las llantas centradas orientables:

$$\begin{aligned}
 [\cos(\alpha_{s1}) \quad \text{sen}(\alpha_{s1}) \quad \text{dsen}(\alpha_{s1}) - b\cos(\alpha_{s1})]R(\theta)\dot{\epsilon}_I - r_1 * \dot{\varphi}_{s1}(t) &= 0 \dots \text{Llanta centrada orientable 1} \\
 [\cos(\alpha_{s2}) \quad \text{sen}(\alpha_{s2}) \quad \text{dsen}(\alpha_{s2}) - b\cos(\alpha_{s2})]R(\theta)\dot{\epsilon}_I - r_2 * \dot{\varphi}_{s2}(t) &= 0 \dots \text{Llanta centrada orientable 2} \\
 [\cos(\alpha_{s3}) \quad \text{sen}(\alpha_{s3}) \quad \text{dsen}(\alpha_{s3}) - b\cos(\alpha_{s3})]R(\theta)\dot{\epsilon}_I - r_3 * \dot{\varphi}_{s3}(t) &= 0 \dots \text{Llanta centrada orientable 3} \\
 &\vdots \\
 [\cos(\alpha_{sNs}) \quad \text{sen}(\alpha_{sNs}) \quad \text{dsen}(\alpha_{sNs}) - b\cos(\alpha_{sNs})]R(\theta)\dot{\epsilon}_I - r_{Ns} * \dot{\varphi}_{sNs}(t) &= 0 \dots \text{Llanta centrada orientable } N_s
 \end{aligned}$$

En matrices:

$$\begin{bmatrix} \cos(\alpha_{s1}) & \text{sen}(\alpha_{s1}) & \text{dsen}(\alpha_{s1}) - b\cos(\alpha_{s1}) \\ \cos(\alpha_{s2}) & \text{sen}(\alpha_{s2}) & \text{dsen}(\alpha_{s2}) - b\cos(\alpha_{s2}) \\ \vdots & \ddots & \vdots \\ \cos(\alpha_{sNs}) & \text{sen}(\alpha_{sNs}) & \text{dsen}(\alpha_{sNs}) - b\cos(\alpha_{sNs}) \end{bmatrix} R(\theta)\dot{\epsilon}_I - \begin{bmatrix} r_1 & 0 & 0 \\ 0 & r_2 & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & r_{Ns} \end{bmatrix} \begin{bmatrix} \dot{\varphi}_{s1}(t) \\ \dot{\varphi}_{s2}(t) \\ \vdots \\ \dot{\varphi}_{sNs}(t) \end{bmatrix} = 0$$

$$J_{1s}(\alpha_s)R(\theta)\dot{\epsilon}_I - J_{2Ns}\dot{\varphi}_{Ns}(t) = 0 \dots \dots \dots \mathbf{2.56}$$

Donde:

$$J_{1s}(\alpha_s) = \begin{bmatrix} \cos(\alpha_{s1}) & \text{sen}(\alpha_{s1}) & \text{dsen}(\alpha_{s1}) - b\cos(\alpha_{s1}) \\ \cos(\alpha_{s2}) & \text{sen}(\alpha_{s2}) & \text{dsen}(\alpha_{s2}) - b\cos(\alpha_{s2}) \\ \vdots & \ddots & \vdots \\ \cos(\alpha_{sNs}) & \text{sen}(\alpha_{sNs}) & \text{dsen}(\alpha_{sNs}) - b\cos(\alpha_{sNs}) \end{bmatrix} \quad J_{2Ns} = \begin{bmatrix} r_1 & 0 & 0 \\ 0 & r_2 & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & r_{Ns} \end{bmatrix}$$

$$\dot{\varphi}_{Ns}(t) = \begin{bmatrix} \dot{\varphi}_{s1}(t) \\ \dot{\varphi}_{s2}(t) \\ \vdots \\ \dot{\varphi}_{sNs}(t) \end{bmatrix}$$

De igual manera, se tiene el siguiente arreglo de ecuaciones para las restricciones cinemáticas de las llantas en su plano ortogonal:

Para las Llantas fijas:

$$\begin{aligned}
 [-\text{sen}(\alpha_{f1}) \quad \cos(\alpha_{f1}) \quad \text{dcos}(\alpha_{f1}) + b\text{sen}(\alpha_{f1})]R(\theta)\dot{\epsilon}_I &= 0 \dots \text{Llanta fija 1} \\
 [-\text{sen}(\alpha_{f2}) \quad \cos(\alpha_{f2}) \quad \text{dcos}(\alpha_{f2}) + b\text{sen}(\alpha_{f2})]R(\theta)\dot{\epsilon}_I &= 0 \dots \text{Llanta fija 2} \\
 [-\text{sen}(\alpha_{f3}) \quad \cos(\alpha_{f3}) \quad \text{dcos}(\alpha_{f3}) + b\text{sen}(\alpha_{f3})]R(\theta)\dot{\epsilon}_I &= 0 \dots \text{Llanta fija 3} \\
 &\vdots \\
 [-\text{sen}(\alpha_{fNf}) \quad \cos(\alpha_{fNf}) \quad \text{dcos}(\alpha_{fNf}) + b\text{sen}(\alpha_{fNf})]R(\theta)\dot{\epsilon}_I &= 0 \dots \text{Llanta fija } N_f
 \end{aligned}$$

En matrices:

$$\begin{bmatrix} -\text{sen}(\alpha_{f1}) & \cos(\alpha_{f1}) & \text{dcos}(\alpha_{f1}) + b\text{sen}(\alpha_{f1}) \\ -\text{sen}(\alpha_{f2}) & \cos(\alpha_{f2}) & \text{dcos}(\alpha_{f2}) + b\text{sen}(\alpha_{f2}) \\ \vdots & \ddots & \vdots \\ -\text{sen}(\alpha_{fNf}) & \cos(\alpha_{fNf}) & \text{dcos}(\alpha_{fNf}) + b\text{sen}(\alpha_{fNf}) \end{bmatrix} R(\theta)\dot{\epsilon}_I = 0$$

$$C_{1f}R(\theta)\dot{\epsilon}_I = 0 \dots \dots \dots \mathbf{2.57}$$

Donde:

$$C_{1S}(\alpha_S) = \begin{bmatrix} -\text{sen}(\alpha_{f1}) & \cos(\alpha_{f1}) & d\cos(\alpha_{f1}) + -b\text{sen}(\alpha_{f1}) \\ -\text{sen}(\alpha_{f2}) & \cos(\alpha_{f2}) & d\cos(\alpha_{f2}) + b\text{sen}(\alpha_{f2}) \\ \vdots & \ddots & \vdots \\ -\text{sen}(\alpha_{fNf}) & \cos(\alpha_{fNf}) & d\cos(\alpha_{fNf}) + b\text{sen}(\alpha_{fNf}) \end{bmatrix}$$

Para las llantas centradas orientables:

$$\begin{aligned} [-\text{sen}(\alpha_{S1}) \quad \cos(\alpha_{S1}) \quad d\cos(\alpha_{S1}) + b\text{sen}(\alpha_{S1})]R(\theta)\dot{\epsilon}_I &= 0 \dots \text{Llanta centrada orientable 1} \\ [-\text{sen}(\alpha_{S2}) \quad \cos(\alpha_{S2}) \quad d\cos(\alpha_{S2}) + b\text{sen}(\alpha_{S2})]R(\theta)\dot{\epsilon}_I &= 0 \dots \text{Llanta centrada orientable 2} \\ [-\text{sen}(\alpha_{S3}) \quad \cos(\alpha_{S3}) \quad d\cos(\alpha_{S3}) + b\text{sen}(\alpha_{S3})]R(\theta)\dot{\epsilon}_I &= 0 \dots \text{Llanta centrada orientable 3} \\ &\vdots \\ [-\text{sen}(\alpha_{SNS}) \quad \cos(\alpha_{SNS}) \quad d\cos(\alpha_{SNS}) + b\text{sen}(\alpha_{SNS})]R(\theta)\dot{\epsilon}_I &= 0 \dots \text{Llanta centrada orientable } N_S \end{aligned}$$

En matrices:

$$\begin{bmatrix} -\text{sen}(\alpha_{S1}) & \cos(\alpha_{S1}) & d\cos(\alpha_{S1}) + b\text{sen}(\alpha_{S1}) \\ -\text{sen}(\alpha_{S2}) & \cos(\alpha_{S2}) & d\cos(\alpha_{S2}) + b\text{sen}(\alpha_{S2}) \\ \vdots & \ddots & \vdots \\ -\text{sen}(\alpha_{SNS}) & \cos(\alpha_{SNS}) & d\cos(\alpha_{SNS}) + b\text{sen}(\alpha_{SNS}) \end{bmatrix} R(\theta)\dot{\epsilon}_I = 0$$

$$C_{1S}(\alpha_S)R(\theta)\dot{\epsilon}_I = 0 \dots \dots \dots \mathbf{2.58}$$

Donde:

$$C_{1S}(\alpha_S) = \begin{bmatrix} -\text{sen}(\alpha_{S1}) & \cos(\alpha_{S1}) & d\cos(\alpha_{S1}) + b\text{sen}(\alpha_{S1}) \\ -\text{sen}(\alpha_{S2}) & \cos(\alpha_{S2}) & d\cos(\alpha_{S2}) + b\text{sen}(\alpha_{S2}) \\ \vdots & \ddots & \vdots \\ -\text{sen}(\alpha_{SNS}) & \cos(\alpha_{SNS}) & d\cos(\alpha_{SNS}) + b\text{sen}(\alpha_{SNS}) \end{bmatrix}$$

Las ecuaciones 2.55 y 2.56 representan las restricciones cinemáticas para el robot móvil en el plano de giro de la rueda y las ecuaciones 2.57 y 2.58, son las restricciones cinemáticas del robot móvil en el plano ortogonal de las ruedas. Estas ecuaciones pueden reagruparse en otras matrices, para las restricciones cinemáticas de las llantas en su plano de giro horizontal:

$$J_{1f}R(\theta)\dot{\epsilon}_I - J_{2f}\dot{\phi}_f(t) = 0 \dots \dots \dots \mathbf{2.55}$$

$$J_{1S}(\alpha_S)R(\theta)\dot{\epsilon}_I - J_{2NS}\dot{\phi}_{NS}(t) = 0 \dots \dots \dots \mathbf{2.56}$$

$$\begin{bmatrix} J_{1f} \\ J_{1S}(\alpha_S) \end{bmatrix} R(\theta)\dot{\epsilon}_I - \begin{bmatrix} J_{2f} & 0 \\ 0 & J_{2NS} \end{bmatrix} \begin{bmatrix} \dot{\phi}_f(t) \\ \dot{\phi}_{NS}(t) \end{bmatrix} = 0$$

$$J_1(\alpha_S)R(\theta)\dot{\epsilon}_I - J_2\dot{\phi}(t) = 0 \dots \dots \dots \mathbf{2.59}$$

Donde:

$$J_1(\alpha_s) = \begin{bmatrix} J_{1f} \\ J_{1s}(\alpha_s) \end{bmatrix} \quad J_2 = \begin{bmatrix} J_{2f} & 0 \\ 0 & J_{2Ns} \end{bmatrix} \quad \dot{\phi}(t) = \begin{bmatrix} \dot{\phi}_f(t) \\ \dot{\phi}_{Ns}(t) \end{bmatrix}$$

La matriz " $J_1(\alpha_s)$ " está en función de " α_s " porque " α_f " es fijo y es de dimensiones " $N \times 3$ ", donde $N = N_f + N_s$, la matriz " J_1 " es una matriz diagonal constante, los términos de su diagonal principal son los radios de cada rueda y es de dimensiones " $N \times N$ ", el término " $\dot{\phi}(t)$ " es un vector de " $N \times 1$ " que contiene las velocidades angulares de cada rueda.

Para las restricciones cinemáticas de las llantas en su plano ortogonal:

$$C_{1f}R(\theta)\dot{\epsilon}_I = 0 \dots \dots \dots 2.57$$

$$C_{1s}(\alpha_s)R(\theta)\dot{\epsilon}_I = 0 \dots \dots \dots 2.58$$

$$\begin{bmatrix} C_{1f} \\ C_{1s}(\alpha_s) \end{bmatrix} R(\theta)\dot{\epsilon}_I = 0$$

$$C_1(\alpha_s)R(\theta)\dot{\epsilon}_I = 0 \dots \dots \dots 2.60$$

Donde:

$$C_1(\alpha_s) = \begin{bmatrix} C_{1f} \\ C_{1s}(\alpha_s) \end{bmatrix}$$

La matriz " $C_1(\alpha_s)$ " está en función de " α_s " porque " α_f " es fijo, es de dimensiones " $N \times 3$ ", donde $N = N_f + N_s$.

En resumen, la ecuación 2.59 representa las restricciones que dicen que todas las ruedas fijas y, centradas orientables deben de girar en su plano horizontal y, la ecuación 2.60 representa todas las restricciones de las ruedas fijas y centradas orientables que dicen que el movimiento ortogonal a su plano horizontal de giro debe de ser cero, ya que no se permite deslizamiento.

2.2.2.3.1. Maniobrabilidad de un Robot Móvil

La movilidad cinemática de un robot móvil, es su capacidad para moverse directamente en el medio ambiente. Las restricciones que limitan la movilidad del robot son las reglas que cada una de sus ruedas tienen que satisfacer para sus restricciones de deslizamiento. Pensando de esta manera la movilidad del robot deriva directamente de la ecuación número 2.60.

2.2.2.3.1.1. Grado de Movilidad (r_m)

"Se refiere al número de grados de libertad de un robot móvil que pueden ser manipulados inmediatamente a través de cambios de velocidad en sus ruedas, sin orientar ninguna de sus llantas".

La ecuación 2.60 impone las restricciones de movimiento de cada rueda y dicta que el movimiento lateral es cero, es decir, no es posible el deslizamiento de ninguna llanta fija ni centrada orientable, sin embargo dos o más llantas pueden imponer la misma restricción en la ecuación 2.60, por ejemplo, en un robot diferencial, se tienen dos llantas fijas y una llanta descentrada orientable libre (figura 2.26), se sabe que sólo las llantas fijas producen restricciones, la ecuación 2.60, entonces puede quedar definida como se puede observar en la ecuación 2.61, el renglón número 1 de la matriz " $C_1(\alpha_s)$ " es exactamente igual que el renglón 2, ya que $\alpha_{f1} = \alpha_{f2}$.

$$C_1(\alpha_s)R(\theta)\dot{\epsilon}_I = 0 \dots \dots 2.61$$

$$\begin{bmatrix} -\sin(\alpha_{f1}) & \cos(\alpha_{f1}) & d\cos(\alpha_{f1}) - b\sin(\alpha_{f1}) \\ -\sin(\alpha_{f2}) & \cos(\alpha_{f2}) & d\cos(\alpha_{f2}) - b\sin(\alpha_{f2}) \end{bmatrix} R(\theta)\dot{\epsilon}_I = 0$$

Donde:

$$C_1(\alpha_s) = \begin{bmatrix} -\sin(\alpha_{f1}) & \cos(\alpha_{f1}) & d\cos(\alpha_{f1}) - b\sin(\alpha_{f1}) \\ -\sin(\alpha_{f2}) & \cos(\alpha_{f2}) & d\cos(\alpha_{f2}) - b\sin(\alpha_{f2}) \end{bmatrix}$$

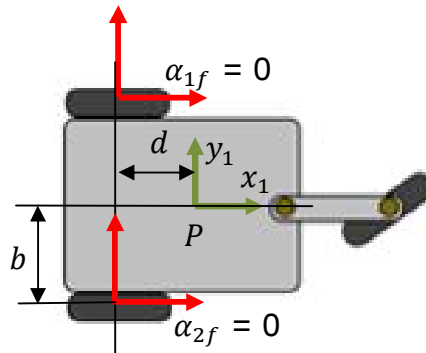


Figura 2.26. Robot tipo diferencial.

El hecho de que los renglones de la matriz " $C_1(\alpha_s)$ " de la ecuación 2.61 sean iguales, quiere decir que en realidad basta con una restricción de una llanta para entender que el robot no puede moverse hacia una dirección de forma instantánea. El número de renglones independientes de la matriz " $C_1(\alpha_s)$ " de la ecuación 2.60, es igual al rango de dicha matriz.

En general, un robot que se mueve en el plano tendrá tres grados de libertad; movimiento en "x", horizontal, movimiento en "y", vertical, y movimiento angular. Pensando de esta manera, si el rango de la matriz " $C_1(\alpha_s)$ " es cero, quiere decir que no hay restricciones, es decir, sus llantas son omnidireccionales, por otro lado, si el rango de la matriz " $C_1(\alpha_s)$ " es igual a tres, quiere decir que el robot está completamente restringido y no se puede mover en ninguna dirección. Este es el rango máximo posible, ya que las restricciones cinemáticas se especifican a lo largo de tres grados de libertad (es decir, la matriz de restricción es de tres columnas de ancho).

Con lo anteriormente mencionado, es posible definir de manera formal el grado de movilidad del robot " r_m ":

$$r_m = 3 - \text{rang}[C_1(\alpha_s)] \dots \dots \dots 2.62$$

2.2.2.3.1.2. Grado de Manejabilidad (r_s)

"Se encuentra definido por el número de llantas centradas orientables, que son libres para ser controladas de forma independiente".

La direccionalidad puede tener un impacto en la postura de un robot móvil, aunque este cambio sólo es perceptible hasta que, después de cambiar el ángulo de una rueda centrada orientable, el robot se empieza a mover.

Como en la movilidad, cuando se define el grado de manejabilidad nos interesa el número de parámetros o ángulos controlados de forma independiente. La matriz " $C_1(\alpha_s)$ " de la ecuación 2.60 está compuesta por las matrices " C_{1f} " y " $C_{1s}(\alpha_s)$ ", en este caso sólo nos interesa la matriz " $C_{1s}(\alpha_s)$ ".

Entonces, el grado de manejabilidad queda definido como el número de ángulos controlables independientemente, en otras palabras el rango de la matriz " $C_{1s}(\alpha_s)$ ".

$$r_s = \text{rang}[C_{1s}(\alpha_s)] \dots \dots \dots 2.63$$

2.2.2.3.1.3. Grado de Maniobrabilidad (r_M)

"Define el total de grados de libertad que el robot puede manipular"

Está definida en términos de la movilidad y la manejabilidad (ecuación 2.64), por lo tanto incluye los grados de libertad que el robot puede manipular directamente a través de cambios de velocidad en sus ruedas y los grados de libertad que son indirectamente manipulados al moverse el robot, por cambios en las direcciones de sus ruedas centradas orientables independientes unas de otras.

$$r_M = r_m + r_s \dots \dots \dots 2.64$$

Con lo dicho anteriormente, es posible definir las configuraciones básicas que un robot móvil puede tener, en la tabla 2.6, [16] es posible notar que el grado de maniobrabilidad, por sí mismo, no define al tipo de robot móvil, porque dos robots con el mismo " r_M ", pero diferente " r_m " no son equivalentes. Por esta razón, para definir el tipo de robot móvil se utilizan los términos " r_m " y el " r_s ", generando una nomenclatura como la siguiente: "*robot móvil del tipo (r_m, r_s)*".

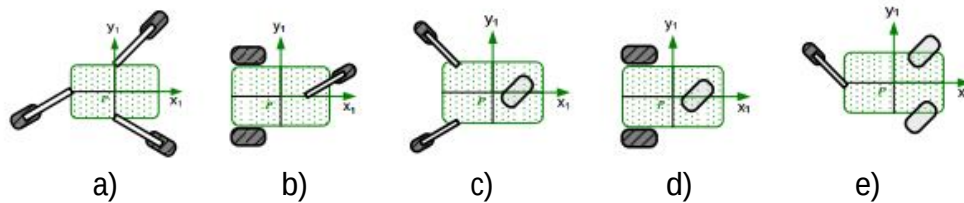


Tabla 2.6. Configuraciones básicas de los robots móviles, [16].

- a) Robot omnidireccional, tipo (3, 0).
- b) Robot diferencial, tipo (2, 0).
- c) Robot omnidireccional dirigido, tipo (2, 1).
- d) Robot triciclo, tipo (1, 1).
- e) Robot bi-dirigido, tipo (1, 2).

2.2.3. Postura de un Robot Móvil con Configuración Ackerman

En el robot tipo Ackerman se tienen cuatro ruedas, dos de ellas se ubican en la parte trasera y son fijas, las otras dos delanteras son centradas orientables.

Como se mencionó en el apartado "2.2.2.1", para poder modelar el robot se considera que es un cuerpo rígido y que su movimiento se realiza sobre un plano horizontal sin deslizamiento. Con el fin de especificar la posición del robot se definieron dos planos de referencia, uno global y uno local sobre el mismo robot, como se muestra en la figura 2.27.

El marco de referencia global está definido por los ejes " I_x " y " I_y ", mientras que el local se encuentra sobre el punto "P" definiendo los ejes locales " x_1 " y " y_1 ". La posición de P en el marco de referencia global está definido por el vector 2.6 y sus velocidades por el vector 2.7, (para más detalles dirigirse al apartado "2.1.2.1").

$$\varepsilon_I = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \dots \dots \dots 2.6$$

$$\dot{\varepsilon}_I = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \dots \dots \dots 2.7$$

Donde " $\dot{x}$ " y " $\dot{y}$ " son las componentes de la velocidad lineal y " $\dot{\theta}$ " es la velocidad angular del robot.

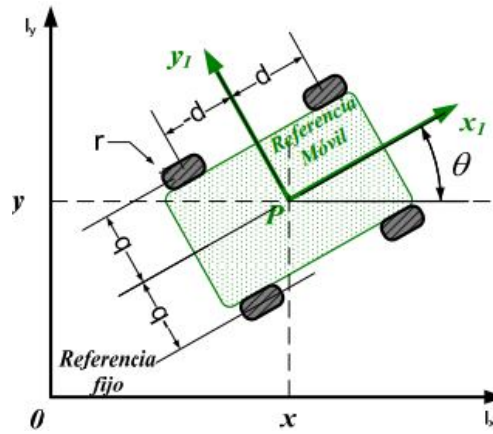


Figura 2.27. Posición del robot sobre un marco de referencia global, [5].

El vector ε_I contiene la posición y orientación del punto "P", es decir, contiene la posición y la orientación que se desea que el robot tenga, midiendo, dichos componentes, desde el marco global o fijo de referencia. Para describir el movimiento del robot en términos del marco de referencia local, o viceversa, es necesario realizar una transformación de una base a otra, esto se logra empleando la matriz de rotación, definida en el apartado 2.1.2.1.1, para el caso particular en el que el movimiento sea en el plano, se emplea la matriz 2.17, ésta se utiliza para mapear los movimientos del robot del marco de referencia global al marco de referencia local, esta operación es denotada por la ecuación 2.22.

$$\dot{\varepsilon}_R = R(\theta)\dot{\varepsilon}_I \dots \dots \dots 2.22$$

Donde:

$$R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \dots \dots \dots 2.17$$

Para el proceso contrario, ubicar al robot del marco de referencia local al marco de referencia global, se utiliza la operación denotada por la ecuación 2.23.

$$R(\theta)^{-1}\dot{\varepsilon}_R = R(\theta)^{-1}R(\theta)\dot{\varepsilon}_I$$

$$\dot{\varepsilon}_I = R(\theta)^{-1}\dot{\varepsilon}_R \dots \dots \dots 2.23$$

Considerando que las matrices de rotación son matrices ortogonales, la matriz inversa $R(\theta)^{-1}$ es igual a la matriz transpuesta de $R(\theta)$, es decir $R(\theta)^{-1} = R(\theta)^T$, entonces la ecuación 2.23 la podemos escribir como la ecuación 2.24.

$$\dot{\varepsilon}_I = R(\theta)^T \dot{\varepsilon}_R \dots \dots \dots 2.24$$

Donde:

$$R(\theta)^T = R(\theta)^{-1} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

La ecuación 2.24 ubica al robot en todo momento en el sistema de referencia global o fijo, sin embargo, aún no es conocido como se calcula el vector 2.7, para ello tomamos como referencia la figura 2.28, en ella se puede observar la velocidad lineal " V_R " y la velocidad angular " W_R ", la intención es: teniendo estas dos velocidades " V_R " y " W_R ", averiguar qué valor toma cada componente del vector 2.7.

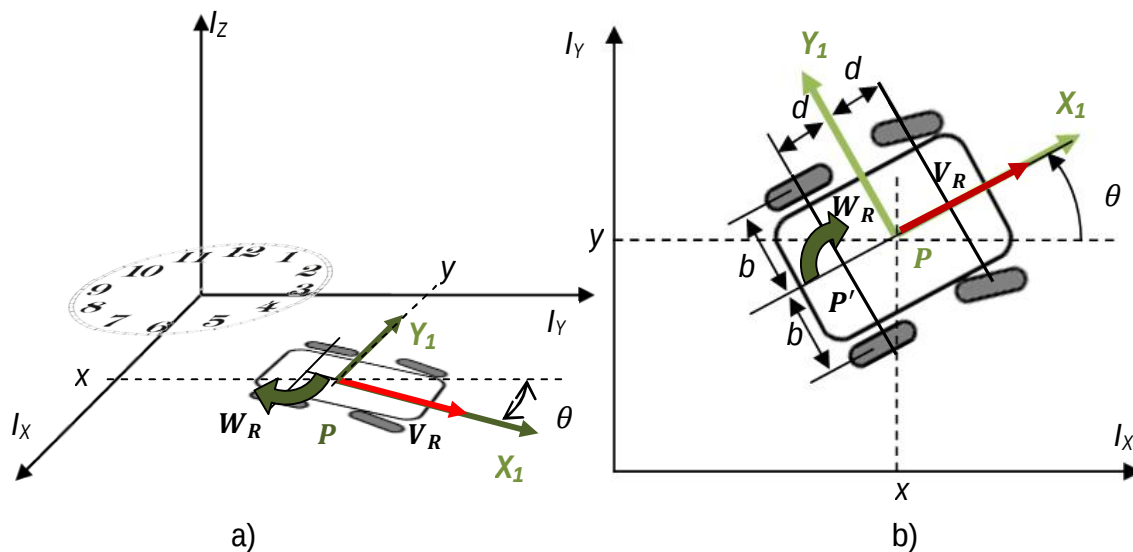


Figura 2.28. Movimiento del robot móvil sobre el plano

Para facilitar su análisis, se toma un nuevo punto " P' " que se encuentra centrado entre las llantas traseras del robot, en este punto se aplica la velocidad angular, " W_R ", resultante de la rotación del robot móvil, como se puede ver en la figura 2.28.a, la rotación del robot móvil provoca una velocidad angular negativa, considerando que es aplicada en sentido de las manecillas del reloj, la figura 2.28.b, muestra el movimiento del robot en el plano " I_X " y " I_Y ", la velocidad del punto " P " se muestra en las ecuaciones 2.65 y 2.66.

$$\dot{x} = V_{RX} + V_{WX} \dots \dots \dots 2.65$$

$$\dot{y} = V_{RY} + V_{WY} \dots \dots \dots 2.66$$

Donde:

V_{RX} = Proyección de la velocidad lineal " V_R " sobre el eje paralelo a I_X .

V_{RY} = Proyección de la velocidad lineal " V_R " sobre el eje paralelo a I_Y .

V_{WX} = Proyección de la velocidad resultante de " $W_R * d$ " sobre el eje paralelo a l_X .

V_{WY} = Proyección de la velocidad resultante de " $W_R * d$ " sobre el eje paralelo a l_Y .

En la figura 2.29, se observan las velocidades " V_{RX} ", " V_{RY} ", " V_{WX} " y " V_{WY} " y se pueden obtener las siguientes relaciones:

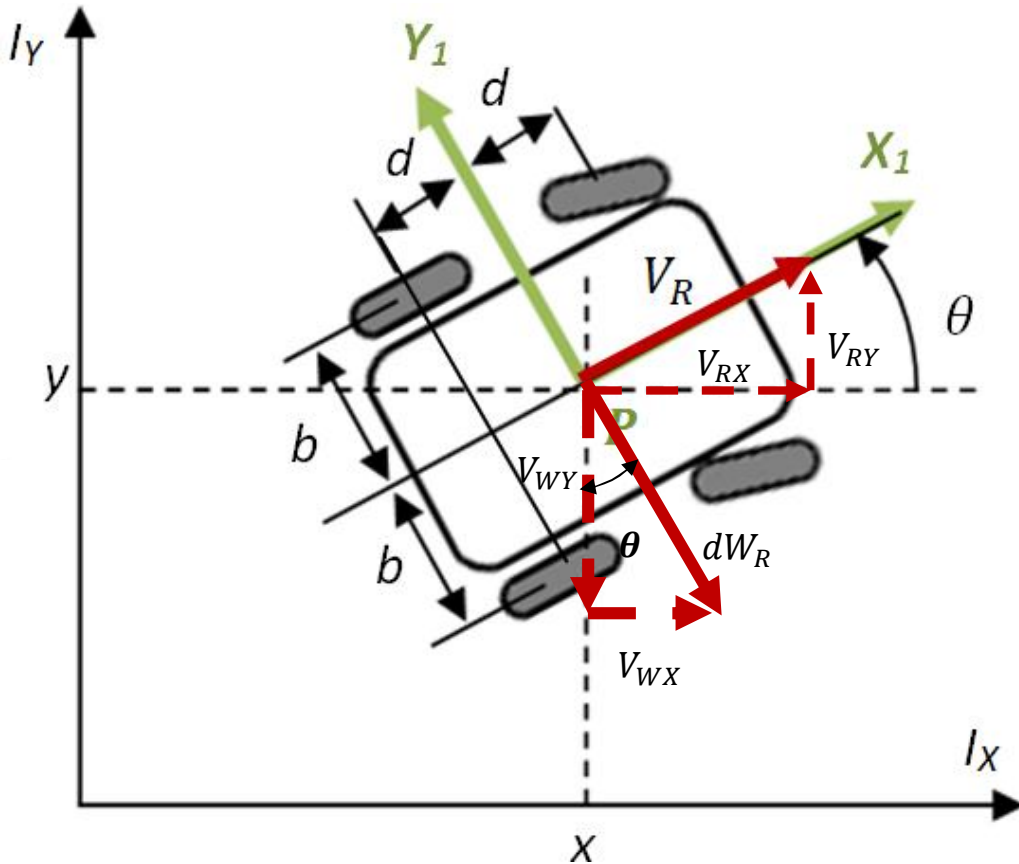


Figura 2.29. Proyecciones de la velocidad lineal y angular del robot.

$$\cos(\theta) = \frac{V_{RX}}{V_R}$$

$$V_{RX} = V_R \cos(\theta) \dots \dots \dots 2.67$$

$$\sin(\theta) = \frac{V_{RY}}{V_R}$$

$$V_{RY} = V_R \sin(\theta) \dots \dots \dots 2.68$$

$$\cos(\theta) = \frac{-V_{WY}}{dW_R}$$

$$V_{WY} = -dW_R \cos(\theta) \dots \dots \dots 2.69$$

$$\text{sen}(\theta) = \frac{V_{WX}}{dW_R}$$

$$V_{WX} = dW_R \text{sen}(\theta) \dots \dots \dots 2.70$$

Las velocidades del punto "P" se obtienen sustituyendo las ecuaciones 2.67 y 2.70 en 2.65, de igual manera las ecuaciones 2.68 y 2.69 en 2.66.

$$\dot{x} = V_R \cos(\theta) + dW_R \text{sen}(\theta) \dots \dots \dots 2.71$$

$$\dot{y} = V_R \text{sen}(\theta) + -dW_R \cos(\theta) \dots \dots \dots 2.72$$

En este momento se tienen los primeros dos componentes del vector 2.7, el tercer elemento es " $\dot{\theta}$ " que resulta ser igual a " W_R ".

$$\dot{\theta} = W_R \dots \dots \dots 2.73$$

Las ecuaciones 2.71, 2.72 y 2.73 se pueden acomodar de la siguiente manera:

$$\dot{\epsilon}_I = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & d\text{sen}(\theta) \\ \text{sen}(\theta) & -d\cos(\theta) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_R \\ W_R \end{bmatrix} \dots \dots \dots 2.74$$

La ecuación 2.74 representa la cinemática de la postura del robot móvil.

2.2.4. Modelo Cinemático de un Robot Móvil con Configuración Ackerman

La velocidad angular de las llantas traseras, así como el ángulo de orientación de las llantas delanteras, siguen una ley cinemática que le permite al robot móvil desplazarse, siguiendo una trayectoria preestablecida o dirigirse hacia un punto de destino de una manera armónica y sin deformar la base del robot.

Como se puede apreciar en la tabla 2.6, la configuración Ackerman no forma parte de las configuraciones básicas de los robots móviles, esto es porque las llantas centradas orientables delanteras son dependientes una de otra, para entender su funcionamiento se toma una llanta centrada entre estas dos como referencia, aunque en algunos trabajos esta puede estar ubicada en algún otro lugar [16], de esta manera es posible ver que se trata de un robot móvil tipo triciclo, sin embargo, un automóvil convencional, normalmente no tiene una configuración tipo triciclo debido que, la plataforma o chasis, no contaría con la suficiente estabilidad en curvas con alta velocidad, este problema se solucionó colocando dos ruedas delanteras en lugar de una.

Para lograr que el robot móvil se desplace es necesario conocer el comportamiento de sus cuatro ruedas para que sea posible controlar su movimiento, en la figura 2.30, se muestra

un esquema para su análisis y en la tabla 2.7, se muestran los parámetros de su cinemática.

Considerando que el robot móvil es rígido, no deformable, se mueve sobre un plano horizontal y sus llantas no se deslizan, se obtiene las siguientes representaciones matemáticas del robot móvil, que indican la velocidad angular de las llantas traseras y la orientación de las llantas delanteras, tomando como variables la velocidad lineal, " V_R ", y la velocidad angular, " W_R ", del prototipo.

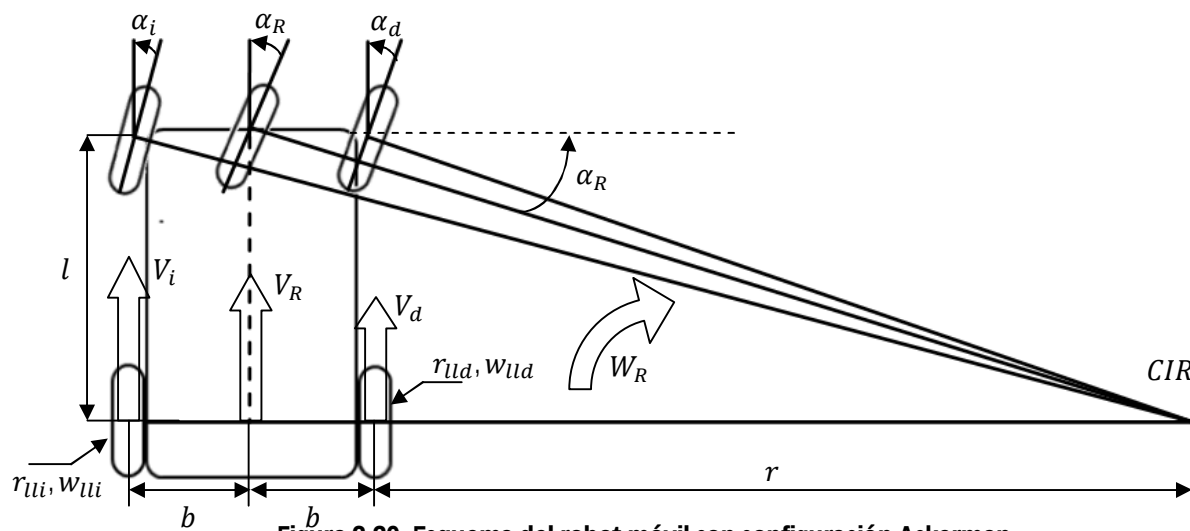


Figura 2.30. Esquema del robot móvil con configuración Ackerman.

De la figura 2.30, es posible obtener las siguientes relaciones:

$$RD = r + b \dots \dots \dots 2.75$$

$$\tan(\alpha_R) = \frac{l}{RD} \dots \dots \dots 2.76$$

La velocidad lineal del robot se obtiene multiplicando el brazo de palanca por su velocidad angular " W_R ":

$$V_R = W_R (RD) \dots \dots \dots 2.77$$

El termino " RD " queda definido por la siguiente ecuación:

$$RD = \frac{V_R}{W_R} \dots \dots \dots 2.78$$

Parámetros de la cinemática del robot móvil	
Parámetro	Descripción
V_R	Velocidad lineal del robot
V_d	Velocidad lineal de la llanta trasera derecha
V_i	Velocidad lineal de la llanta trasera izquierda

W_R	Velocidad angular del robot
w_{lld}	Velocidad angular de la llanta trasera derecha
w_{lli}	Velocidad angular de la llanta trasera izquierda
α_R	Ángulo de dirección del robot
α_d	Ángulo de dirección de la llanta delantera derecha
α_i	Ángulo de dirección de la llanta delantera izquierda
b	Mitad del ancho del robot móvil
$l = 2d$	Largo del robot móvil
r	Radio de giro al centro instantáneo de rotación
r_{lld}	Radio de la llanta trasera derecha
r_{lli}	Radio de la llanta trasera izquierda
CIR	Centro instantáneo de rotación

Tabla 2.7. Parámetros de la cinemática del robot móvil.

De igual manera, las velocidades de sus llantas traseras son:

$$V_d = W_R (RD - b) \dots \dots \dots 2.79$$

$$V_i = W_R (RD + b) \dots \dots \dots 2.80$$

En la figura 2.31, podemos ver un esquema de la rueda trasera derecha y, es posible observar la relación del radio de la llanta con su velocidad lineal y su velocidad angular.

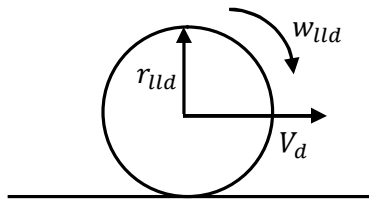


Figura 2.31. Variables en la rueda derecha trasera del robot.

Las velocidades angulares de la llanta derecha e izquierda están dadas por las ecuaciones 2.81 y 2.82 respectivamente.

$$V_d = w_{lld} (r_{lld})$$

$$w_{lld} = \frac{V_d}{r_{lld}} \dots \dots \dots 2.81$$

$$V_i = w_{lli} (r_{lli})$$

$$w_{lli} = \frac{V_i}{r_{lli}} \dots \dots \dots 2.82$$

Sustituyendo la ecuación 2.79 y 2.80 en las ecuaciones 2.81 y 2.82 respectivamente:

$$w_{lld} = \frac{W_R (RD - b)}{r_{lld}} \dots \dots \dots 2.83$$

$$w_{lli} = \frac{W_R (RD + b)}{r_{lli}} \dots \dots \dots 2.84$$

Sustituyendo 2.78 en 2.83 y 2.84 y simplificando:

$$w_{lld} = \frac{W_R \left(\frac{V_R}{W_R} - b \right)}{r_{lld}}$$

$$w_{lld} = \frac{1}{r_{lld}} (V_R - W_R b) \dots \dots \dots 2.85$$

$$w_{lli} = \frac{W_R \left(\frac{V_R}{W_R} + b \right)}{r_{lli}}$$

$$w_{lli} = \frac{1}{r_{lli}} (V_R + W_R b) \dots \dots \dots 2.86$$

La dirección del robot queda determinada con los ángulos " α_d " y " α_i " de las llantas delanteras respectivamente, para su cálculo es necesario tomar como referencia el ángulo " α_R " de la llanta ficticia, de la figura 2.30, podemos hallar las siguientes relaciones para las llantas delanteras, incluyendo la ficticia:

$$\cot(\alpha_R) = \frac{RD}{l} \dots \dots \dots 2.87$$

$$\cot(\alpha_d) = \frac{RD - b}{l} \dots \dots \dots 2.88$$

$$\cot(\alpha_i) = \frac{RD + b}{l} \dots \dots \dots 2.89$$

Relacionando 2.88 y 2.89 de la siguiente manera y simplificando:

$$\cot(\alpha_i) - \cot(\alpha_d) = \frac{RD + b}{l} - \frac{RD - b}{l}$$

$$\cot(\alpha_i) - \cot(\alpha_d) = \frac{2b}{l} \dots \dots \dots 2.90$$

De igual manera, relacionando 2.87 y 2.88 de la misma manera y simplificando:

$$\begin{aligned} \cot(\alpha_d) - \cot(\alpha_R) &= \frac{RD - b}{l} - \frac{RD}{l} \\ \cot(\alpha_d) - \cot(\alpha_R) &= -\frac{b}{l} \dots \dots \dots 2.91 \end{aligned}$$

Despejando de 2.91 " $\cot(\alpha_d)$ " se obtiene:

$$\cot(\alpha_d) = \cot(\alpha_R) - \frac{b}{l} \dots \dots \dots 2.92$$

Para obtener el ángulo de la llanta izquierda despejamos " $\cot(\alpha_i)$ " de 2.90, sustituimos 2.92 y simplificamos:

$$\begin{aligned} \cot(\alpha_i) &= \cot(\alpha_d) + \frac{2b}{l} \\ \cot(\alpha_i) &= \cot(\alpha_R) - \frac{b}{l} + \frac{2b}{l} \\ \cot(\alpha_i) &= \cot(\alpha_R) + \frac{b}{l} \dots \dots \dots 2.93 \end{aligned}$$

Sustituyendo 2.87 en 2.92 y en 2.93 respectivamente:

$$\begin{aligned} \cot(\alpha_d) &= \frac{RD}{l} - \frac{b}{l} \dots \dots \dots 2.94 \\ \cot(\alpha_i) &= \frac{RD}{l} + \frac{b}{l} \dots \dots \dots 2.95 \end{aligned}$$

Se sustituye 2.78 en 2.94 y se simplifica.

$$\begin{aligned} \cot(\alpha_d) &= \frac{1}{l} \left(\frac{V_R}{W_R} - b \right) \\ \frac{1}{\tan(\alpha_d)} &= \frac{1}{l} \left(\frac{V_R - W_R b}{W_R} \right) \\ \tan(\alpha_d) &= \frac{l W_R}{V_R - W_R b} \\ \alpha_d &= \text{atan} \left(\frac{l W_R}{V_R - W_R b} \right) \dots \dots \dots 2.96 \end{aligned}$$

De igual manera, se sustituye 2.78 en 2.95 y se simplifica.

$$\begin{aligned}
\cot(\alpha_i) &= \frac{1}{l} \left(\frac{V_R}{W_R} + b \right) \\
\frac{1}{\tan(\alpha_i)} &= \frac{1}{l} \left(\frac{V_R + W_R b}{W_R} \right) \\
\tan(\alpha_i) &= \frac{l W_R}{V_R + W_R b} \\
\alpha_i &= \text{atan} \left(\frac{l W_R}{V_R + W_R b} \right) \dots \dots \dots 2.97
\end{aligned}$$

Las ecuaciones 2.85 y 2.86 representan las velocidades angulares de la llanta trasera derecha e izquierda del robot, respectivamente, y las ecuaciones 2.96 y 2.97 representan la posición angular de la llanta delantera derecha e izquierda, correspondientemente.

2.3. Plataforma Móvil

El teléfono móvil es el nuevo ordenador personal. La computadora de escritorio no va a desaparecer, pero el mercado de teléfonos inteligentes está creciendo rápidamente. Los teléfonos están siendo utilizados como ordenadores por más personas y para más propósitos. Los teléfonos inteligentes son generalmente más baratos que los ordenadores, más convenientes debido a su portabilidad y, a menudo más útil para ciertos propósitos debido a los sensores que incorpora como por ejemplo el GPS con el cual es posible proporcionar un servicio de geolocalización, por ejemplo [20].

Como los teléfonos inteligentes ganan popularidad entre la población de manera general, la computadora de escritorio, posiblemente, será relegada a los profesionistas relacionados al desarrollo de software o especialistas. Muchas de las tareas rutinarias que realizamos actualmente en una computadora de escritorio o portátil, las realizará un teléfono inteligente. El desarrollo de software se desplazará hacia el desarrollo móvil, puesto que es muy probable que se tengan más teléfonos inteligentes que computadoras, actualmente existen más dispositivos móviles conectados a internet que computadoras personales [21]. La industria del software se movilizará y lo está haciendo en estos días, [20], un ejemplo claro de la potencia de los "smartphone" puede ser apreciado en el programa de la "NASA" llamado "PhoneSat" en el cual se emplean teléfonos con sistema operativo Android como corazón de pequeños satélites, en el momento de la escritura de este trabajo, la NASA ha mandado tres de estos satélites [19].

2.3.1. ¿Qué es un Smartphone?

La línea entre lo que es un dispositivo móvil y lo que no lo es puede ser un poco difusa, pero en general, se pueden definir como aquellos microordenadores que son lo suficientemente ligeros como para ser transportados por una persona, y que disponen de la capacidad de batería suficiente como para poder funcionar de forma independiente. Los ordenadores portátiles no se consideran como dispositivos móviles, ya que consumen más batería y suelen ser un poco más pesados de lo que se espera de algo pensado para llevar siempre con sigo.

Un **teléfono inteligente**, "*smartphone*" en inglés, es un teléfono móvil construido sobre un sistema operativo, con una mayor capacidad de almacenamiento de datos y conectividad que un teléfono móvil convencional, además son capaces de realizar tareas semejantes a un miniordenador. El término «inteligente» hace referencia a la capacidad de usarse como un ordenador de bolsillo, llegando incluso a remplazar a un ordenador personal en algunos casos.

La mayoría de los teléfonos inteligentes permiten al usuario instalar programas adicionales, normalmente inclusive desde terceros. Otras características comunes de los "smartphone" son:

- ❖ Acceso a Internet vía WiFi o red 3G.
- ❖ Función multimedia (cámara y reproductor de videos).
- ❖ Habilidad para abrir documentos de diversos formatos, como PDF y Microsoft Office.
- ❖ Incorporación de diversos sensores como acelerómetro y GPS.

De forma parecida a el caso de las computadoras de escritorio, los teléfonos inteligentes tienen un conjunto de diversos sistemas operativos (véase la tabla 2.8). Por otra parte, a diferencia de los sistemas operativos de escritorio, el sistema operativo en la informática móvil normalmente determina el lenguaje de programación que los desarrolladores deben utilizar para poder programar de forma nativa.

OS	Symbian	RIM BlackBerry	Apple Iphone	Windows Mobile	Google Android
Lenguaje	C++	Java	Objective-C	C #	Java

Tabla 2.8 Sistemas operativos y lenguajes de programación en los smartphone.

Como se puede ver, existen una gran variedad de plataformas móviles las cuales pueden ser utilizadas para el desarrollo del presente trabajo; la tabla 2.9 muestra el porcentaje y la cantidad de unidades vendidas a nivel mundial en el año 2011 y 2012 de los diversos

sistemas operativos que los "smartphone" tienen, en este se puede apreciarse que el número de terminales con Android es el más consumido, además de que existe un incremento de este tipo de terminales [21].

Sistema Operativo	Unidades Vendidas en el 2012	Porcentaje de unidades vendidas en 2012	Unidades Vendidas en el 2011	Porcentaje de unidades vendidas en 2011
Android	497.1	68.8 %	243.5	49.2 %
iOS	135.9	18.8 %	93.1	18.8 %
BlackBerry	32.5	4.5 %	51.1	10.3 %
Symbian	23.9	3.3 %	81.5	16.5 %
Windows Phone	17.9	2.5 %	9.0	1.8 %
Otros	15.1	2.1 %	16.3	3.3 %
TOTAL	722.4	100 %	494.5	100 %

Tabla 2.9 Sistemas operativos más vendidos en el 2011 y 2012.

El sistema operativo seleccionado para el desarrollo de este trabajo es Android, para aprovechar los recursos que este sistema ofrece, se trabajará particularmente con el lenguaje nativo el cual es java.

2.3.2. Plataforma Android

El sistema operativo Android es la primera plataforma móvil de código abierto, es un proyecto de Google, en colaboración con la "Open Handset Alliance", la cual es una agrupación de casi 50 corporaciones comprometidas con el objetivo de comercializar un mejor teléfono móvil y abierto.

Android es un entorno de software creado para dispositivos móviles. Incluye un sistema operativo basado en Linux, una completa interfaz de usuario, aplicaciones, bibliotecas de código, estructuras para aplicaciones, compatibilidad multimedia, etc. Como la plataforma es abierta, es posible modificarla. Google pone a la disposición de todos, el código completo de las versiones que van quedándose atrás, para mayor información visitar la página web <https://developers.google.com/open-source/?hl=es-419>. En las siguientes secciones se presenta al sistema operativo Android de manera general, si se desea conocer el desarrollo de estas aplicaciones con más profundidad es necesario referirse a una bibliografía más especializada en el tema [20], o también consultar la página de desarrollo de Google; <http://developer.android.com/index.html>.

2.3.3. Arquitectura de Android

La figura 2.32 muestra los componentes principales del sistema operativo Android, este está formado por cuatro capas. Una de las características más importantes es que todas las capas están basadas en *software* libre.

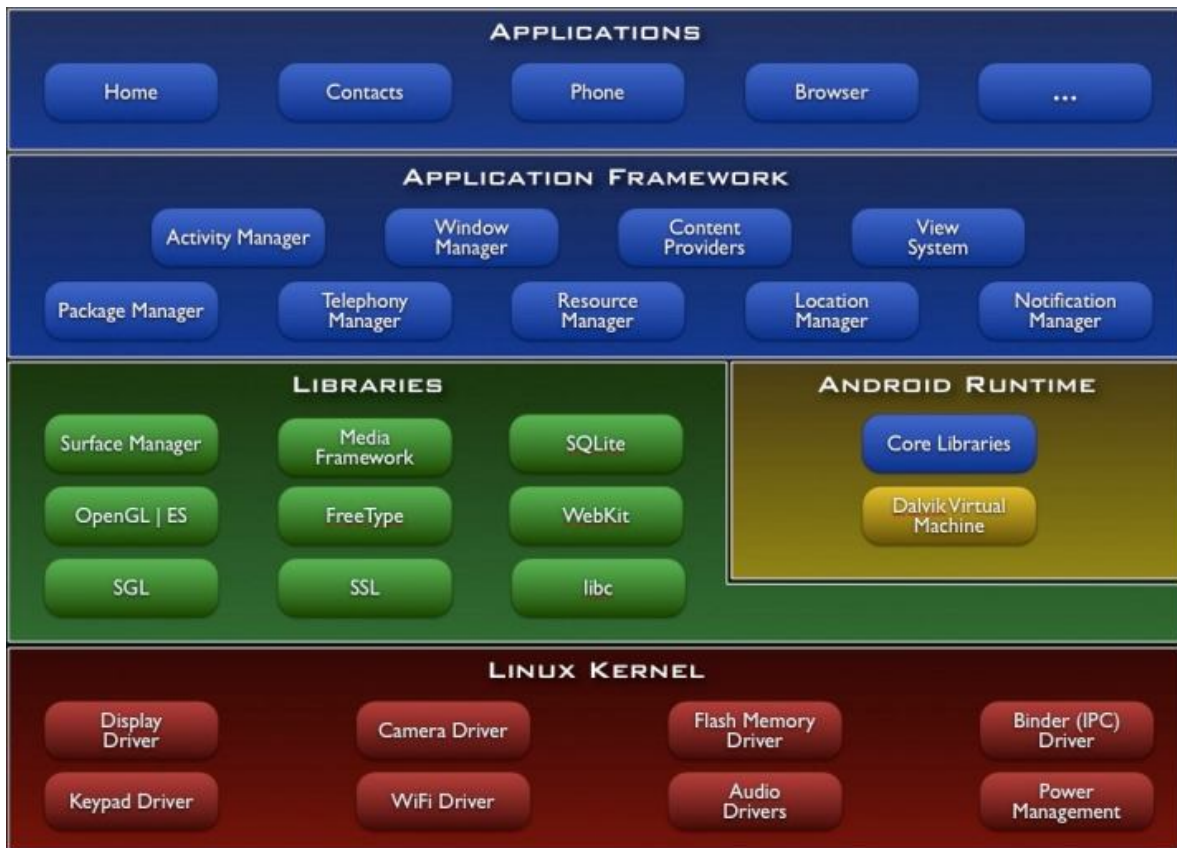


Figura 2.32 Arquitectura del sistema operativo Android.

2.3.3.1. El núcleo Linux

El núcleo de Android está formado por el sistema operativo Linux versión 2.6. Esta capa proporciona servicios como la seguridad, el manejo de la memoria, el multiproceso, la pila de protocolos y el soporte de *drivers* para dispositivos. Este núcleo también actúa como capa de abstracción entre el hardware y el resto de la pila. Por lo tanto, es la única que es dependiente del *hardware*.

2.3.3.2. Runtime de Android

Está basado en el concepto de máquina virtual utilizado en Java. Dado las limitaciones de los dispositivos donde ha de correr el sistema Android, poca memoria y procesador limitado, no es posible utilizar una máquina virtual Java estándar. Google tomó la decisión de crear una nueva, la máquina virtual Dalvik, que respondiera mejor a estas limitaciones.

Una máquina virtual puede considerarse como un intérprete entre un lenguaje de alto nivel como Java y un lenguaje de bajo nivel, listo para ser entendido por el hardware de la plataforma.

Cada aplicación corre en su propio proceso Linux con su propia instancia de la máquina virtual Dalvik.

2.3.3.3. Librerías Nativas

Incluye un conjunto de librerías en C/C++ usadas en varios componentes de Android. Muchas de las librerías utilizan proyectos de código abierto. Algunas de estas librerías son:

- ❖ **System C library:** Una derivación de la librería BSD de C estándar (libc), adaptada para dispositivos embebidos basados en Linux.
- ❖ **Media Framework:** Librería basada en *PacketVideo's OpenCORE*; soporta códigos de reproducción y grabación de varios formatos de audio, vídeo e imágenes MPEG4, H.264, MP3, AAC, AMR, JPG y PNG.
- ❖ **Surface Manager:** Maneja el acceso al subsistema de representación gráfica en 2D y 3D.
- ❖ **WebKit:** Soporta un moderno navegador web y es utilizado en el navegador Android y en la vista *webview*. Se trata de la misma librería que utiliza *Google Chrome* y *Safari* de *Apple*.
- ❖ **SGL:** Motor de gráficos 2D.
- ❖ **Librerías 3D:** Es una aplicación basada en el API de *OpenGL ES 1.0*. Las librerías utilizan un hardware de aceleración 3D, si está disponible, o el software altamente optimizado de proyección 3D.
- ❖ **FreeType:** Utiliza bitmap y renderizado vectorial.
- ❖ **SQLite:** Es un potente y ligero motor de bases de datos relacionales disponible para todas las aplicaciones.

2.3.3.4. Entorno de Aplicación

Proporciona una plataforma de desarrollo libre para aplicaciones con gran riqueza e innovaciones (sensores, localización, servicios, barra de notificaciones, etc.).

Esta capa ha sido diseñada para simplificar la reutilización de componentes. Las aplicaciones pueden publicar sus capacidades y otras pueden hacer uso de ellas (sujetas a las restricciones de seguridad). Este mismo mecanismo permite a los usuarios reemplazar componentes.

Una de las mayores fortalezas del entorno de aplicación de Android es que se aprovecha el lenguaje de programación Java.

Los servicios más importantes que incluye son:

- ❖ **Views:** Extenso conjunto de vistas, (parte visual de los componentes).
- ❖ **Resource Manager:** Proporciona acceso a recursos que no son en código.
- ❖ **Activity Manager:** Maneja el ciclo de vida de las aplicaciones y proporciona un sistema de navegación entre ellas.
- ❖ **Notification Manager:** Permite a las aplicaciones mostrar alertas personalizadas en la barra de estado.
- ❖ **Content Providers:** Mecanismo sencillo para acceder a datos de otras aplicaciones (como los contactos).

2.3.3.5. Aplicaciones

Este nivel está formado por el conjunto de aplicaciones instaladas en el dispositivo Android, ejemplos de estas son la aplicación para los contactos, navegador web, calendario, mapas, etc. Todas las aplicaciones han de correr en la máquina virtual Dalvik para garantizar la seguridad del sistema.

Normalmente las aplicaciones Android están escritas en Java. Para desarrollar aplicaciones en Java podemos utilizar el Android SDK. Existe otra opción, consistente en desarrollar las aplicaciones utilizando C/C++, para esta opción podemos utilizar el Android NDK (*Native Development Kit*).

2.3.4. Componentes de una aplicación Android

El término "*app*" es una abreviatura de la palabra en inglés "*application*", es decir, una "*app*" es un programa, pero con unas características especiales. Se refieren sobre todo a aplicaciones destinadas a "*tablets*" o a teléfonos del tipo "*smartphone*" y suelen ser más dinámicas que los programas tradicionales. Hasta cierto punto son una buena combinación entre un programa normal y un "*gadget*" de escritorio.

Hay cuatro tipos diferentes de componentes de aplicación. Cada tipo tiene un propósito distinto y tiene un ciclo de vida diferente que define cómo el componente se crea y se destruye. Los cuatro tipos de componentes de una aplicación Android son:

- ❖ *Activity*.

- ❖ *Service.*
- ❖ *Content Provider.*
- ❖ *Broadcast Receiver.*

Cabe mencionar que una aplicación, no necesariamente debe de contener todos estos cuatro elementos, todo depende de la finalidad o tarea que la aplicación deba de realizar, por ejemplo, si se desea tener una aplicación que muestre un formulario, no será necesario más que implementar una pantalla que muestre la información necesaria para el usuario, si además esta aplicación puede recibir información de forma automática al recibir mensajes de texto, por ejemplo, además de una pantalla, deberá implementar un mecanismo que ayude a la aplicación a leer automáticamente el mensaje SMS y acceder a la información contenida en él para utilizarla. A continuación se muestra una descripción de los cuatro componentes principales que puede contener una aplicación.

2.3.4.1. Activity

Las actividades (*activities*) representan el componente principal de la interfaz gráfica de una aplicación Android. Se puede pensar en una actividad como el elemento análogo a una ventana o pantalla [20].

2.3.4.2. Service

Los servicios (*service*) son componentes sin interfaz gráfica que se ejecutan en segundo plano. Los servicios pueden realizar cualquier tipo de acción, por ejemplo actualizar datos, lanzar notificaciones, o incluso mostrar elementos visuales (por ejemplo actividades) si se necesita en algún momento la interacción con del usuario [20].

2.3.4.3. Content Provider

Un proveedor de contenidos (*content provider*) es el mecanismo que implementa Android, para compartir datos entre aplicaciones [20].

2.3.4.4. Broadcast Receiver

Un *broadcast receiver* es un componente destinado a detectar y reaccionar ante determinados mensajes o eventos globales, generados por el sistema (por ejemplo: “Batería baja”, “SMS recibido”, “Tarjeta SD insertada”, etc.) o por otras aplicaciones; cualquier aplicación puede generar mensajes [20].

2.3.5. Ciclo de vida de una *Activity*

De los cuatro componentes mencionados en el apartado anterior, es posible percibir que una "*Activity*" es el componente más común en las aplicaciones, ya que permiten interactuar con el usuario, una aplicación puede contener solo una "*Activity*" o más y cada una de ellas tendrá un ciclo de vida que es controlado, ya sea por el usuario o por el sistema de acuerdo a las necesidades de recursos del dispositivo.

Una vez que el usuario selecciona una aplicación para ser ejecutada, su ciclo de vida comienza, es decir la aplicación pasa por diferentes estados desde que se inicia hasta que se destruye. Normalmente se presta poca atención al ciclo de vida de una "*Activity*", y como consecuencia no es posible manejar correctamente las transiciones de estado. Por ejemplo, la "*Activity*" puede ser suspendida, ya sea debido a que el sistema hizo una pausa, esto se presenta, por ejemplo, cuando se recibe una llamada o el usuario decide terminar la ejecución de la aplicación. Así mismo, los requisitos específicos de memoria del sistema pueden obligar a la aplicación a manejar estos casos, e incluso es posible volver al estado en que estaba antes de la interrupción, como si nunca hubiera salido.

La figura 2.33, muestra el ciclo de vida de una "*Activity*", lo que hace es describir los estados en los que puede encontrarse una "*Activity*", los 3 posibles estados son:

- ❖ **Running o Ejecutándose:** Ocurre cuando la actividad está en ejecución, es decir, es la tarea principal o es la "*Activity*" que se muestra en la pantalla del usuario.
- ❖ **Paused o Pausado:** La actividad se encuentra semi-suspendida, es decir, aún se está ejecutando y es visible, pero no es la tarea principal, este estado se puede deber, por ejemplo, a que apareció un dialogo, o la pantalla del teléfono está bloqueada. En este estado el sistema Android puede decidir eliminar la "*Activity*", debido a problemas de memoria. Se debe guardar la información en este estado para prevenir una posible pérdida de datos en caso de que el sistema decida destruirla para liberar memoria.
- ❖ **Stopped o Parado:** La actividad está detenida, no es visible al usuario y el sistema, al igual que en estado "*Paused*", puede eliminarla para liberar memoria. En caso de necesitarla de nuevo, será reiniciada desde el principio.

Como se puede apreciar, en el estado "*Paused*" y en el estado "*Stopped*", el sistema puede eliminar la "*Activity*" sin previo aviso, pero mientras esto no ocurra la "*Activity*" sigue ocupando la misma memoria en la máquina virtual de Android y, por lo tanto, puede ser llevada de vuelta al estado "*Running*", desde cualquiera de estos dos estados, sin perder información de las variables que estaba manejando [20].

Las actividades cuentan con una serie de métodos, que pueden observarse en la figura 2.33, y son métodos que se pueden sobrescribir, es decir, realizar tareas a través de código, ya sea para estar informados de los cambios de estado, o para salvar algunas variables importantes que sean necesarias para la aplicación, [20].

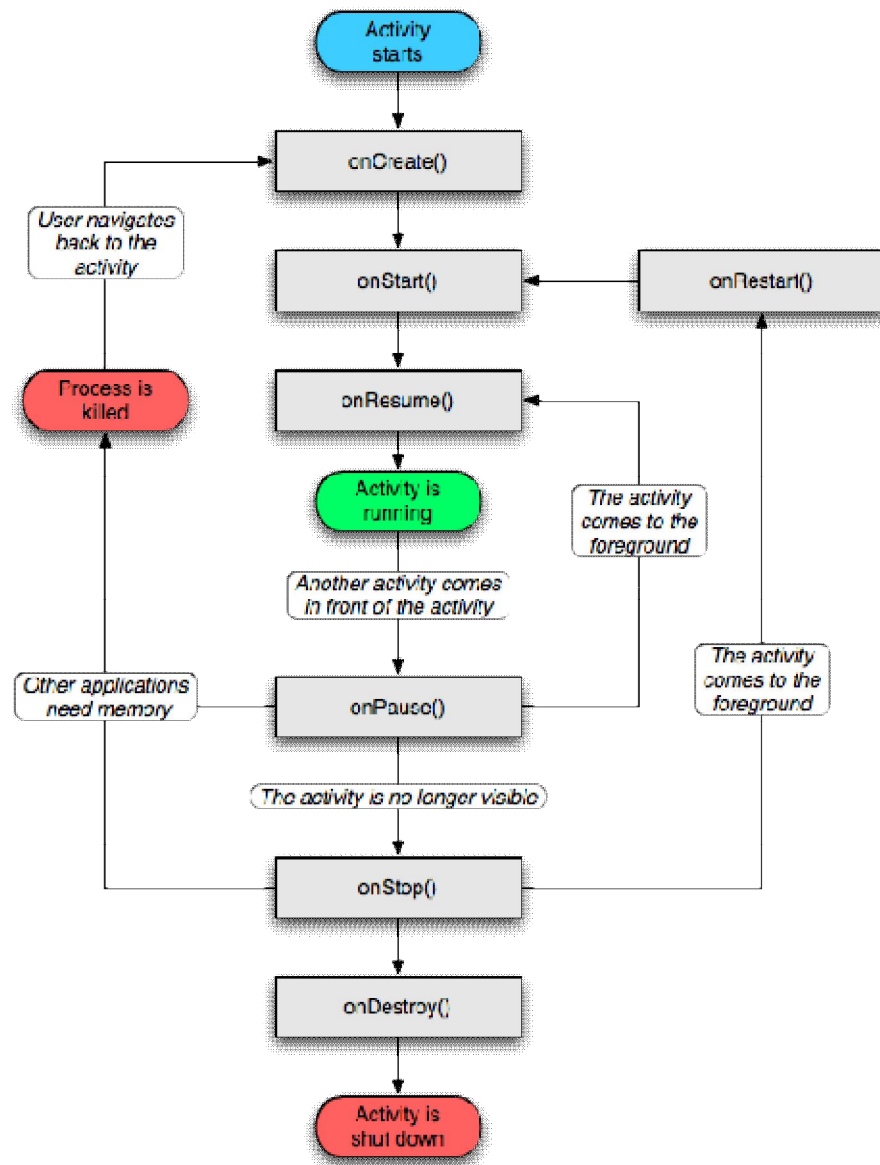


Figura 2.33 Ciclo de vida de una Activity.

onCreate(): Este método es llamado cuando la "Activity" es iniciada por primera vez y es aquí donde se configuran todos los componentes de la interfaz del usuario, además de instanciarlos o referenciarlos a algunas variables para poder manipular la interfaz.

onStart(): Es llamado después de "onCreate()", también puede ser llamado cuando la "Activity" es recuperada desde un estado "Stopped", si es así, antes hay que hacer una llamada al método "onRestart()", como se puede ver en la figura 2.33.

onResume(): Es llamado desde "onStart()" o cuando la actividad es recuperada desde un estado "Paused". Se ejecuta una vez que la "Activity" ha terminado de cargarse en el dispositivo y el usuario empieza a interactuar con la aplicación.

Cuando la "Activity" está en estado "Running" o ejecutándose, es cuando ha pasado por el método "onResume()". Se puede llegar de varios modos a esta situación. La primera forma es cuando la "Activity" se inicia y se debe de seguir la secuencia de métodos "onCreate()", "onStart()" y "onResume()". En este momento es cuando la "Activity" se está ejecutando, por tanto, es la que se muestra en la pantalla. A continuación se muestran los métodos necesarios para que la "Activity" desaparezca de la pantalla.

onPause(): Es el primer método en camino a que la "Activity" desaparezca de la pantalla, es llamado cuando la "Activity" entra en estado "Paused". Puede ser la última notificación que se reciba porque el sistema Android puede eliminar la "Activity" si tiene problemas de memoria, por tanto, es aquí donde se deben de guardar todas las variables que se consideren importantes para que sus valores sean tomados en cuenta al iniciarse de nuevo la aplicación, si es que el sistema elimina totalmente la "Activity". Se puede ver en la figura 2.33, que se puede salir del estado "Paused" llamando al método "onResume()".

onStop(): Es llamado cuando la "Activity" entra en estado "Stopped", y esta precedido siempre por una llamada al método "onPause()".

onRestart(): Es llamado cuando la "Activity" es recuperada desde un estado "Stopped" y tiene que estar siempre precedido por el método "onStop()", como se puede observar en la figura 2.33, este método nos permite devolver al primer plano la "Activity" que estaba detenida.

Sin embargo, es recomendable centrarse en los métodos realmente importantes para el desarrollo de una aplicación, esto permitirá no caer en confusiones; existen tres lecciones importantes para el buen manejo del ciclo de vida de una "Activity".

- ❖ **Lección 1:** El método "onResume()" siempre es llamado antes de que la aplicación entre en estado "Running", no importa si la "Activity" es iniciada por primera vez o es recuperada de un estado "Paused" o "Stopped", por tanto, para escribir el código, es posible ignorar los métodos "onStart()" y "onRestart()" y sólo concentrarse en el método "onResume()", pues este nos indicará que la "Activity" está a punto de ejecutarse, esto es posible verlo en la figura 2.33.

- ❖ **Lección 2:** La "Activity" puede ser destruida sin previo aviso después del método "onPause()", por tanto, no se puede tener la certeza de que los métodos "onStop()" y "onDestroy()" se ejecuten, debido a esta razón, los últimos dos métodos mencionados pueden ser ignorados y considerar al método "onPause()" como el último que se ejecuta en el ciclo de vida de la actividad.
- ❖ **Lección 3:** Teniendo en cuenta la lección número 2, es en el método "onPause()" donde se deben de almacenar todas las variables en la memoria no volátil, de tal manera que persistan en el caso de que la actividad, sea destruida e iniciada desde el principio.

Tomando esto en cuenta, los métodos principales de una "Activity" son "onCreate()", "onResume()" y "onPause()", en la tabla 2.10, se resumen las tareas principales a realizar en cada método principal y en la figura 2.34, muestra un trozo de código en donde se aprecian estos tres métodos.

Métodos Principales	Principales Tareas a Realizar
onCreate()	Instanciar los elementos de la interfaz del usuario.
	Configurar los elementos de la interfaz del usuario.
	Configurar elementos a intervenir como sensores, etc.
onResume()	Iniciar o reiniciar el hilo principal de la Activity.
	Leer variables de la memoria no volátil, en el caso de que se inicia la aplicación desde cero.
onPause()	Pausar el hilo principal.
	Almacenar variables que son necesarias para la aplicación en la memoria no volátil.

Tabla 2.10. Métodos principales de una Activity.

```

/**
 * Método para iniciar la Activity
 */
@Override
public void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    // Instanciar los elementos de la interfaz del usuario.
    // Configurar los elementos de la interfaz del usuario.
    // Configurar elementos a intervenir como sensores, etc.
}

/**
 * Método para iniciar la Activity
 */
@Override
protected void onResume()
{
    // TODO Auto-generated method stub
    super.onResume();

    // Iniciar o reiniciar el hilo principal de la Activity.
    // Leer variables de la memoria no volátil, en el caso de que se inicia
    // la aplicación desde cero.
}

/**
 * Método para pausar la Activity
 */
@Override
protected void onPause()
{
    // TODO Auto-generated method stub
    super.onPause();

    // Pausar el hilo principal.
    // Almacenar variables que son necesarias para la aplicación en la
    // memoria no volátil.
}

```

Figura 2.34. Código de los tres métodos principales de una Activity.

2.3.6. Introducción a la Tecnología GSM

La red **GSM** (*Sistema Global de Comunicaciones Móviles*) es el estándar más usado de Europa. Se denomina estándar "de segunda generación" (2G) porque, a diferencia de la primera generación de teléfonos portátiles, las comunicaciones se producen de un modo completamente digital.

En 1982, cuando fue estandarizado por primera vez, fue denominado "*Groupe Spécial Mobile*" y en 1991 se convirtió en un estándar internacional llamado "Sistema Global de Comunicaciones Móviles".

En Europa, el estándar GSM usa las bandas de frecuencia de 900MHz y 1800MHz. Sin embargo, en América se usa la banda de frecuencia de 1900 MHz. Por esa razón, los teléfonos portátiles que funcionan tanto en Europa como en los Estados Unidos se llaman *tribanda* y aquellos que funcionan sólo en Europa se denominan *bibanda*.

El estándar GSM permite un rendimiento máximo de 9.6 kbps, que permite transmisiones de voz y de datos digitales de volumen bajo, por ejemplo, mensajes de texto (**SMS**, *Servicio de mensajes cortos*) o mensajes multimedia (**MMS**, *Servicio de mensajes multimedia*) [25].

2.3.6.1. La Red Celular

Las redes de telefonía móvil se basan en el concepto de celdas, es decir, zonas circulares que se superponen para cubrir un área geográfica, figura 2.35.

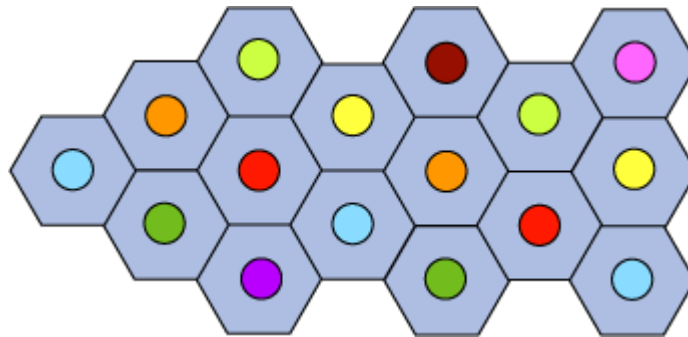


Figura 2.35. Celdas que cubren cierta área geográfica.

Las redes celulares se basan en el uso de un transmisor-receptor central en cada celda, denominado "Estación Base" o "Estación base transceptora", "*BTS*".

Cuanto menor sea el radio de una celda, mayor será el ancho de banda disponible. Por lo tanto, en zonas urbanas muy pobladas, hay celdas con un radio de unos cientos de metros, mientras que en zonas rurales hay celdas enormes de hasta 30 kilómetros que proporcionan cobertura.

En una red celular, cada celda está rodeada por 6 celdas contiguas (por esto las celdas generalmente se dibujan como un hexágono). Para evitar interferencia, las celdas adyacentes no pueden usar la misma frecuencia. En la práctica, dos celdas que usan el mismo rango de frecuencia deben estar separadas por una distancia equivalente a dos o tres veces el diámetro de la celda, figura 2.36.

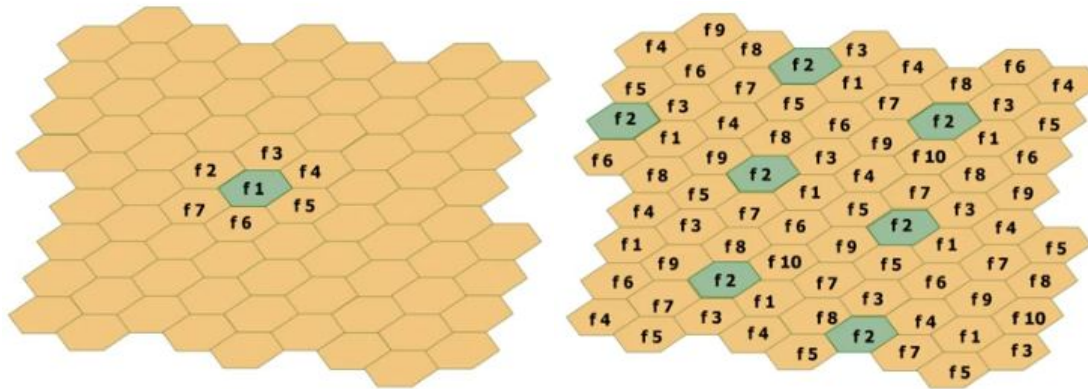


Figura 2.36. Distribución de frecuencias en cada celda.

2.3.6.2. Arquitectura de la Red GSM

En una red GSM, la terminal del usuario se llama "estación móvil". Una estación móvil está constituida por una tarjeta "SIM" (*Módulo de Identificación de Abonado*), que permite identificar de manera única al usuario y a la terminal móvil, o sea, al dispositivo del usuario (normalmente un teléfono portátil).

Las terminales se identifican por medio de un número único de identificación de 15 dígitos denominado "IMEI" (*Identificador Internacional de Equipos Móviles*). Cada tarjeta "SIM" posee un número de identificación único y secreto denominado "IMSI" (*Identificador Internacional de Abonados Móviles*). Este código se puede proteger con una clave de 4 dígitos llamada *código PIN*.

Por lo tanto, la tarjeta SIM permite identificar a cada usuario independientemente de la terminal utilizada durante la comunicación con la estación base. Las comunicaciones entre una estación móvil y una estación base se producen a través de un vínculo de radio, por lo general denominado *interfaz de aire*.

Todas las estaciones base de una red celular están conectadas a un "Controlador de Estaciones Base" conocido como "BSC", que administra la distribución de los recursos. El sistema compuesto del controlador de estaciones base y sus estaciones base conectadas, es el "Subsistema de Estaciones Base", "BSS".

Por último, los controladores de estaciones base están físicamente conectados al "Centro de Conmutación Móvil" (MSC), que los conecta con la red de telefonía pública y con Internet; lo administra el operador de la red telefónica. El "MSC" pertenece a un "Subsistema de Conmutación de Red" (NSS) que gestiona las identidades de los usuarios, su ubicación y el establecimiento de comunicaciones con otros usuarios.

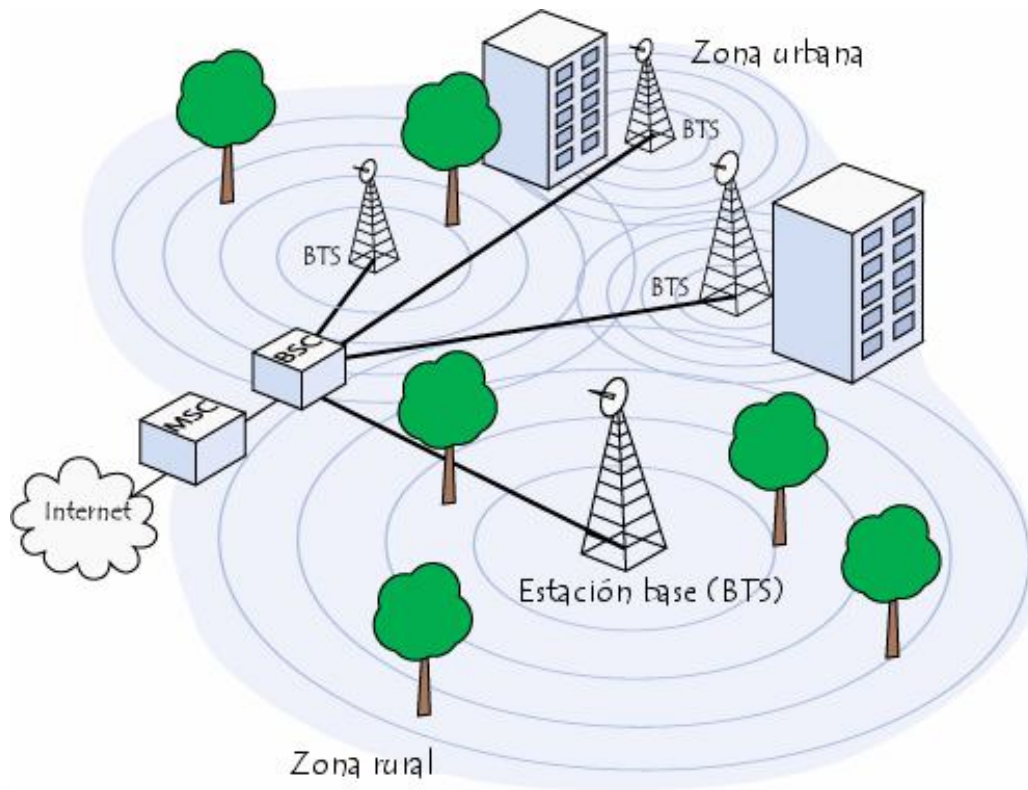


Figura 2.37. Arquitectura de la red GSM.

Generalmente, el MSC se conecta a bases de datos que proporcionan funciones adicionales:

- **El Registro de ubicación de origen (HLR):** es una base de datos que contiene información (posición geográfica, información administrativa, etc.) de los abonados registrados dentro de la zona del conmutador (MSC).
- **El Registro de ubicación de visitante (VLR):** es una base de datos que contiene información de usuarios que no son abonados locales. El "VLR" recupera los datos de un usuario nuevo del "HLR" de la zona de abonado del usuario. Los datos se conservan mientras el usuario está dentro de la zona y se eliminan en cuanto abandona la zona o después de un período de inactividad prolongada.
- **El Registro de identificación del equipo (EIR):** Es una base de datos que contiene la lista de terminales móviles.
- **El Centro de autenticación (AUC):** verifica las identidades de los usuarios.

La red celular compuesta de esta manera está diseñada para admitir movilidad, a través de la gestión de *traspasos* (movimientos que se realizan de una celda a otra).

Finalmente, las redes GSM admiten el concepto de "*roaming*"; el movimiento desde la red de un operador a otra, [25].

Con un "*smartphone*" es posible realizar casi todas las tareas que se hacen con una computadora de escritorio, sin embargo, uno de los componentes fundamentales de la plataforma es el teléfono móvil, el cual dispone de servicios para marcar, recibir llamadas, enviar y recibir mensajes de texto y multimedia, etc. En un teléfono inteligente, particularmente con sistema operativo "*Android*", la ventaja añadida es que todos los elementos propios de un teléfono celular son accesibles para los programadores a través de diversas instrucciones, es decir, es posible combinar o incluir la tecnología telefónica en diversas aplicaciones para el sistema "*Android*", [20].

3. IMPLEMENTACIÓN DEL SISTEMA

En esta sección se presenta el diseño e implementación de un robot móvil autónomo con configuración Ackerman, el cual es comandado por el usuario usando dispositivos y tecnologías móviles. Para realizar esto se seguirá el siguiente esquema:

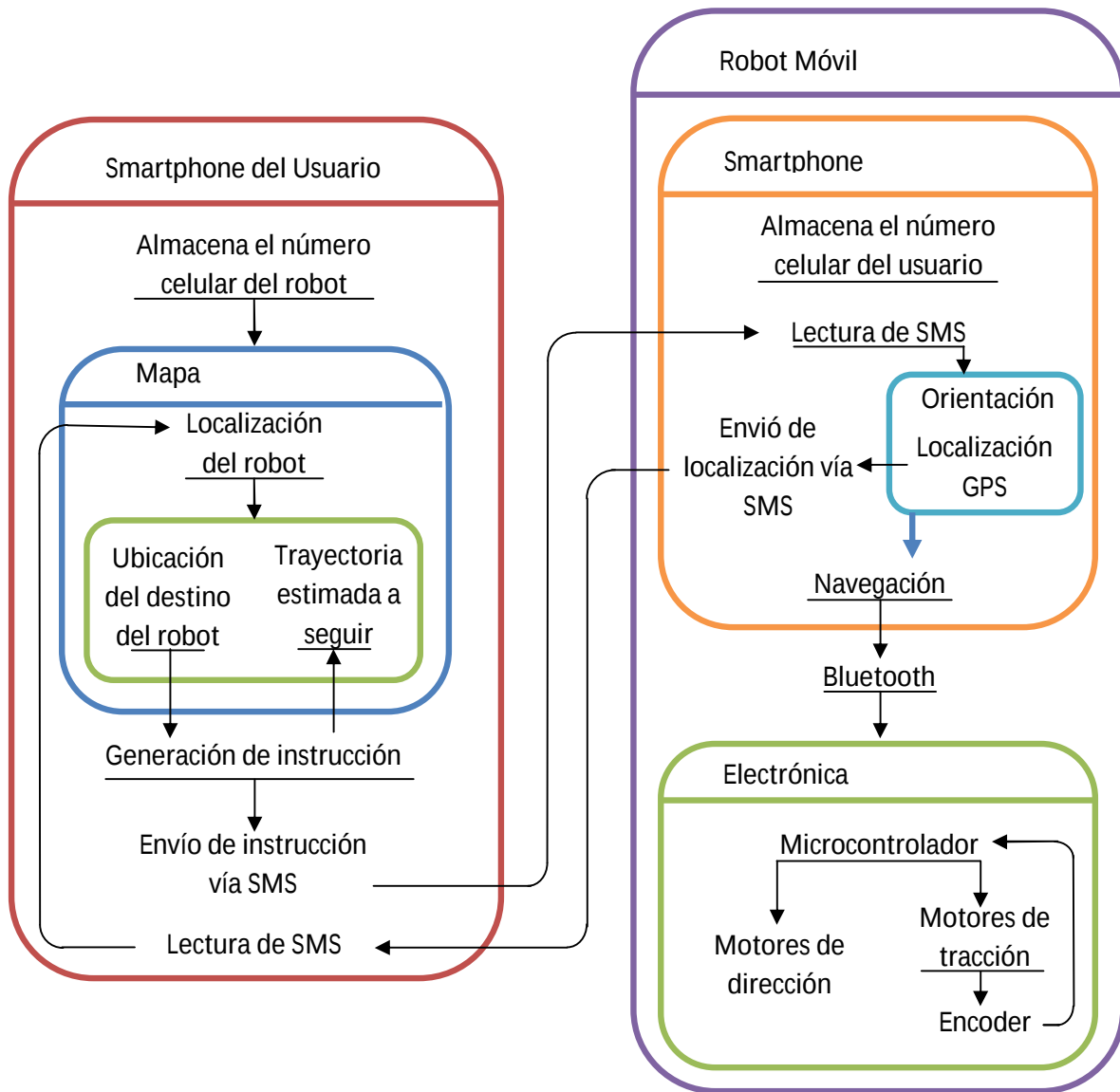


Figura 3.1 Arquitectura general del sistema.

3.1. Smartphone del Usuario

El usuario es el encargado de definir el destino del robot móvil, conociendo previamente su localización de éste, para ello se emplea una aplicación que corre en un "smartphone", esta "app" está programada en el lenguaje "java" en el entorno de programación "MOTODEV Studio for Android 4.1". Se trata de una aplicación móvil, similar a las

encontradas en la tienda "Play Store" para Android, estructurada en una sola "activity". Una "activity" representa una vista en la interfaz gráfica del usuario o pantalla, sin embargo, una "activity" puede contener varios módulos que estructuran su funcionamiento lógico, llamadas clases, y cada clase puede contener varios subprogramas que la conforman, estos subprogramas son llamados métodos, esta forma de agrupar subprogramas que tienen una cierta relación en una clase y varias clases en una aplicación, de tal forma que realicen una tarea, se conoce como "programación orientada a objetos".

La aplicación está basada en la interfaz de programación de aplicación, API, de "Google" que es una serie de servicios o funciones que el sistema operativo "Android" ofrece al programador, en la figura 3.2, se presenta la interfaz de usuario de esta aplicación.



Figura 3.2 Pantalla principal del usuario.

La tarea que desarrolla cada componente de la interfaz de usuario se describe en la tabla 3.1 y el funcionamiento principal de toda la aplicación se puede apreciar en la figura 3.3, este es un diagrama de secuencia que ilustra cómo interaccionan los elementos del programa, haciendo énfasis en el orden en el que ocurren los eventos a través del tiempo.

El robot móvil es operado para efectuar recorridos autónomos. En la pantalla principal del usuario se presenta un mapa de "Google Maps" en donde se ubica al prototipo de auto. Por medio de esta pantalla es posible seleccionar el punto de destino del robot y con tan

solo apretar el botón llamado "Operar" las coordenadas de latitud y longitud de la meta seleccionada son trasferidas al robot.

Componentes de la interfaz del usuario	
Elemento	Descripción
Botón Satélite	Permite cambiar la vista del mapa de modo satélite a mapa, mostrando una vista de la ciudad desde arriba.
Botón Navegar	Permite navegar sobre el mapa arrastrando el dedo sobre él, al tocar el botón se cambia esta función, permitiendo seleccionar una meta para el robot.
Botón Operar	Seleccionado un destino para el robot sobre el mapa, este botón transfiere las instrucciones hacia el robot vía SMS.
Campo de Edición	Al tocarlo se visualiza el teclado numérico para colocar el número celular del robot a 10 dígitos.
Botón Guardar	Verifica la estructura del número colocado en el campo de edición y lo guarda.
Mapa	Se visualiza un mapa obtenido desde "Google Maps".
Ubicación del robot	Punto que ubica al robot sobre el mapa.
Ubicación de la meta	Punto que ubica el destino final del robot sobre el mapa.
Trayectoria	Línea que dibuja la trayectoria estimada que el robot debe de seguir.

Tabla 3.1 Componentes de la interfaz del usuario.

La información pasa desde el "*smartphone*" del usuario hasta el robot móvil autónomo a través de las redes telefónicas comerciales. La información se genera desde el "*smartphone*" del usuario produciendo mensajes SMS, estos contienen texto que representan instrucciones para el robot. El "*smartphone*" del usuario, y el "*smartphone*" colocado sobre el robot móvil, son el medio de enlace para transmitir información.

3.2. Robot Móvil

El propósito del presente trabajo es el desarrollo de un algoritmo capaz de emplear dispositivos y tecnologías móviles para el seguimiento de trayectorias en autos autónomos, sin embargo, emplear un automóvil convencional implica adentrarse en diversas tecnologías que estas máquinas usan actualmente, además de un incremento en el costo de la investigación, es por ello que se emplea un robot móvil que contenga las características básicas que se requieren de los autos convencionales, para poder validar el sistema desarrollado.

El robot móvil a emplear en este trabajo es del tipo (1,1) y tiene la configuración conocida como Ackerman, la cual es la configuración que los automóviles actuales tienen, en dicho arreglo se incrementa la estabilidad lateral de la plataforma móvil, en curvas a alta velocidad, y su control es relativamente sencillo, considerando otras configuraciones, acosta de la disminución en su direccionalidad, es evidente que para que un automóvil convencional adapte una cierta postura, es necesario hacer varios movimientos, para más información se puede referir al apartado 2.2 de este trabajo, o consultar la bibliografía utilizada.

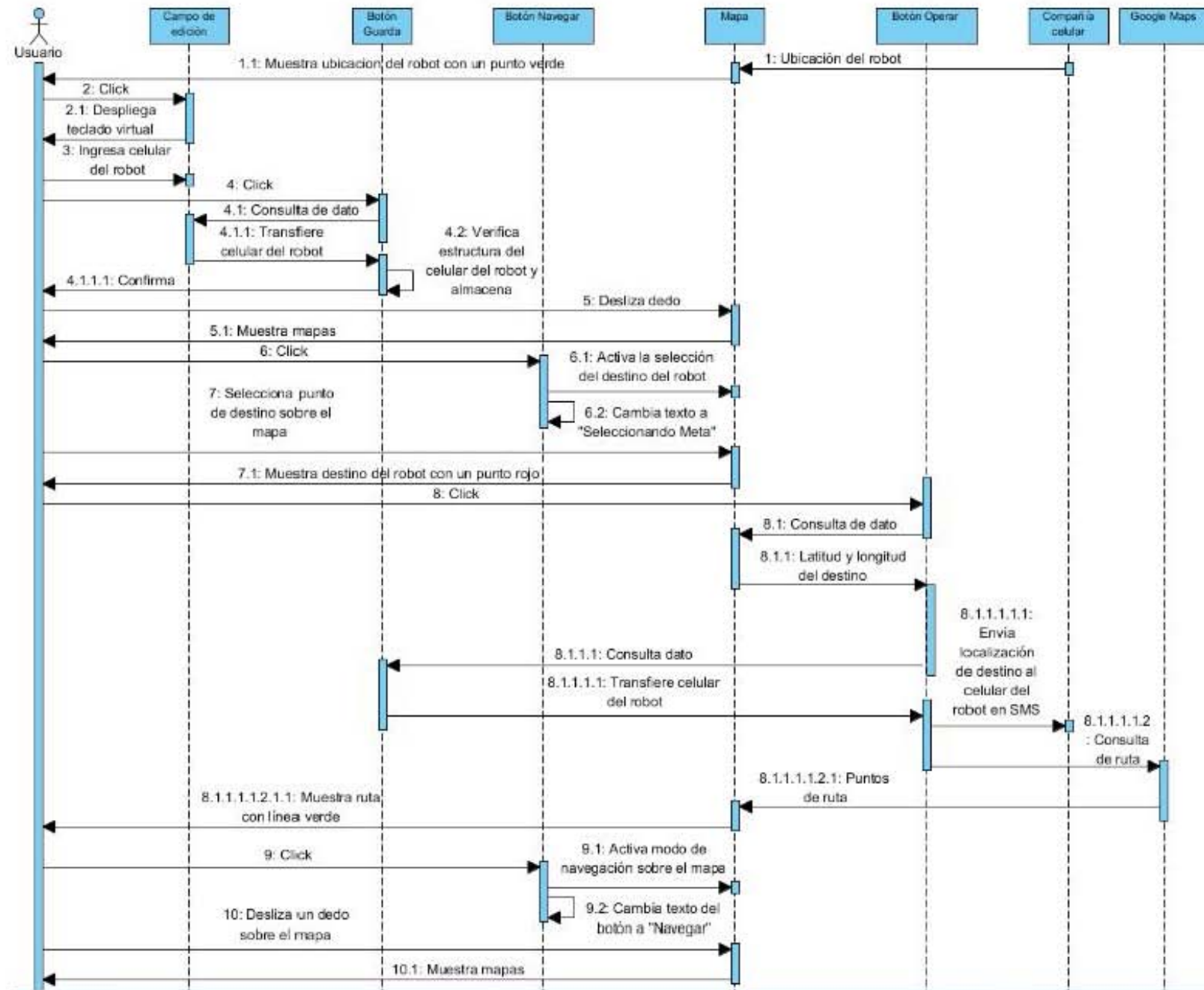


Figura 3.3. Diagrama de secuencia de la aplicación del usuario.

Particularmente, el robot a utilizar cuenta con los siguientes elementos; un chasis rígido de un material conocido como "*Sintra*"* que a la vez funciona como plataforma, en su parte posterior se encuentran dos llantas motorizadas centradas fijas, las cuales son actuadas de forma independiente utilizando un motor de corriente continua para cada una, en la parte de adelante de la plataforma se encuentran dos llantas centradas orientables, acopladas, cada una, a servomotores que las operan.

Sobre la plataforma se coloca la tarjeta electrónica que contiene al microcontrolador y todos sus periféricos necesarios para el control de los motores, además de la batería que provee energía a todos los motores y circuitos de control. Adherido a la plataforma, se encuentra una estructura rectangular, en la cual, en su parte superior, se encuentra un arreglo de tres sensores ultrasónicos y en la parte de adelante se tiene una pestaña igualmente pegada que permite colocar el "*smartphone*" que el robot lleva con sigo. En la figura 3.4 se puede observar una foto describiendo sus elementos.

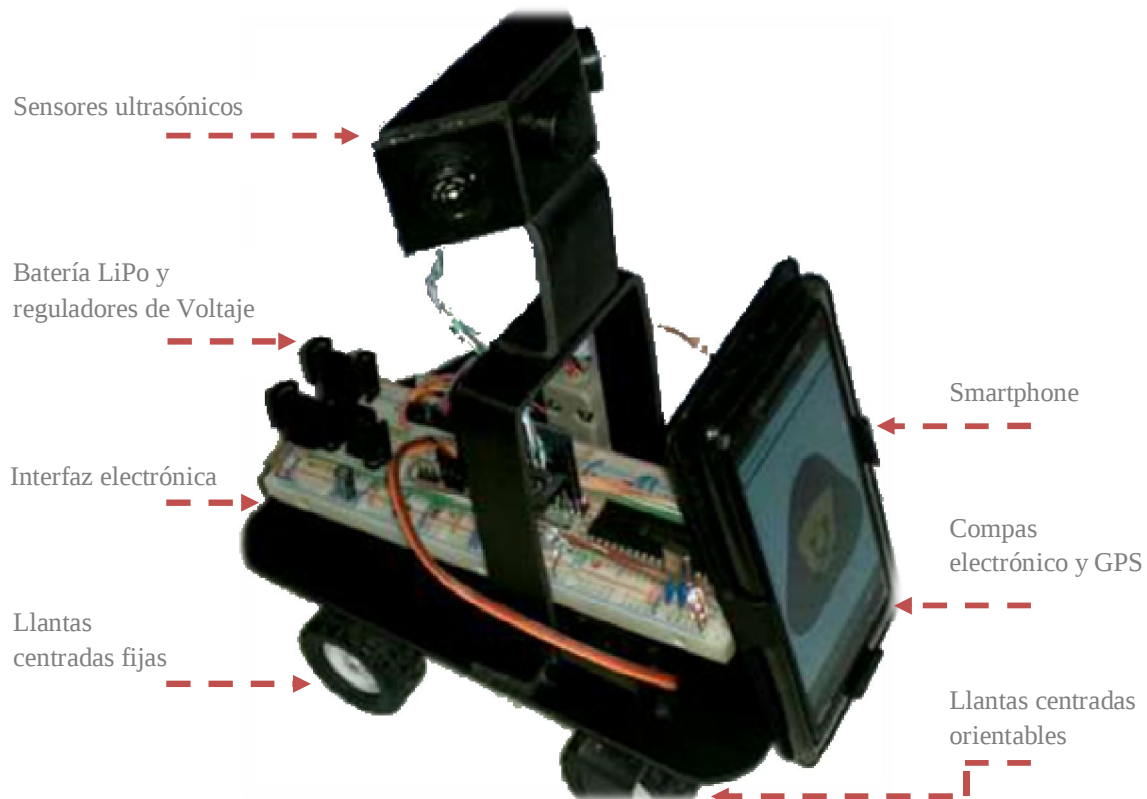


Figura 3.4. Robot móvil autónomo empleado.

En la sección 2.2.4 se presenta el análisis cinemático de este tipo de robot móvil, la figura 3.5, representa el esquema empleado para dicho análisis.

*Sintra: panel de espuma de cloruro de polivinilo (PVC) de células cerradas, siendo un material ligero, pero rígido, durable y resistente a los golpes.

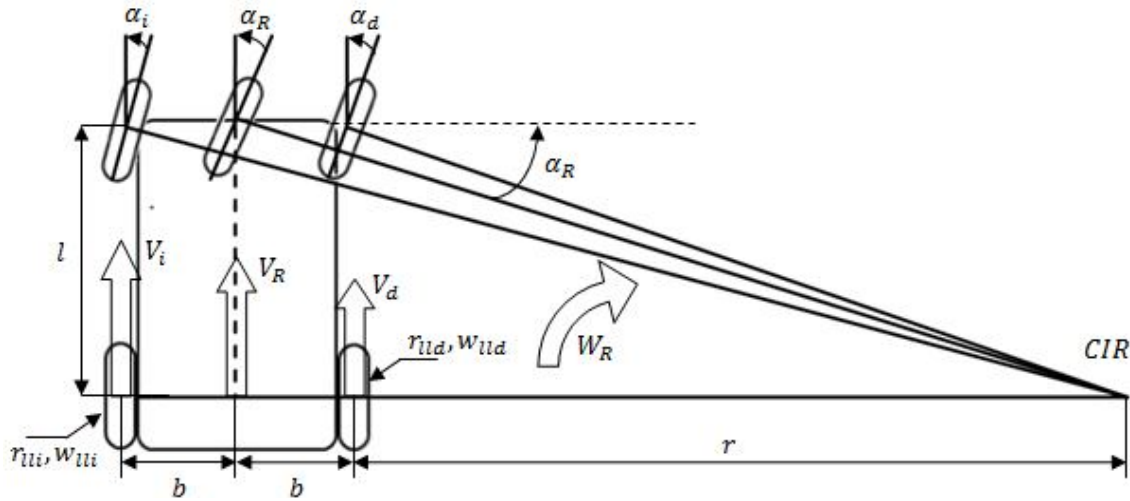


Figura 3.5. Robot móvil tipo Ackerman.

Las ecuaciones 2.85 y 2.86 representan las velocidades angulares de las llantas traseras derecha e izquierda del robot y las ecuaciones 2.96 y 2.97 representan las posiciones angulares de la llanta delantera derecha e izquierda, respectivamente.

$$w_{ld} = \frac{1}{r_{ld}} (V_R - W_R b) \dots \dots \dots 2.85$$

$$w_{li} = \frac{1}{r_{li}} (V_R + W_R b) \dots \dots \dots 2.86$$

$$\alpha_d = \text{atan} \left(\frac{l W_R}{V_R - W_R b} \right) \dots \dots \dots 2.96$$

$$\alpha_i = \text{atan} \left(\frac{l W_R}{V_R + W_R b} \right) \dots \dots \dots 2.97$$

En la tabla 2.7 se pueden ver los parámetros que implementan las anteriores ecuaciones y en la tabla 3.2 se aprecian los valores que estos parámetros toman para el caso específico del robot que se emplea.

Parámetros y sus valores de la cinemática del robot móvil		
Parámetro	Descripción	Valor
V_R	Velocidad lineal del robot	Valor máximo = 0.401 m/s
W_R	Velocidad angular del robot	Valor máximo = 4.41 rad/s
b	Un medio del ancho del robot móvil	0.05 m
$l = 2d$	Largo del robot móvil	0.15 m
r_{ld}	Radio de la llanta trasera derecha	0.02 m
r_{li}	Radio de la llanta trasera izquierda	0.02 m

Tabla 3.2. Parámetros y sus valores de la cinemática del robot móvil.

3.2.1. Smartphone del robot móvil

Un "smartphone", colocado sobre el robot móvil, es el dispositivo encargado de realizar el procesamiento necesario para la lectura de las instrucciones provenientes del usuario, el reconocimiento de la posición geográfica y de la orientación del robot, además de ejecutar el algoritmo empleado para la navegación autónoma [5]. Se trata de una aplicación móvil que está basada en la API de "Google Maps", en la figura 3.6, se presenta la interfaz del usuario de esta aplicación.

La tarea que desarrolla cada componente de la interfaz de usuario para el robot móvil se describe en la tabla 3.3.

La aplicación desarrollada es capaz de encontrar rutas de forma automática, basándose en la información del GPS, del destino dado por el usuario y de datos obtenidos de "Google Maps", la arquitectura de esta aplicación se describe en la figura 3.7, consta de una sola "activity" distribuida en varias clases; **MainActivity**, **MapRoute** y **Comandos**, estas implementan diversos parámetros del robot como sus longitudes, así como datos del GPS y del sensor de orientación que trae consigo el robot.

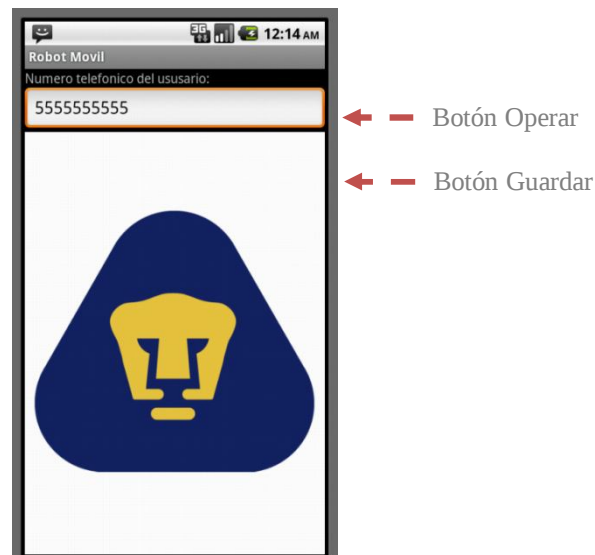


Figura 3.6. Pantalla principal del usuario para el robot móvil.

Componentes de la pantalla del robot móvil	
Elemento	Descripción
Campo de Edición	Al tocarlo se visualiza el teclado numérico para colocar el número celular del usuario a 10 dígitos.
Botón de Conexión	Verifica la estructura del número colocado en el campo de edición y lo guarda, posteriormente inicia la conexión bluetooth con el robot para controlar sus motores.

Tabla 3.3. Componentes de la interfaz del usuario para el robot móvil.

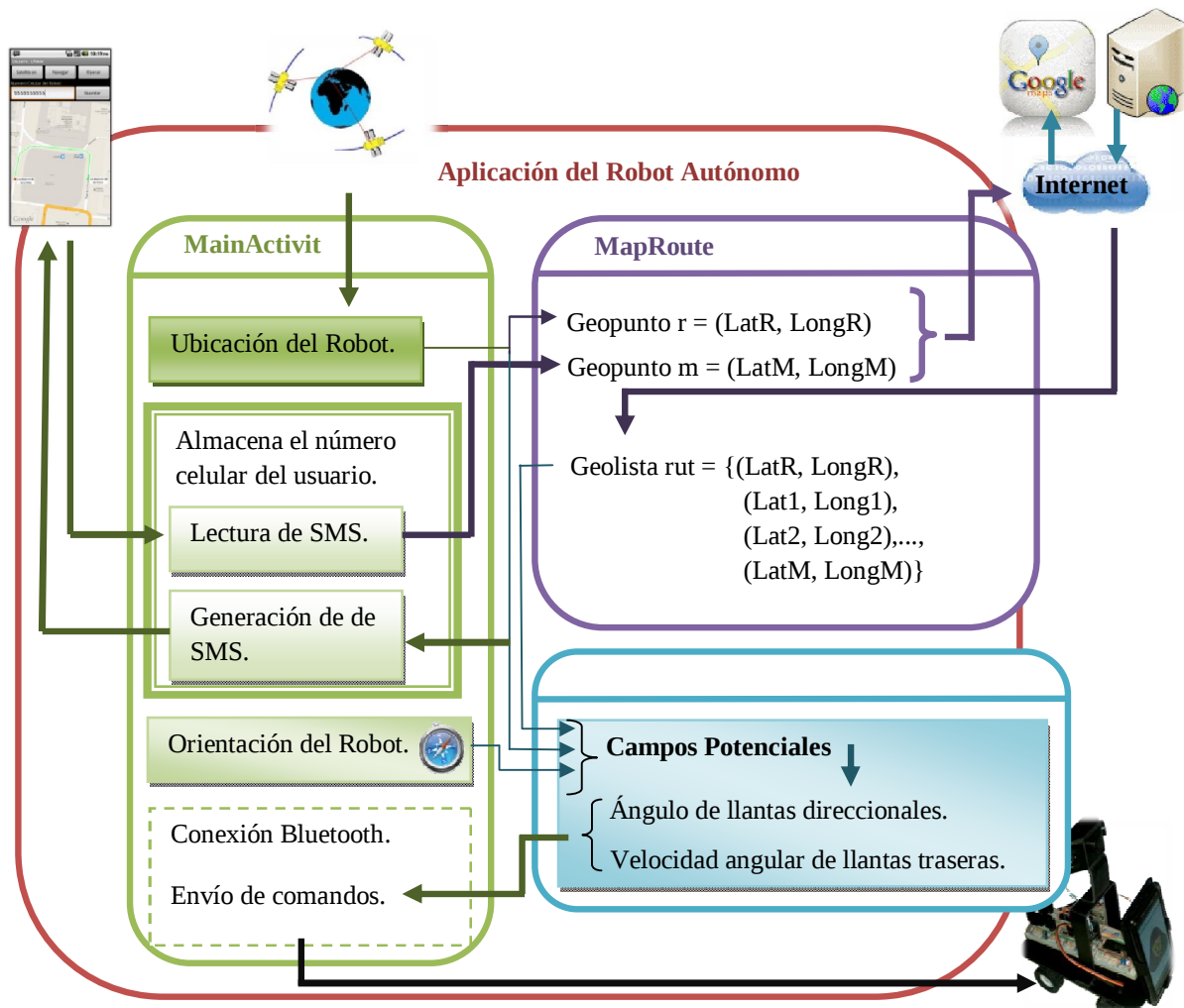


Figura 3.7. Arquitectura de la aplicación del robot móvil.

En la figura 3.9, se puede ver un diagrama de secuencia de la aplicación del robot móvil, con la cual se ilustra cómo interaccionan los elementos del programa, haciendo énfasis en el orden en el que ocurren los eventos a través del tiempo.

En la pantalla principal del usuario se presenta un campo de edición, donde se coloca el número celular del usuario, y un botón, que cubre la mayor parte de la pantalla. Colocado el número celular del usuario, al tocar el botón, el "smartphone" del robot se conecta a la tarjeta electrónica, empleando una conexión bluetooth para empezar a comandar sus motores, mandando información al microcontrolador.

3.2.1.1. Ubicación y Orientación del Robot Móvil

El "smartphone" del robot móvil contiene sensores equipados en él, dos de los cuales son un GPS y un sensor de navegación, entre otros, al establecer una conexión con el robot y al verificar que el GPS está habilitado, este empieza a ubicarse en coordenadas geográficas, es decir, en coordenadas de latitud y longitud, localizando de manera indirecta al robot, en la figura 3.8, se puede apreciar la proyección cilíndrica del mundo, particularmente la proyección de Mercator, usada por "Google Maps", esta proyección es muy buena para preservar los ángulos en los mapas, pero no es buena para la visualización de las áreas del mundo cerca de los polos Norte y Sur, [15], para mayor información se puede consultar la sección 2.1. Estas coordenadas de latitud y longitud ubican al robot y son enviadas al dispositivo del usuario para que, en su mapa, se señale su ubicación de manera gráfica.

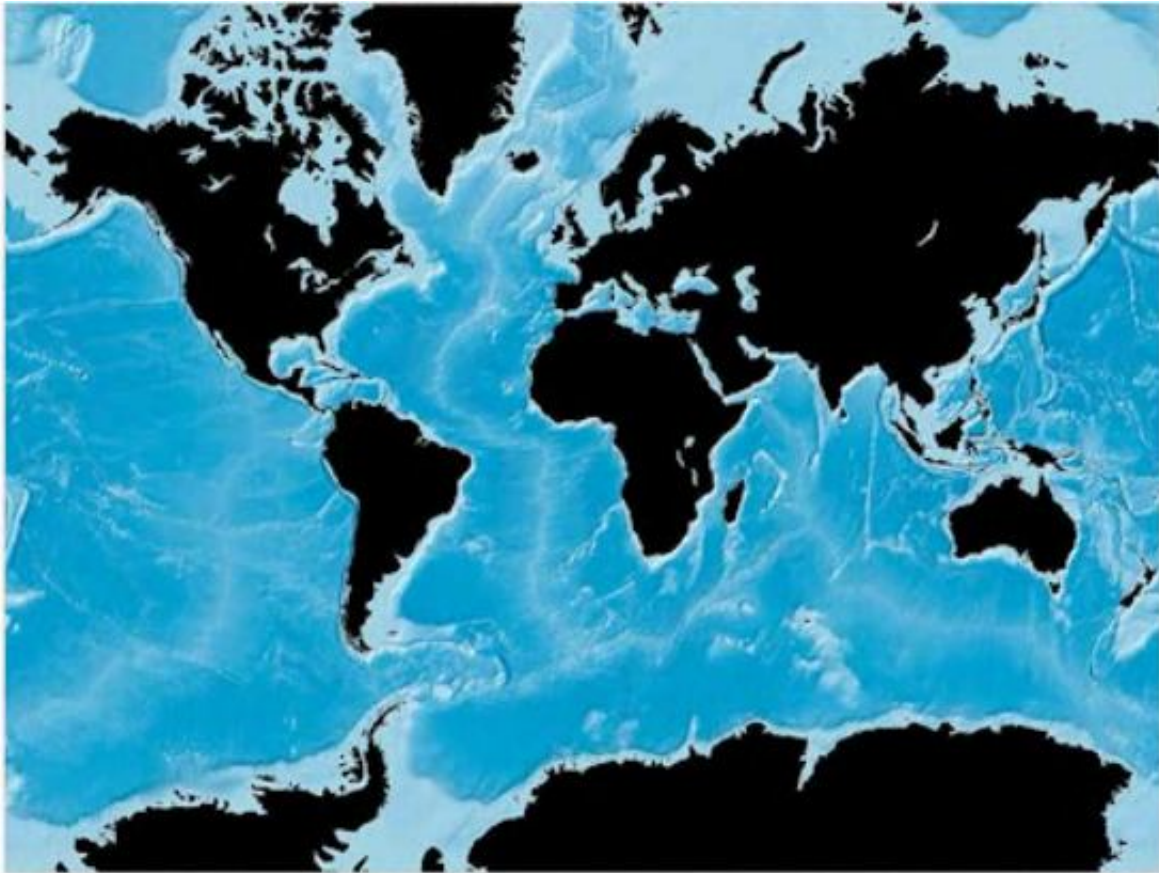


Figura 3.8. Proyección de Mercator del mundo.

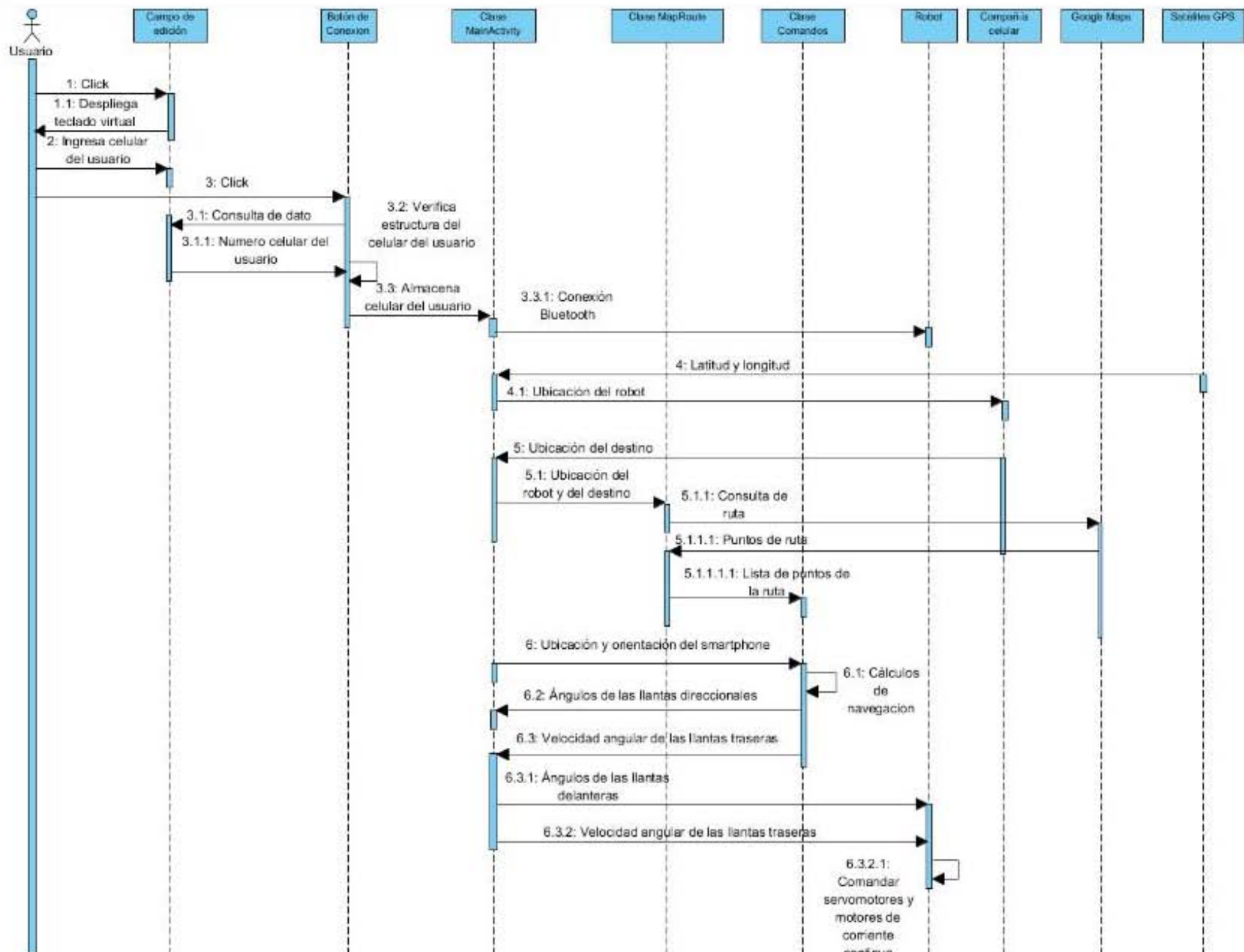


Figura 3.9. Diagrama de secuencia de la aplicación del robot móvil.

Para orientar al robot, se hace uso del sensor de orientación que trae incorporado el "smartphone", este sensor es capaz de ofrecer, en grados, la información de azimut, "pitch" y "roll" del dispositivo móvil, de acuerdo a la información que se encuentra disponible en "<http://developer.android.com/reference/android/hardware/SensorListener.html>" [23], el "azimut" está definido como la rotación alrededor del eje "Z", el "azimut", figura 3.10, es una dirección que tiene como origen la línea norte, pasa por el este a un ángulo de 90°, posteriormente toca el sur con 180° y el oeste cuando tiene un valor de 270°, el "pitch" se define como la rotación alrededor del eje "X" y el "roll" es la rotación alrededor del eje "Y". Los ejes, para el caso de la orientación del "smartphone", tienen el siguiente sentido:

- ❖ El eje "X". tiene como origen la parte izquierda inferior y apunta hacia la derecha, es decir, hacia la esquina inferior derecha de la pantalla.
- ❖ El eje "Y": tiene como origen la esquina izquierda inferior de la pantalla y apunta hacia la esquina superior izquierda de la pantalla.
- ❖ El eje "Z": tiene como origen la esquina izquierda inferior de la pantalla y apunta hacia el cielo.

Para esta aplicación, particularmente, se hace uso de los datos que se obtienen del grado de rotación del "azimut".

Conociendo como está colocado el teléfono inteligente en el robot móvil, es posible conocer la orientación del robot, de esta manera, con el GPS del celular conocemos la ubicación del robot en coordenadas absolutas de latitud y longitud, en una proyección cilíndrica, y con el sensor de navegación, particularmente con el ángulo de "azimut", conocemos la orientación del robot.

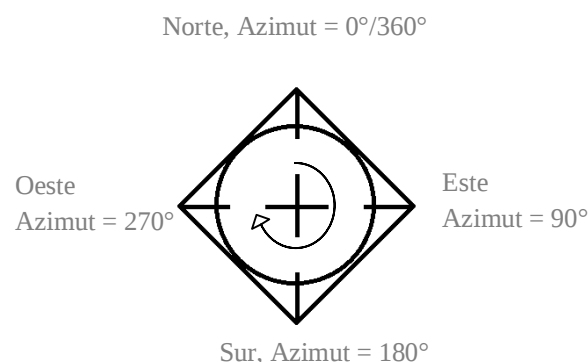


Figura 3.10. Ángulo de azimut.

En la figura 3.11, se puede observar el código básico para hacer uso del GPS y en las figuras 3.12 y 3.13, se presenta el código básico para hacer uso de algunos sensores, entre ellos el sensor de orientación, todo esto para dispositivos con sistema Android.

```

//Variables globales
double latitud=0.0;
double longitud=0.0;
/**
 * Método para iniciar la Activity
 */
@Override
public void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    //Accediendo al servicio de localización
    LocationManager milocManager =
        (LocationManager) getSystemService(Context.LOCATION_SERVICE);

    LocationListener milocListener = new LocationListener()
    {
        @Override
        public void onLocationChanged(Location loc)
        {
            //Si la localización cambia, se almacena la latitud y la
            //longitud en las variables globales_
            latitud = loc.getLatitude();
            longitud = loc.getLongitude();
        }
        @Override
        public void onProviderDisabled(String provider)
        {
            //Acciones a realizar si el GPS está apagado.

            //Normalmente se manda a la pantalla de configuración para
            //que el usuario prenda el GPS.
            Intent in = new
            Intent(android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);

            startActivity(in);
        }
        @Override
        public void onProviderEnabled(String provider)
        {
            //Acciones a realizar si el GPS está prendido.
        }
        @Override
        public void onStatusChanged(String provider, int status,
                                   Bundle extras)
        {
            //Acciones a realizar si el GPS pasa de prendido ha apagado
            //o viceversa.
        }
    };

    //Acedemos a la localización a través del GPS
    milocManager.requestLocationUpdates( LocationManager.GPS_PROVIDER,
                                         0, 0, milocListener);
}

```

Figura 3.11. Código básico para el uso del GPS en dispositivos con sistema Android.

```

/**
 * Método para iniciar la Activity
 */
@Override
public void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    //Obtenemos una referencia de la clase SensorManager
    //la cual dispone de los sensores
    SensorManager sensor =
        (SensorManager) getSystemService(Context.SENSOR_SERVICE);

    //Con la referencia "sensor" ya podemos interactuar con los sensores

    //ACELEROMETRO
    //Obtenemos una variable que hace referencia al sensor y le damos su
    tiempo de monitoreo

    /**
     *SENSORES QUE GENERALMENTE SIEMPRE ESTAN PRESENTES EN TODOS LOS CELULARES
     *ANDROID
     */
    Sensor acelera = sensor.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
    sensor.registerListener(this, acelera, SensorManager.SENSOR_DELAY_NORMAL);

    //APROXIMIDAD
    Sensor cercania = sensor.getDefaultSensor(Sensor.TYPE_PROXIMITY);
    sensor.registerListener(this, cercania, SensorManager.SENSOR_DELAY_NORMAL);

    //ORIENTACIÓN
    Sensor orienta = sensor.getDefaultSensor(Sensor.TYPE_ORIENTATION);
    sensor.registerListener(this, orienta, SensorManager.SENSOR_DELAY_NORMAL);

    /**
     *ALGUNOS SENSORES PRESENTES EN CELULARES DE GAMA ALTA
     */
    //GIROSCOPIO
    Sensor giro = sensor.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
    sensor.registerListener(this, giro, SensorManager.SENSOR_DELAY_NORMAL);

    //TEMPERATURA AMBIENTE
    Sensor tempe = sensor.getDefaultSensor(Sensor.TYPE_TEMPERATURE);
    sensor.registerListener(this, tempe, SensorManager.SENSOR_DELAY_NORMAL);
    //ALGUNOS SENSORES PRESENTES EN CELULARES DE GAMA ALTA

}

@Override
public void onAccuracyChanged(Sensor sensor, int accuracy)
{
}

```

Figura 3.12. Código básico para usar algunos sensores en el sistema Android

```

@Override
public void onSensorChanged(SensorEvent event)
{
    //Este método se manda llamar automáticamente, como una interrupción, cada vez
    //que pasa un tiempo determinado por la constante "delay_normal" que es el
    //argumento del método registerListener de la clase "SesnorManger"

    //Para detectar que se produjo un cambio en el sensor de proximidad
    if (event.sensor.getType() == Sensor.TYPE_PROXIMITY)
    {
        //La variable "Valor" almacena la información del sensor de
        //proximidad
        float valor = event.values[0];

        //Acciones a realizar, dependiendo del contenido de la variable
        //"valor"
    }

    //Para detectar que se produjo un cambio en el sensor de aceleración
    if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER)
    {
        float aceleracionEnX = event.values[0];
        float aceleracionEnY = event.values[1];
        float aceleracionEnZ = event.values[2];

        //Acciones a realizar, dependiendo del contenido de las variables
    }

    //Para detectar que se produjo un cambio en el sensor de orientación
    if (event.sensor.getType()==Sensor.TYPE_ORIENTATION)
    {
        float azimut = event.values[0];
        float pitch = event.values[1];
        float roll = event.values[2];

        Acciones a realizar, dependiendo del contenido de las variables
    }

    //Para detectar que se produjo un cambio en el sensor giroscopio
    if (event.sensor.getType()==Sensor.TYPE_GYROSCOPE)
    {
        float velocidadAngularAlrededorDeX = event.values[0];
        float velocidadAngularAlrededorDeY = event.values[1];
        float velocidadAngularAlrededorDeZ = event.values[2];

        Acciones a realizar, dependiendo del contenido de las variables
    }

    //Para detectar que se produjo un cambio en el sensor de temperatura
    if (event.sensor.getType()==Sensor.TYPE_TEMPERATURE)
    {
        //Temperatura en Celcius
        float temperatura = event.values[0];
        Acciones a realizar, dependiendo del contenido de la variable
    }
}

```

Figura 3.13. Código básico para obtener el valor de algunos sensores en el sistema Android.

3.2.1.2. Navegación

La información del destino del robot pasa desde el "smartphone" del usuario, hasta el robot móvil autónomo a través de las redes telefónicas comerciales. Al recibir el destino y al conocer su ubicación, gracias al GPS incorporado dentro del "smartphone", este consulta a "Google Maps" la ruta que se debe de seguir pasándole como información su localización y su destino, "Google Maps" regresa una serie de puntos, que forman la ruta a seguir, posteriormente, con el algoritmo detallado en [5], alimentado con la información procedente del GPS y del sensor de orientación del "smartphone", se empiezan a comandar los motores traseros y los servomotores delanteros de la plataforma, de tal manera que se alcancen los puntos obtenidos por "Google Maps", de manera sistemática y, por consecuencia, se vaya recorriendo la trayectoria deseada. Adicionalmente el "smartphone" del robot realimenta al usuario informándole de su posición.

Como se mencionó en el apartado 3.2.1, la aplicación está formada de tres clases, cada una de ellas realiza una tarea importante para la navegación del robot. A continuación se realiza una breve explicación del funcionamiento de cada clase:

MainActivity: Se trata de la clase principal, coordina a las demás clases y también está encargada de guardar el número telefónico del usuario, para que el robot le envíe información de su posición y reciba instrucciones, además se encarga de la conexión bluetooth con el robot, mandando las instrucciones para cada motor del robot, vigila los cambios del GPS y del sensor de orientación, además almacena toda esta información para su análisis posterior; en la figura 3.14, se puede ver un bloque de código básico para el envío de mensajes SMS, se puede apreciar la clase principal "MainActivity" que alberga al método "onCreate()" en donde se crean dos variables locales, es decir, sólo existen dentro del método donde fueron creadas, estas variables son "msg" y "número", la primera alberga el mensaje y la segunda variable contiene el número celular del destinatario, posteriormente se manda el mensaje a través del objeto "sms" que referencia a la clase "SmsManager".

MapRoute: Consulta la ubicación del robot y la almacena en una estructura conocida como "Geopunto"¹, de igual manera, esta clase recibe la ubicación de la meta en coordenadas de latitud y longitud y la guarda en otro "Geopunto"¹ que determinará el destino o la meta del robot, al tener estos dos datos se realiza una consulta a los

¹Geopunto: Variable que almacena las coordenadas de latitud y longitud de un objeto, por ejemplo, si la variable "localiza" almacena la ubicación del robot de la siguiente manera; *localiza* = (*latitud de robot*, *longitud de robot*), entonces la variable "localiza", es conocido como un "Geopunto".

servidores de "Google Maps" para generar una trayectoria, desde el punto donde se encuentra el robot hasta la meta, esta trayectoria que regresa la consulta a "Google Maps" está formada por una serie de "Geopuntos"¹ que están almacenados en un arreglo conocido como "Geolista"², se podría entender que la trayectoria obtenida estaría formada por varias submetas para el robot, de tal manera que al ir alcanzando cada una de ellas, se estaría recorriendo la ruta formada hasta llegar a la meta seleccionada por el usuario.

```
public class MainActivity extends Activity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        //Variable que contiene el mensaje
        String msg = "hola";

        //Variable que contiene el número del destinatario
        String numero = "5555555555";

        //Obtenemos una referencia de la clase SmsManager
        //la cual dispone de los métodos necesarios para el envío de
        // mensajes SMS
        SmsManager sms = SmsManager.getDefault();
        sms.sendTextMessage(numero, null, msg, null, null);
    }
}
```

Figura 3.14. Código básico para mandar mensajes SMS.

Comandos: Es responsable de generar las instrucciones que serán enviadas al robot para controlar los motores, estas instrucciones están basadas en un algoritmo conocido como "*campos potenciales*", el cual es desarrollado a detalle por el Dr. Víctor J. González Villela en el artículo con nombre "*A wheeled mobile robot with obstacle avoidance capability*", [5], éste es un algoritmo de navegación para robots móviles, el cual necesita como información, la orientación y ubicación del robot así como la ubicación de la meta, estos datos son obtenidos de las clases anteriormente descritas, en cuanto a la ubicación de la meta, esta estaría formada por los "Geopuntos" que se obtendrían de la "Geolista", que se formó en la clase anterior, de tal manera que al ir recorriendo esta lista se estaría recorriendo la trayectoria hasta llegar a la meta final.

A continuación se realiza una explicación de los cálculos realizados por el algoritmo encontrado en [5].

²Geolista" es un arreglo que almacena una serie de "Geopuntos".

Obtenidas las coordenadas de latitud y longitud de la meta y del robot móvil así como su ángulo de orientación, a través del GPS y del sensor de orientación del "smartphone", referenciados al cero del sistema manejado por el GPS y al ángulo de azimut, ver sección 2.1.6 y 3.2.1.1, es necesario pasar su ubicación a metros, esto se logra multiplicando su coordenada de latitud y longitud, en formato de grados con decimales por la cantidad de metros que corresponde a cada grado de latitud y longitud, como se indica en la sección 2.1.6.5, particularmente en la tabla 2.4.

Con la información anterior, se procede a realizar los cálculos correspondientes para las velocidades angulares de sus llantas traseras y el ángulo de dirección de las llantas delanteras del robot, de tal manera que se logre dirigir al punto determinado como meta.

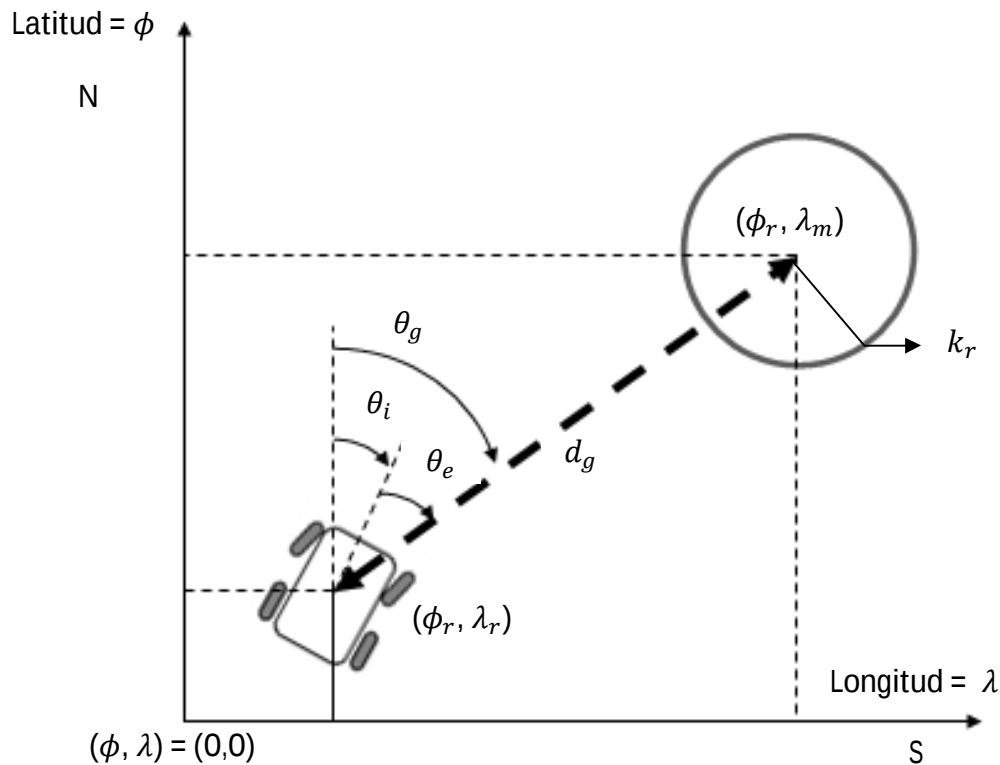


Figura 3.15. Posición del robot y la meta a alcanzar, adaptado de [5].

En la figura 3.15, es posible observar los parámetros que se involucran para llevar al robot de una ubicación a otra, para ello, el primer paso es calcular la distancia que separa al robot defensor de su meta, esto se realiza aplicando la ecuación 3.1, calculada esta distancia, que representa una fuerza virtual de atracción que dirige al robot, se procede a calcular el ángulo θ_g , que es el ángulo entre la posición del robot y su objetivo, esto se realiza empleando la ecuación número 3.2.

$$d_g = \sqrt{(\phi_m - \phi_r)^2 + (\lambda_m - \lambda_r)^2} \dots \dots \dots 3.1$$

Sin embargo, el robot tiene un ángulo con respecto al norte terrestre, obtenido gracias al sensor de orientación del "smartphone", por lo tanto se debe de calcular el ángulo que realmente debe de girar desde su orientación actual, θ_i , hasta orientarse a su objetivo, este ángulo está dado por la ecuación 3.3 y es representado por θ_e .

$$\theta_g = \text{Arctan}\left(\frac{\lambda_m - \lambda_r}{\phi_m - \phi_r}\right) \dots \dots \dots 3.2$$

$$\theta_e = \theta_g - \theta_i \dots \dots \dots 3.3$$

Con las variables dadas por las ecuaciones 3.1 y 3.3, proseguimos a calcular la velocidad lineal y angular del robot móvil:

Para la velocidad lineal:

$$V_R = \begin{cases} V_{max} & \text{si } |d_g| > k_r \\ \frac{V_{max}}{k_r} d_g & \text{si } |d_g| \leq k_r \end{cases} \dots \dots \dots 3.4$$

Donde:

V_{max} : Velocidad máxima del robot.

V_R : Velocidad lineal del robot.

k_r : Es el radio del área para que el robot frene.

Para la velocidad angular:

$$W_R = W_{max} * \text{Sen}(\theta_e) \dots \dots \dots 3.5$$

Donde:

W_{max} : Velocidad angular máxima del robot.

W_R : Velocidad angular del robot.

Obtenidos V_R y W_R , se procede a realizar los cálculos necesarios para calcular las velocidades angulares de las llantas traseras del robot y el ángulo que tomarán las llantas delanteras del robot, en la sección 2.2.4, se explica a detalle cómo conseguir las expresiones que fijan estos valores para la arquitectura del robot que se está empleando.

Las ecuaciones 2.85 y 2.86 representan las velocidades angulares de la llanta trasera derecha e izquierda del robot y las ecuaciones 2.96 y 2.97 representan la posición angular de la llanta delantera derecha e izquierda, respectivamente.

$$w_{lld} = \frac{1}{r_{lld}} (V_R - W_R b) \dots \dots \dots 2.85$$

$$w_{lli} = \frac{1}{r_{lli}} (V_R + W_R b) \dots \dots \dots 2.86$$

$$\alpha_d = \text{atan} \left(\frac{lW_R}{V_R - W_R b} \right) \dots \dots \dots 2.96$$

$$\alpha_i = \text{atan} \left(\frac{lW_R}{V_R + W_R b} \right) \dots \dots \dots 2.97$$

En la tabla 2.7, se pueden ver los parámetros que implementan las anteriores ecuaciones y en la tabla 3.2, se aprecian los valores que estos parámetros toman para el caso específico del robot que se emplea. Por último los valores obtenidos por las ecuaciones 2.85, 2.86, 2.96 y 2.97, son enviados a la electrónica del robot por medio de una interfaz bluetooth.

3.2.2. Interfaz Electrónica

Entendido como es que se realiza la navegación, debe diseñarse la interfaz electrónica que permita conectar el robot móvil al "smartphone". Esta tarjeta fue desarrollada para recibir los caracteres enviados por el "smartphone" de forma inalámbrica vía bluetooth para, posteriormente, comandar a los cuatro motores que ayudan a desplazar al robot móvil.

Se basa en un microcontrolador PIC16F877A, en donde se programa la lógica de operación, un chip L293D se utiliza como interfaz de potencia para comandar los motores de corriente directa que actúan a las llantas centradas fijas traseras, un encóder, en cada una de estas llantas, se usa para realimentar al algoritmo del microcontrolador y poder vigilar su velocidad angular, por otro lado, dos servomotores son utilizados para comandar las llantas centradas orientables que direccionan al robot, tres sensores ultrasónicos se utilizan para detectar algún posible obstáculo en su camino, en la figura 3.16, se puede observar el esquema electrónico de esta tarjeta.

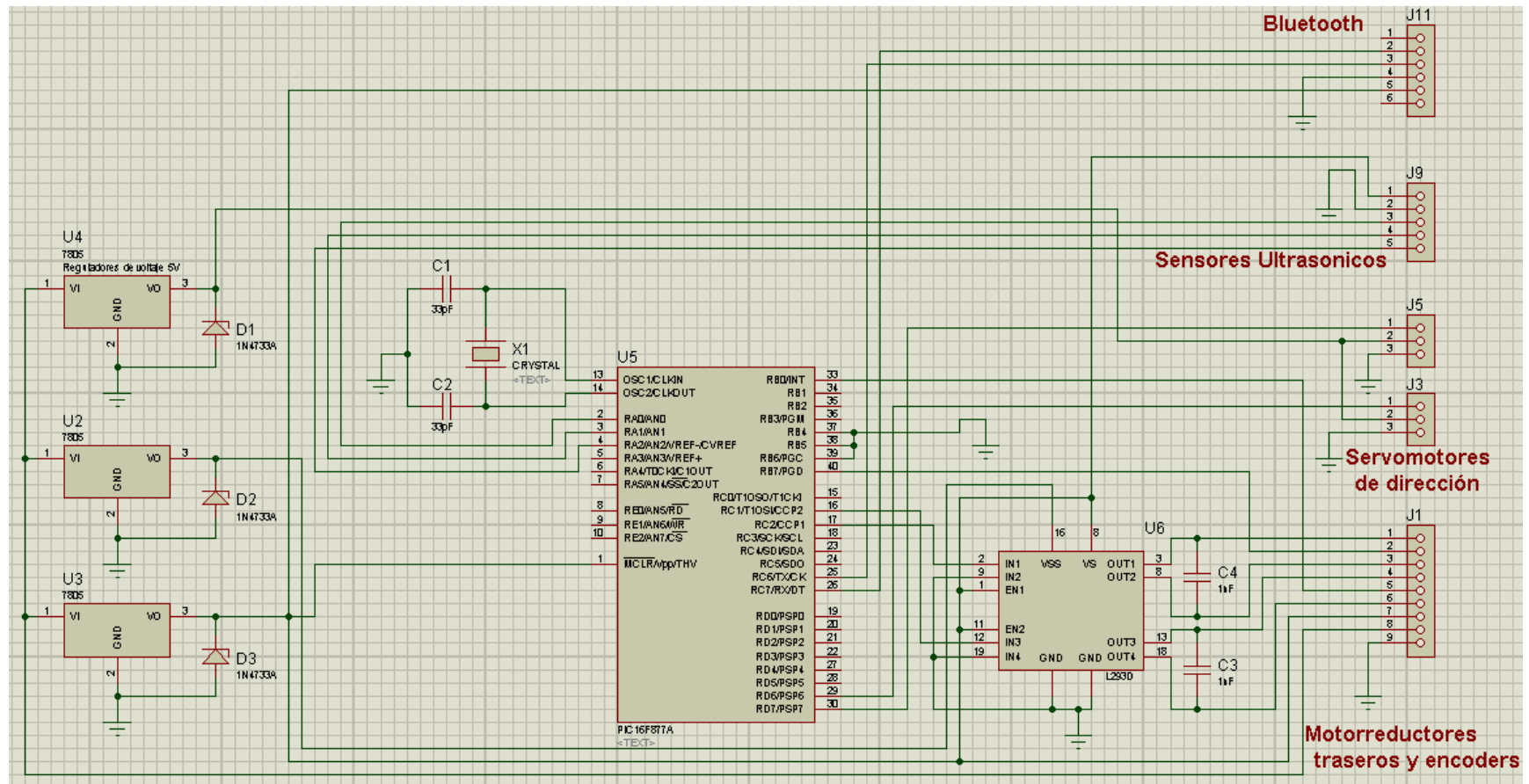


Figura 3.16. Esquema electrónico del robot móvil.

El robot utiliza un módulo bluetooth "BT0417C ATcommand", figura 3.17. Este módulo tiene la antena incorporada, con un rango de transmisión de hasta 10 metros.

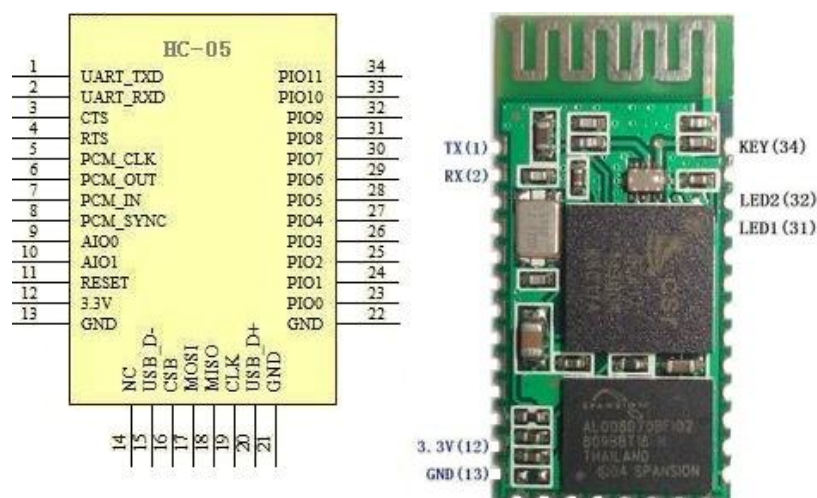


Figura 3.17. Interfaz bluetooth al serial.

Con respecto al método de conexión del módulo bluetooth, este utiliza el protocolo SPP (Serial Port Profile), para enlazarse con otro módulo Bluetooth que también utilice este protocolo, el cual establece una conexión serie virtual RS232 inalámbrica entre el módulo y el "smartphone", en tanto, para el enlace entre el módulo bluetooth y el microcontrolador, se emplea una interfaz RS232 a través de los pines TX(1) y RX(2) del módulo con una velocidad de 9600 baudios, 8 bits de datos y sin paridad.

El microcontrolador seleccionado es el PIC16F877A, esto debido a que contiene todas las características en cuanto a memoria e interrupciones necesarias, el módulo bluetooth, anteriormente descrito, se utiliza como interfaz entre el "smartphone" del robot y su electrónica. El PIC16F877A recibe cuatro datos del "smartphone"; el ángulo de la llanta derecha, el ángulo de la llanta izquierda, la velocidad angular de la llanta derecha y la velocidad angular de la llanta izquierda. La lógica empleada dentro del microcontrolador se encarga de orientar los servomotores delanteros, de acuerdo a los ángulos indicados, y de comandar a los motores traseros, con ayuda de los encoders colocados en las llantas traseras, el microcontrolador puede supervisar que sus velocidades angulares sean las correctas. En la figura 3.18, se puede apreciar un esquema de este microcontrolador.

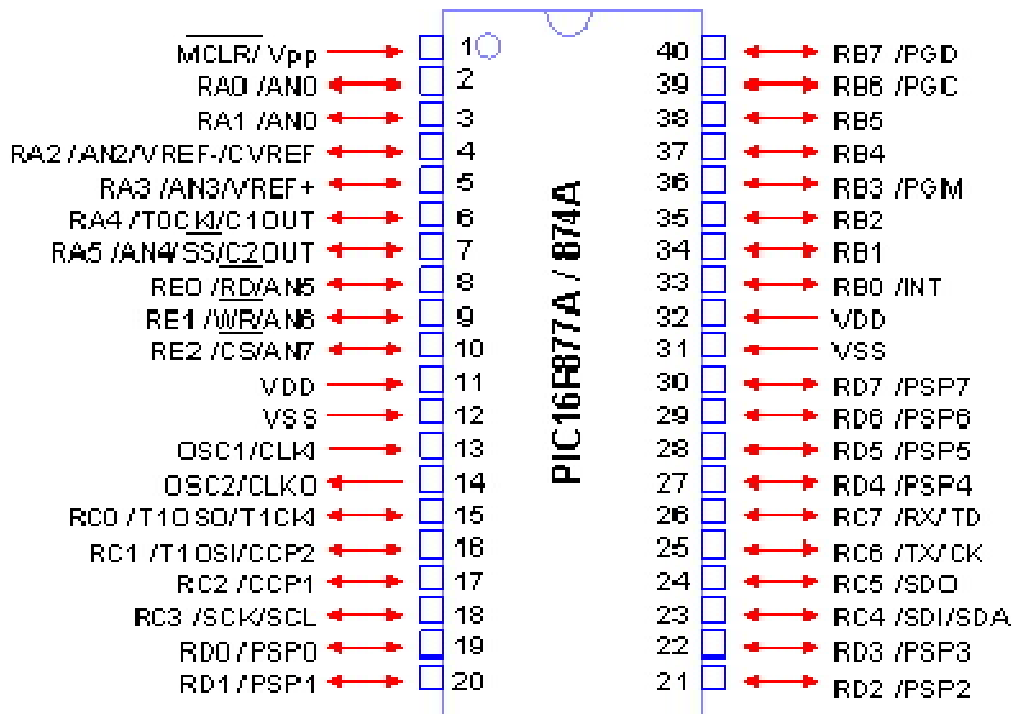


Figura 3.18. Microcontrolador PIC16F877A.

Con la finalidad de entender el funcionamiento del algoritmo programado en el microcontrolador, se presenta en las figuras 3.19 y 3.20, un diagrama de flujo del método principal, al inicio se configuran los puertos a utilizar, posteriormente se inicializan los valores que tomarán las señales que controlarán a los servomotores delanteros y a los motores traseros, a continuación se configuran las interrupciones que se emplearán y por último se tiene un ciclo infinito, en donde se toman las distancias hacia los obstáculos, por medio de la lectura de los tres sensores ultrasónicos con la finalidad de evitar colisiones.

Las dos primeras interrupciones, empleadas, son utilizadas para poder medir la velocidad angular que tienen las llantas fijas traseras y poder realizar correcciones si fuese necesario, en las figuras 3.21 y 3.22, se puede ver el diagrama de flujo que hace representación de la lógica empleada para la medición de la velocidad angular de la llanta izquierda y derecha, respectivamente, en ellas simplemente se realiza un conteo de los pulsos que arroja un encóder, colocado en cada llanta.

En las figuras 3.23 y 3.24, se expone el algoritmo empleado en la comunicación serial, como se ha mencionado anteriormente, el "smartphone" manda, como resultado del algoritmo de navegación empleado en él, cuatro datos; la velocidad angular de las llantas traseras y los ángulos de dirección de las llantas delanteras, en la figura 3.24, se puede observar cómo se toman estos caracteres y se transforman a enteros para poder realizar las operaciones necesarias para comandar los motores del robot.

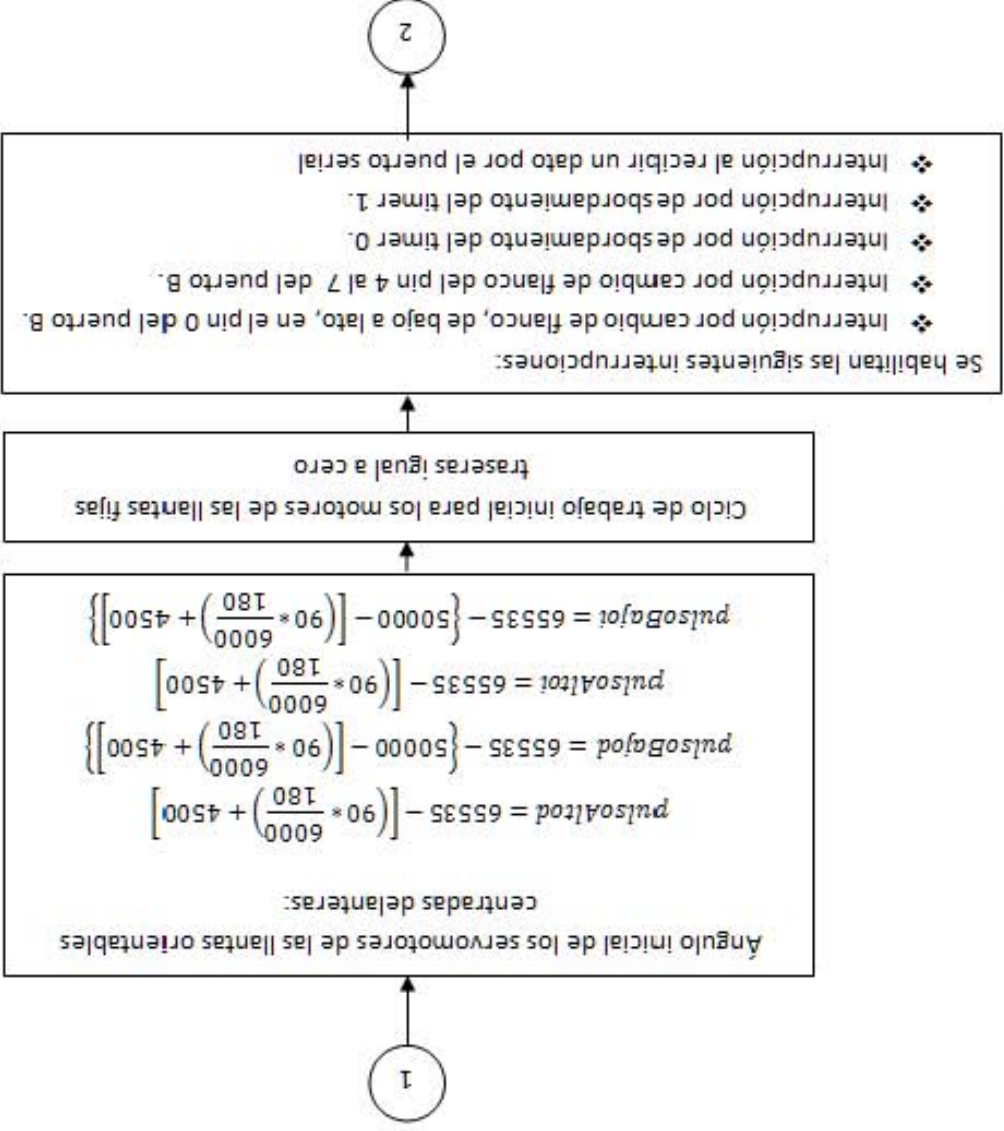
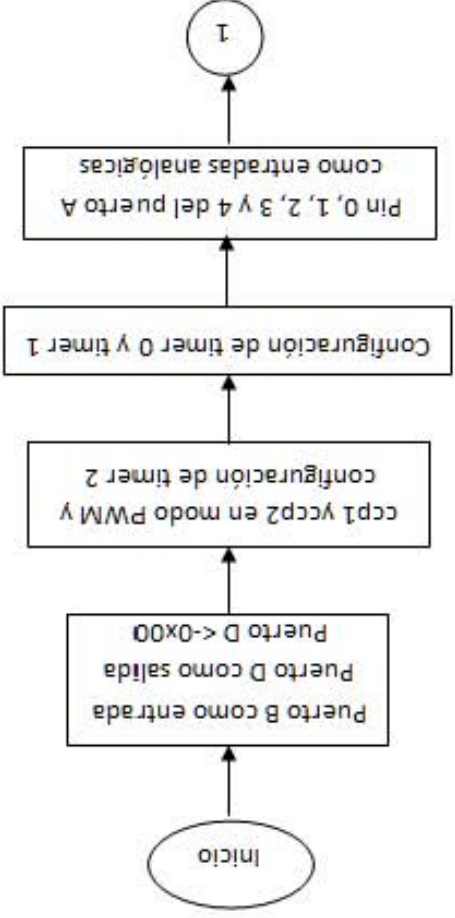


Figura 3.19. Algoritmo de la electrónica, método principal.

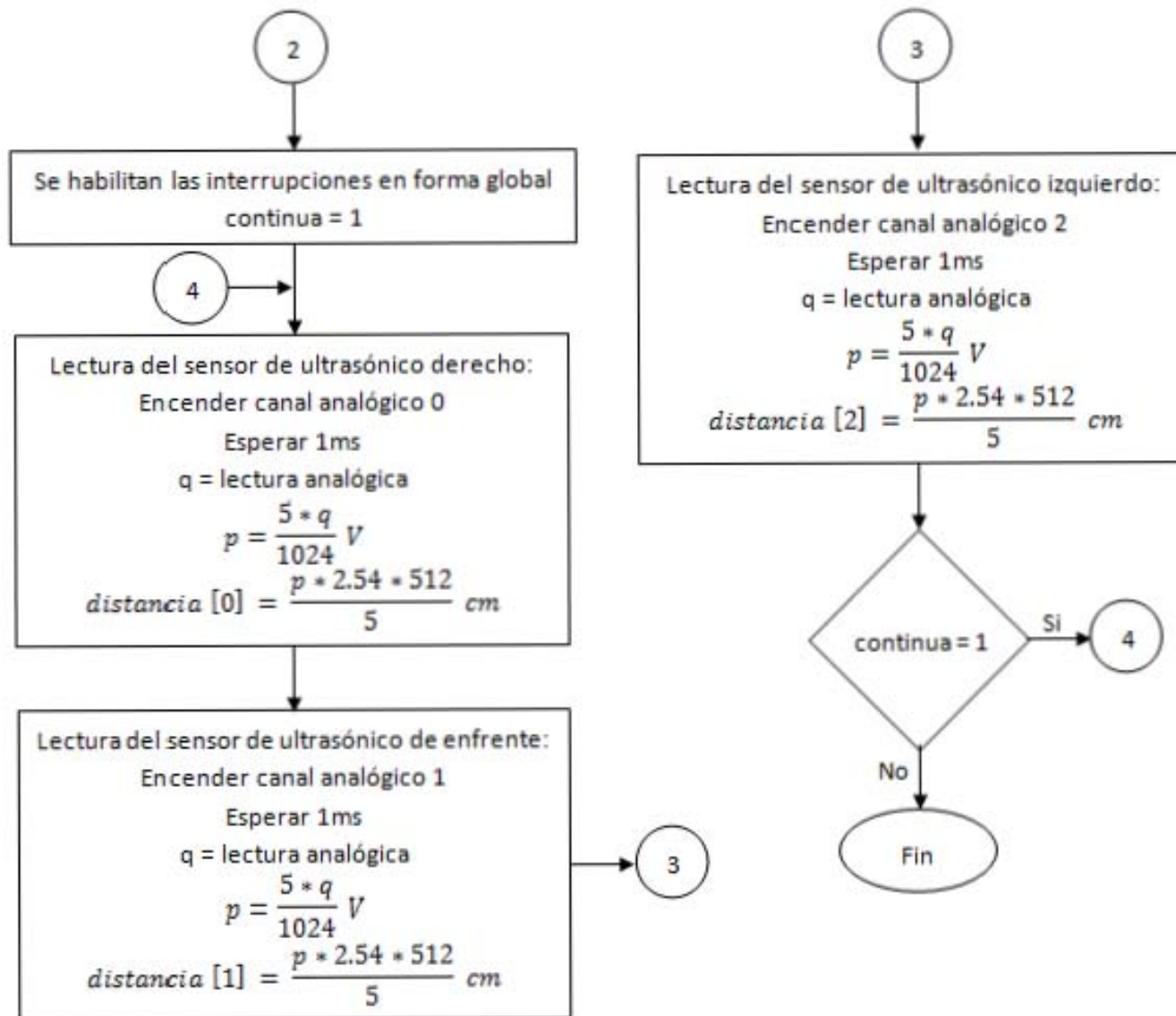


Figura 3.20. Algoritmo de la electrónica, continuación del método principal.

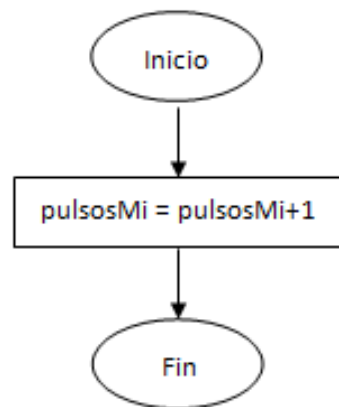


Figura 3.21. Algoritmo empleado en la interrupción del pin 0 del puerto B.

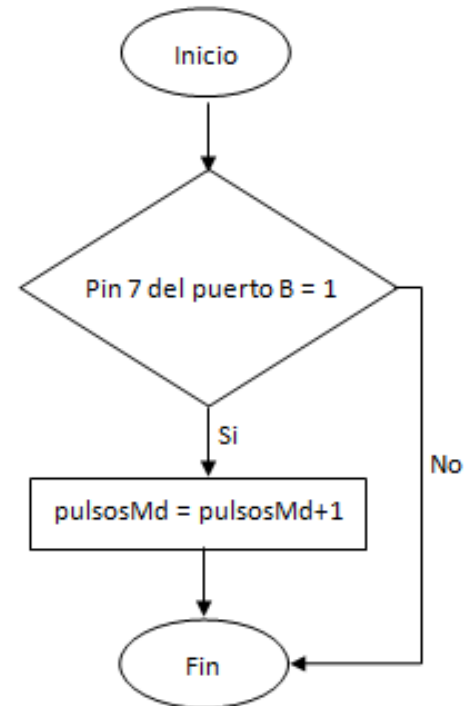


Figura 3.22. Algoritmo empleado en la interrupción del pin 4 al 7 del puerto B.

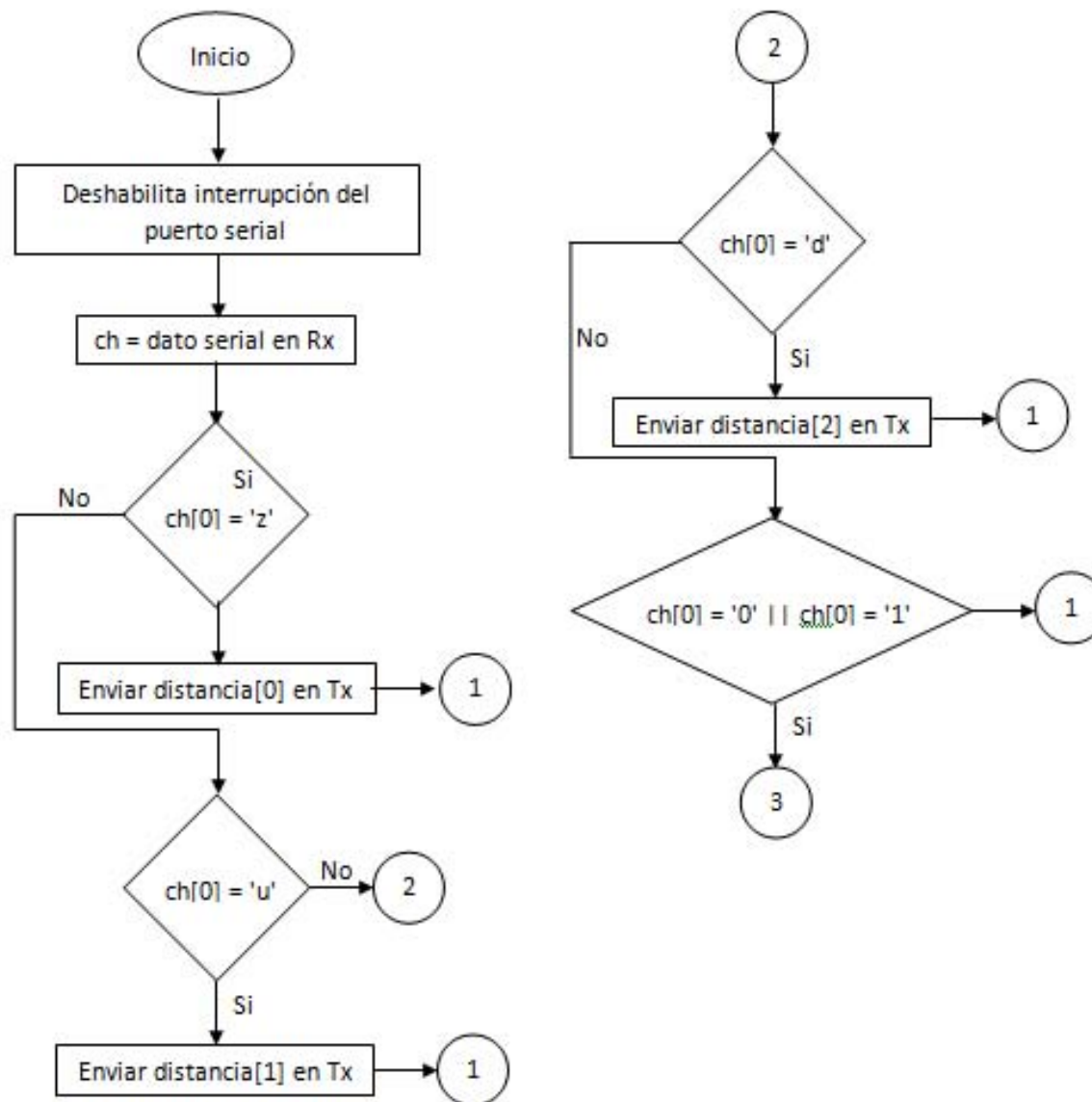


Figura 3.23. Algoritmo empleado para la comunicación vía serial RS232.

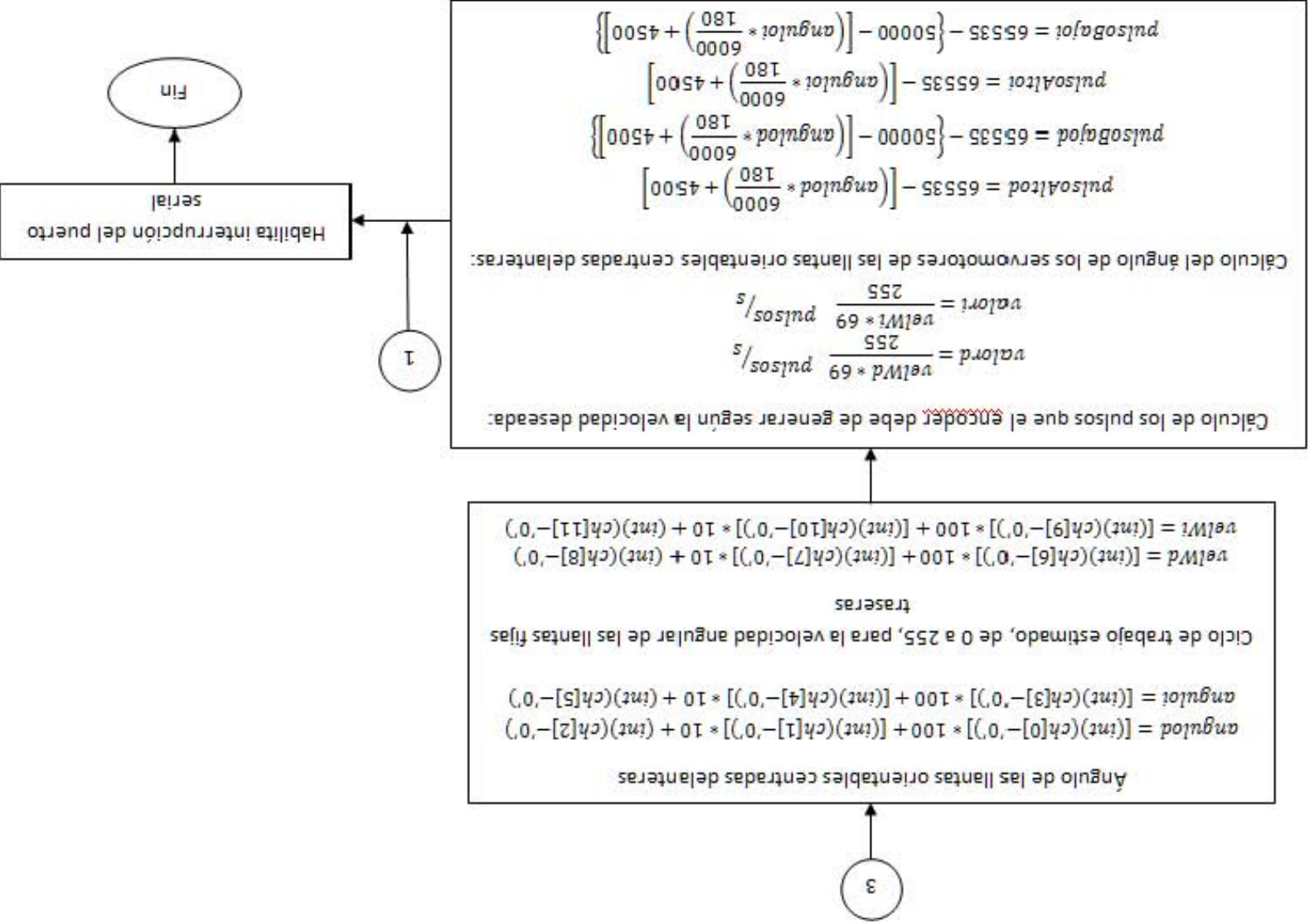


Figura 3.24. Algoritmo empleado para la comunicación via serial RS232, continuación.

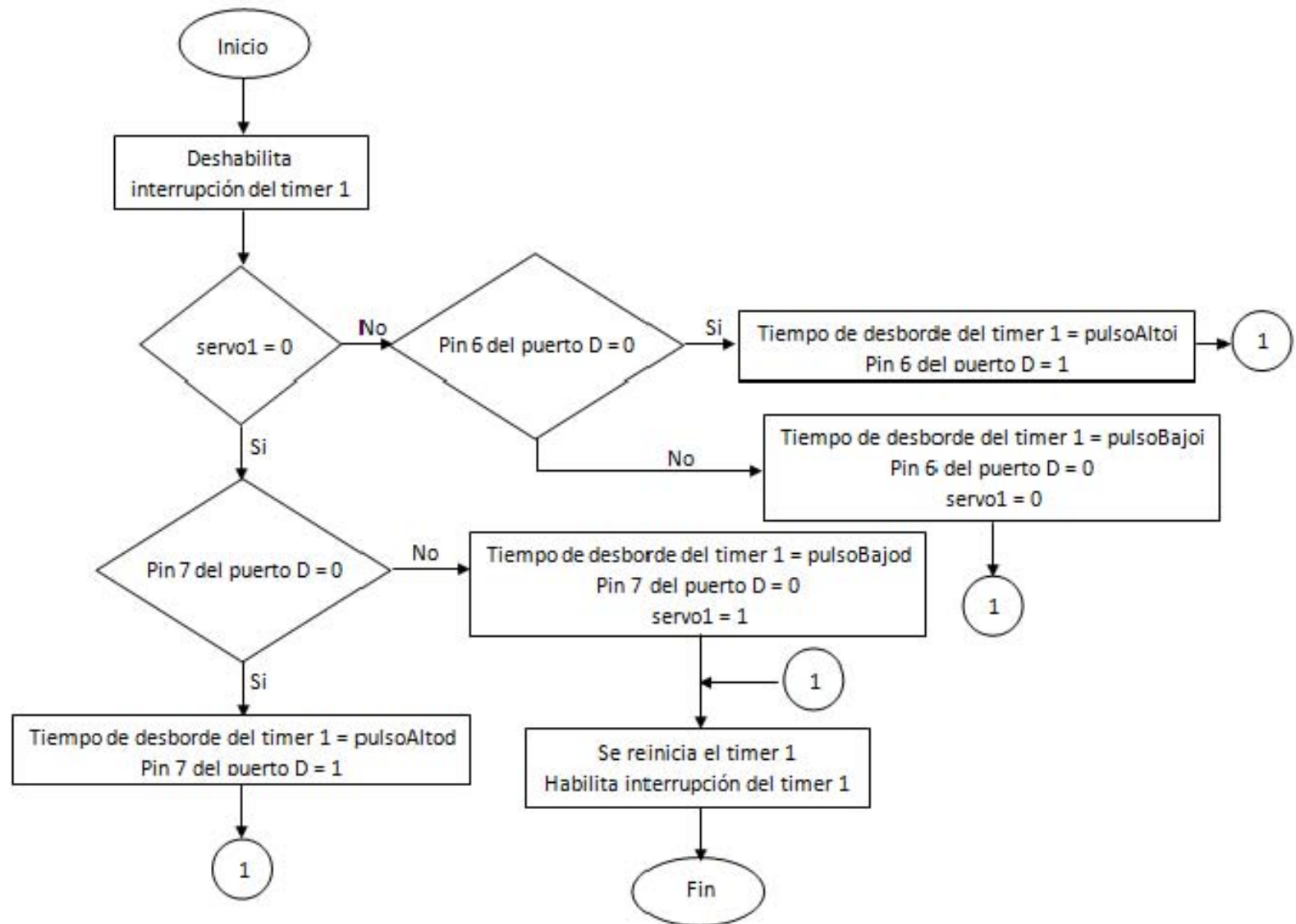


Figura 3.25. Algoritmo empleado para el control de los servomotores del robot.

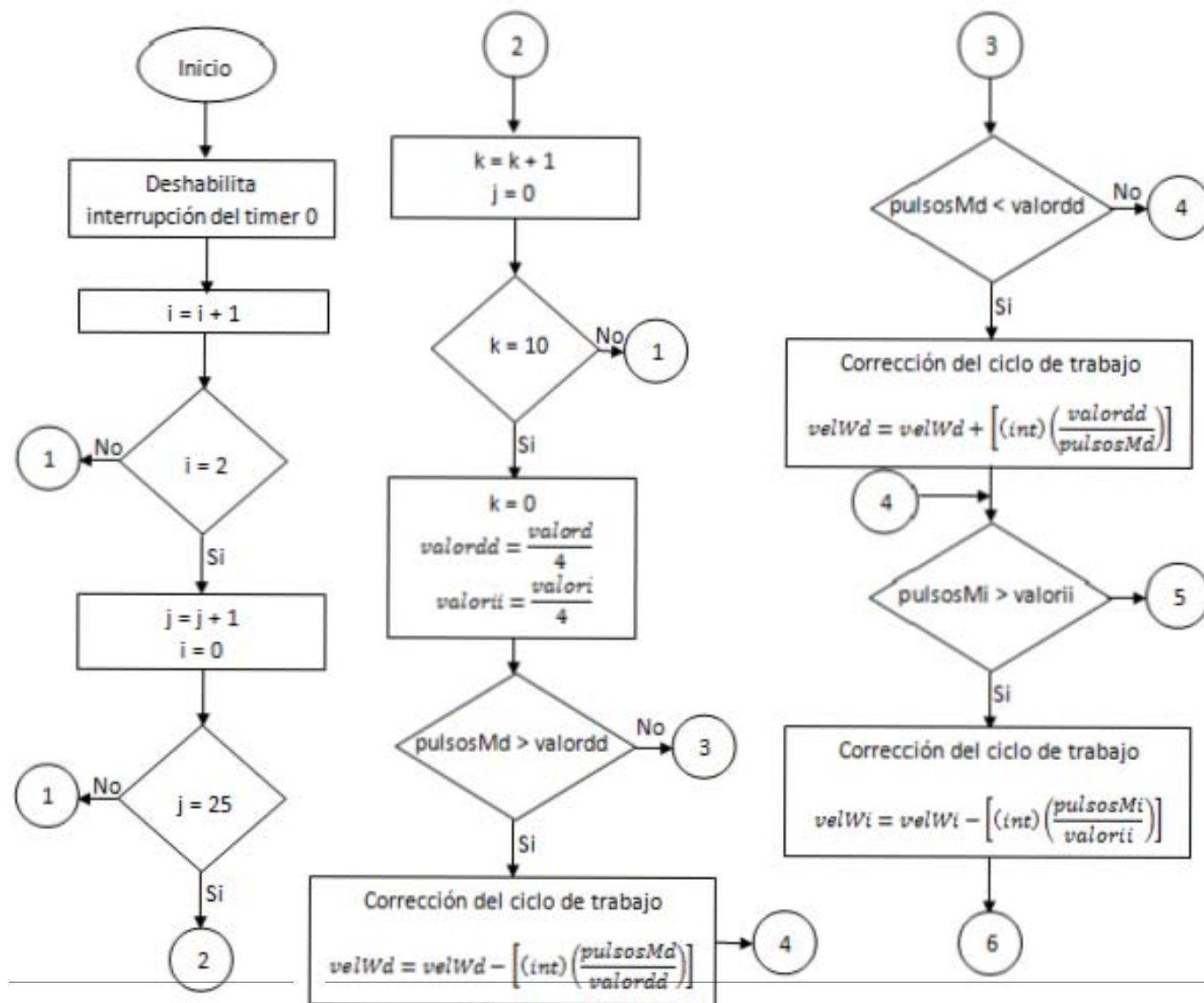


Figura 3.26. Algoritmo empleado para el control de la velocidad de los motores de corriente continúa.

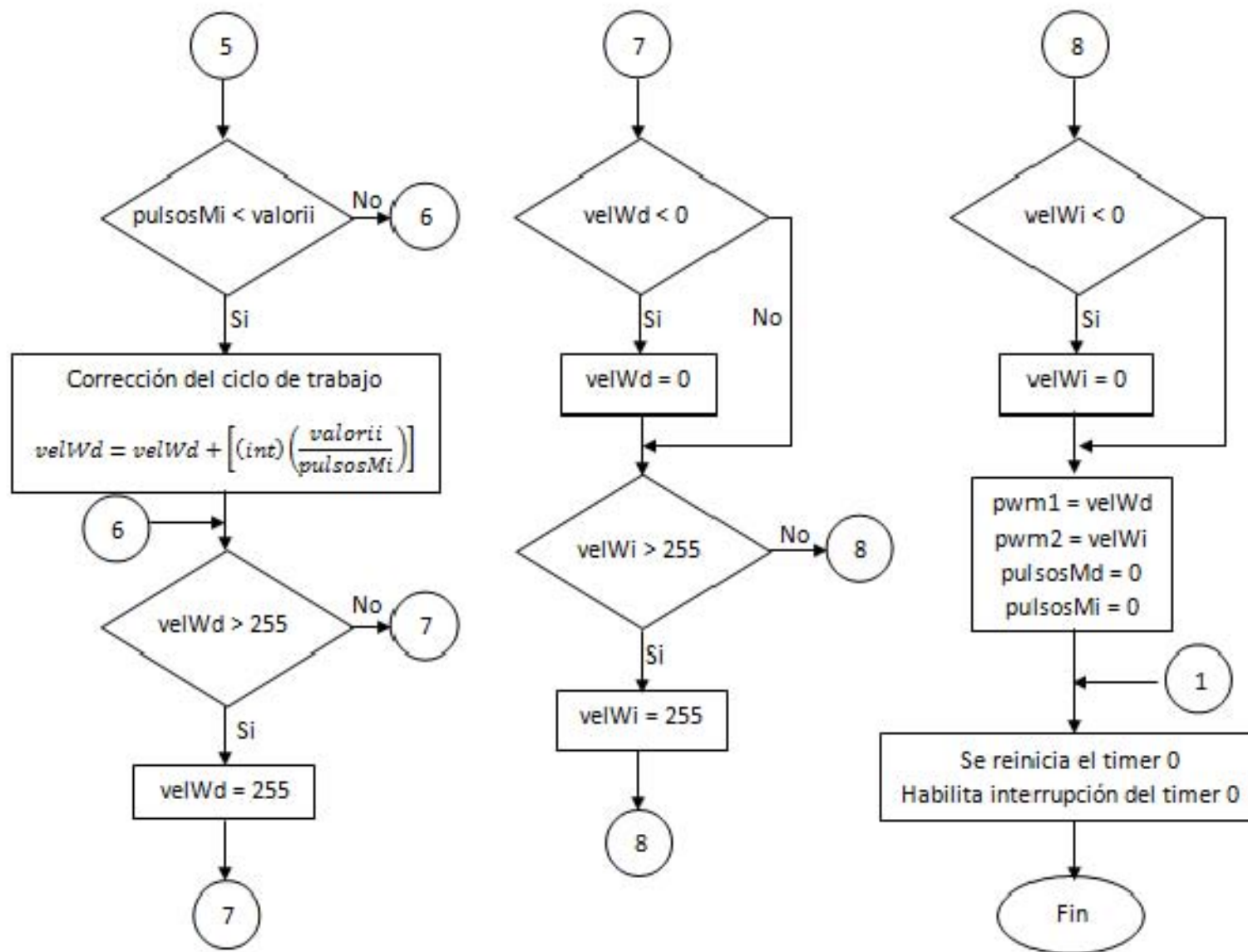


Figura 3.27. Algoritmo empleado para el control de la velocidad de los motores de corriente continua, (continuación).

La interrupción del "*timer 1*" del microcontrolador se utiliza para crear las señales de control para los servomotores que comandan a las llantas orientables centradas del robot, en la figura 3.25, se observa la lógica empleada, básicamente se crean dos señales "PWM" que tienen una frecuencia de 50Hz y un ancho de pulso que está en función del ángulo al que se desea que se posicione cada servomotor.

Para el control de las velocidades angulares de los motores de corriente continua que actúan a las llantas fijas traseras, se emplea la interrupción del "*timer 0*" del microcontrolador, figuras 3.26 y 3.27, esta interrupción se produce cada medio milisegundo y en ella se verifica la velocidad angular de cada llanta fija cada 250 ms, con ayuda de los pulsos almacenados en las interrupciones de las figuras 3.21 y 3.22, se compara la cantidad de pulsos con la cantidad de pulsos que se estiman, de acuerdo a la velocidad que se desea que tenga cada llanta, el encóder y la llanta fija utilizada en el robot se muestra en la figura 3.28, si la cantidad de pulsos cada 250 ms es mayor a la calculada entonces se reduce la velocidad del motor, en caso contrario se aumenta la velocidad del motor.



Figura 3.28. Encóder y llanta fija utilizada en el robot móvil.

4. PRUEBAS Y RESULTADOS

Este capítulo describe los resultados que se obtuvieron del sistema en general. La evaluación del prototipo del vehículo autónomo consistió en el seguimiento de una serie de trayectorias, obtenidas de "Google Maps".

4.1. Sensor GPS

El prototipo de auto autónomo necesita ubicarse, como se describe en el apartado número 3, para la realización del presente trabajo se cuenta con dos teléfonos inteligentes; un Motorola MB860 y un Motorola XT910, como se explicó en la sección 2.1, la precisión del GPS depende de varios factores, uno de ellos es como el software incorporado en el dispositivo procesa la señal que recibe de los satélites GPS, es por ello que se realiza un experimento, similar al encontrado en [4], el enfoque es comparar la exactitud de estos dos dispositivos GPS durante unas horas cuando se coloca en un lugar fijo (conocido).

En este caso, se han colocado estos dos teléfonos con sistema Android en una ubicación conocida en un edificio y se ha creado una aplicación que guarda los datos de la latitud y la longitud que el dispositivo detecta cada cinco minutos. Se empiezan a recoger los datos a las 10 am y se termina a las 8:00 pm. El conjunto de datos se considera suficiente para hacer una comparación razonable, ya que también se tomaría en cuenta la translación de la constelación de satélites GPS durante la mayor cantidad del día.

La exactitud de la posición detectada es reportada tomando en cuenta un radio de tolerancia. Para el Motorola MB860 el 35% de los datos recibidos estuvieron dentro de un radio de 3 metros de exactitud y un 65% cayeron dentro de 10 metros, esto se puede ver en la figura 4.1. El Motorola XT910 reportó un 57% de sus datos dentro de 3 metros y un 19% en un radio de 5 metros, la figura 4.2, muestra lo anterior.

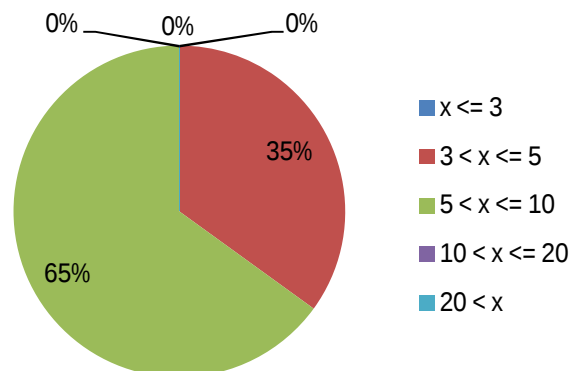


Figura 4.1. Conjunto de datos reportados por el Motorola MB860.

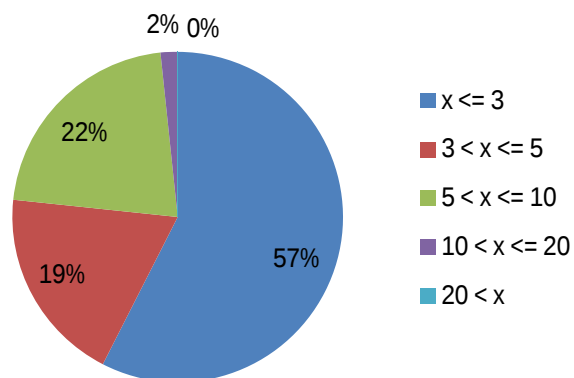


Figura 4.2. Conjunto de datos reportados por el Motorola XT910.

Para una mejor comprensión del desempeño en esta prueba del GPS incorporado en cada equipo móvil, se presentan en las figuras 4.3 y 4.4, una gráfica que muestra el error de la latitud en Azul y la longitud en rojo para el equipo Motorola MB860 y XT910 respectivamente, el eje "y" es el error en metros y el eje "x" es el número de la muestra.

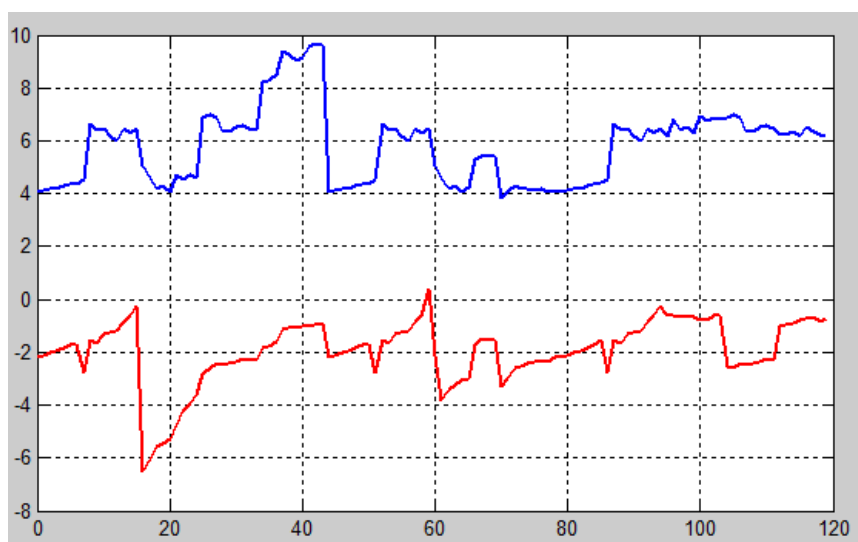


Figura 4.3. Error de la latitud (Azul) y longitud (rojo) en metros para el Motorola MB860.

Para ambos teléfonos la precisión fue variante, esto puede ser debido a la posición de los satélites. De manera más notable, es posible observar que la precisión obtenida por estos equipos es mucho mejor que la reportada en la documentación oficial, [23], es decir, la diferencia real de error es mucho menos de lo esperado, este valor generalmente va desde poco más de 5 metros a menos de un metro en algunos casos.

Aunque ambos equipos móviles tuvieron un gran desempeño, el "smartphone" que tuvo un menor error fue el Motorola XT910, al presentar un mayor porcentaje de muestras en un radio menor de aciertos y, tomando en cuenta el error, el MB860 tuvo un error promedio de 5.7828 y 1.9464 metros, para la latitud y longitud respectivamente, en tanto

el XT910 reportó un error promedio de 2.7551 y 0.5048 metros, en su latitud y longitud respectivamente, razones que favorecen al XT910.

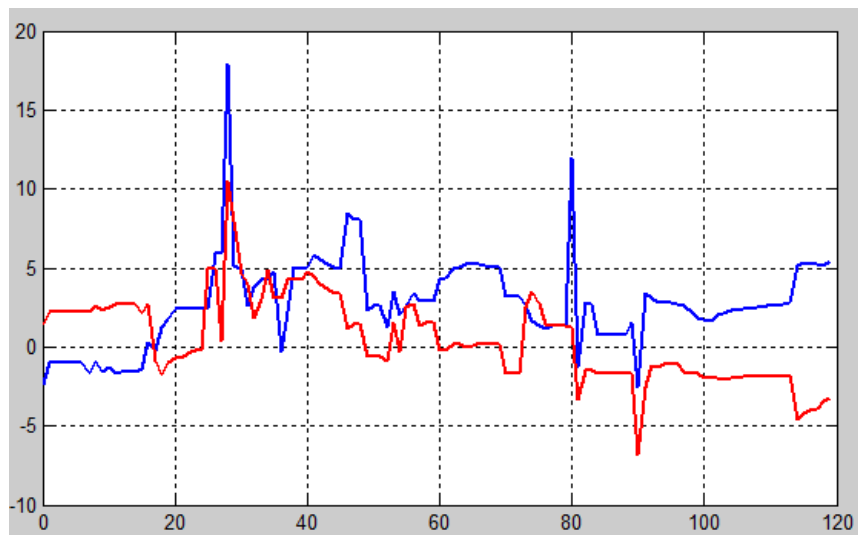


Figura 4.4. Error de la latitud (azul) y longitud (rojo) en metros para el Motorola XT910.

4.2. Sensor de Orientación

El robot móvil autónomo necesita orientarse, específicamente para conocer su dirección y saber si debe de girar para alcanzar su meta, en el apartado 3 se da a conocer cómo trabaja este sensor, entre otros. Para cumplir los objetivos del presente trabajo, este sensor es importante, al igual que el GPS, ya que el algoritmo de navegación utiliza su información para funcionar.

A diferencia del sensor GPS, el sistema Android es capaz de proveer información del tipo de sensor que se tiene en el dispositivo, de manera programática. Para ambos casos, es decir, tanto para el Motorola MB860 y el XT910 se tiene un sensor "AK8975" de la marca "Asahi Kasei", particularmente el sistema Android provee información de este sensor en grados, de esta manera se tiene un rango de 0° a 360° con una resolución de 0.015625°, por tal motivo, no se espera tener mucha diferencia entre la precisión de un dispositivo y el otro.

Para conocer la precisión del sensor de orientación, que estos dispositivos manejan, particularmente el de "azimut", se realiza una prueba comparando los datos que se obtienen de los "smartphone" con una brújula real.

Se crea una aplicación muy parecida a la mostrada en las figuras 3.11 y 3.12, de la sección 3.2.1.1 en donde se muestran datos de varios sensores, uno de ellos es el ángulo de "azimut", la prueba consiste en comparar los datos que se obtienen de la brújula

analógica y de los "smartphone", la figura 4.5, muestra algunos datos obtenidos para el modelo MB860 y la figura 4.6, muestra los datos para el modelo XT910.



Figura 4.5. Comparación entre una brújula analógica y el sensor de orientación del modelo Motorola MB860.

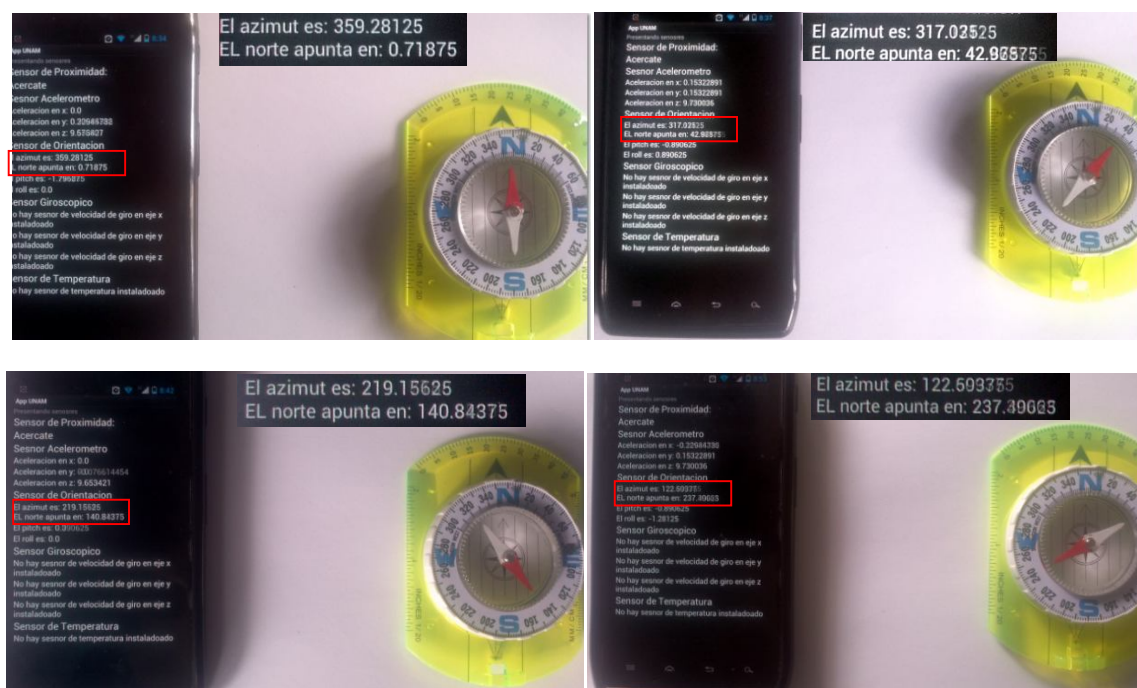


Figura 4.6. Comparación entre una brújula analógica y el sensor de orientación del modelo Motorola XT910.

En la tabla 4.1 se pueden apreciar los datos obtenidos de las observaciones del experimento anteriormente descrito. Se observa que para ambos teléfonos el sensor de orientación funciona muy bien y tiene bastante exactitud, se tiene un error promedio de 0.58 y 0.43 grados para el modelo Motorola MB810 y el XT910, respectivamente, aunque la precisión de los dispositivos móviles, para este sensor, disminuye mucho al estar cerca de objetos metálicos o imanes, sin embargo empresas privadas trabajan en este problema [24], con lo cual esta dificultad puede ser resuelta.

Ángulo al que apunta la aguja norte			Error	
Motorola		Brújula Real		
MB810	XT910		MB810	XT910
358.35	0.71	0	-1.65	0.71
19.25	24.56	20	-0.75	4.56
37.7	42.9	40	-2.3	2.9
58.84	63.53	60	-1.16	3.53
80.34	82.45	80	0.34	2.45
101.87	102.93	100	1.87	2.93
122.45	120.95	120	2.45	0.95
144.93	140.84	140	4.93	0.84
162.54	160.54	160	2.54	0.54
181.23	178.03	180	1.23	-1.97
205.5	199.21	200	5.5	-0.79
224.09	218.07	220	4.09	-1.93
240.73	237.49	240	0.73	-2.51
260.25	259.48	260	0.25	-0.52
280.0	275.12	280	0	-4.88
297.82	302.57	300	-2.18	2.57
318.67	318.62	320	-1.33	-1.38
338.12	340.46	340	-1.88	0.46
358.5	359.8	360	-1.5	-0.2
Error promedio:			0.5884211	0.43473684

Tabla 4.1. Error presentado entre la brújula analógica y digital de los modelos Motorola MB810 y XT910.

4.3. Navegación

El presente trabajo tiene como uno de los objetivos; utilizar la información obtenida de los sensores de navegación y GPS del dispositivo móvil para alimentar el algoritmo de navegación, con la finalidad de que el prototipo de automóvil autónomo siga una trayectoria obtenida por medio de la tecnología de "Google Maps", la arquitectura mencionada en el apartado 2 es fundamental para el presente trabajo, la cual se recapitula brevemente a continuación.



Figura 4.7. Diagrama del sistema a implementar.

En la figura 4.7, se puede apreciar el sistema de manera general, este está compuesto por cuatro elementos interconectados entre sí, se tienen dos dispositivos móviles, uno para el usuario y otro para el robot, comunicándose entre ellos por medio de mensajería SMS, la red de satélites GPS y el robot móvil, que sirve como prototipo de auto autónomo, cada uno de estos elementos se encuentra ampliamente explicado en las secciones anteriores. Por medio de una interfaz gráfica de usuario, programada en un "smartphone", se reporta la ubicación del robot en un mapa y se le indica hacia dónde dirigirse, señalando un punto en el mismo. Las intenciones del usuario se transmiten vía SMS hasta el robot y con ayuda del GPS incorporado en el "smartphone" del robot, es posible obtener su localización a través de su latitud y su longitud, posteriormente, este crea una trayectoria hasta su objetivo con ayuda de "Google Maps". Empleando un algoritmo de control basado en la información de un compás electrónico y GPS embebidos en el sistema móvil celular, se calcula la orientación de cada una de las llantas frontales y la velocidad angular de las llantas traseras. Esta información es almacenada en un arreglo y enviada a través de una interfaz bluetooth hacia el robot móvil. La información es recibida y procesada por el microcontrolador del robot, quien finalmente comanda a cada uno de los actuadores con la finalidad de alcanzar la posición y la velocidad indicada, logrando que el robot se desplace alcanzando cada una de las metas que se obtuvieron de "Google Maps". Al ir alcanzando cada uno de estos objetivos, de manera sistemática, se logra recorrer la trayectoria obtenida y alcanzar el destino seleccionado por el usuario desde el otro "smartphone".

En base a las pruebas realizadas en las secciones 4.1 y 4.2, el "*smartphone*" que portará el robot móvil será el Motorola XT910, el usuario usará el modelo MB860 debido a que sólo es usado como interfaz para comunicación entre el robot y el usuario principalmente.

La eficiencia de esta tecnología y del algoritmo empleado, es validada realizando un recorrido autónomo y midiendo el error que se tiene comparando la trayectoria realizada por el robot con la trayectoria deseada. Los experimentos realizados se desarrollaron en ambientes exteriores, debido a la finalidad del trabajo. El proceso que se sigue se puede apreciar en las figuras 3.3 y 3.9, resumiendo:

1. El usuario ingresa el número telefónico del "*smartphone*" del robot.
2. Se ingresa el número telefónico del usuario en el "*smartphone*" del robot.
3. El usuario recibe la ubicación del robot en un mapa.
4. El usuario señala el punto de destino del robot sobre el mapa y le manda la instrucción.
5. El "*smartphone*" del robot consulta una ruta a "Google Maps".
6. De acuerdo a la orientación y ubicación del robot, así como la ubicación de las submetas, obtenidas de "Google Maps", que van formando la ruta a seguir, el algoritmo de navegación va comandando las cuatro ruedas del robot.

4.3.1. Trayectoria definida por dos puntos

La primera trayectoria evaluada es una compuesta por dos puntos, es decir, una línea, el primer punto es el de partida y el segundo es la meta definida por el usuario. Esta trayectoria es fundamental y la base para la navegación en este proyecto.

El robot móvil se colocó al azar y se realizó un recorrido con la trayectoria formada por "Google Maps", para evaluar el desempeño en general del robot móvil. En la figura 4.8, se observa la trayectoria de referencia (azul) y la descrita por el robot móvil (rojo) en el plano de trabajo y la gráfica de la figura 4.9, muestra el recorrido realizado por el robot (rojo) comparándola con el recorrido deseado (azul) en coordenadas de latitud y longitud, en ellas es posible observar que tanto la trayectoria obtenida por "Google Maps" y la del robot inician en aproximadamente en el mismo punto. En las figuras 4.8 y 4.9, es posible observar como el robot, representado en color rojo, trata de llegar a su meta, aun cuando este inició dando la espalda a la meta, justo después de orientarse hacia el objetivo, el robot se traslada casi en línea recta reduciendo poco a poco su error. En términos generales es posible observar un seguimiento de trayectoria aceptable, considerando que se cuenta con poca información de la trayectoria a seguir, es decir, sólo se cuenta con el punto de inicio y final de la trayectoria. También hay que considerar el tiempo de

procesamiento de los cálculos y el tiempo en el que el GPS se actualiza, este tiempo es variable, pero en mayor de los casos se obtienen dos muestras cada segundo.



Figura 4.8. Trayectoria de referencia en azul y trayectoria realizada por el robot móvil en rojo.

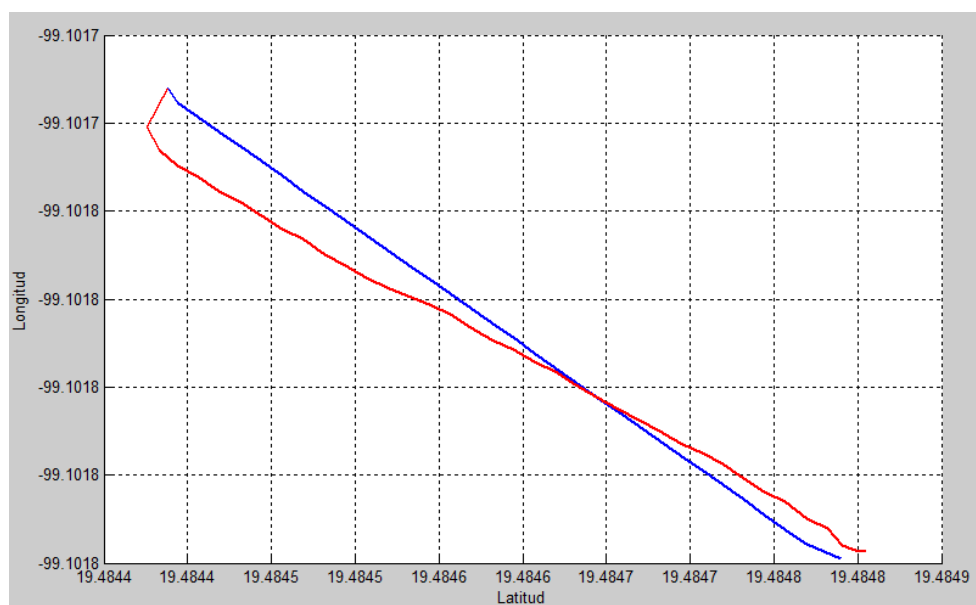


Figura 4.9. Comparación entre la trayectoria deseada a seguir (azul), con la realizada por el robot móvil (rojo).

Por último se presenta en la figura 4.10, una gráfica que muestra el error que se obtuvo a través del recorrido del robot, es posible notar que el error se mantuvo aproximadamente dentro de un radio de 1.5 metros, el cual está dentro del rango encontrado en la sección 4.1 del presente trabajo. Este error de 1.5 metros representa el 3.3% de la longitud total de 45 metros aproximadamente de la trayectoria a seguir, por lo cual se considera un buen resultado.

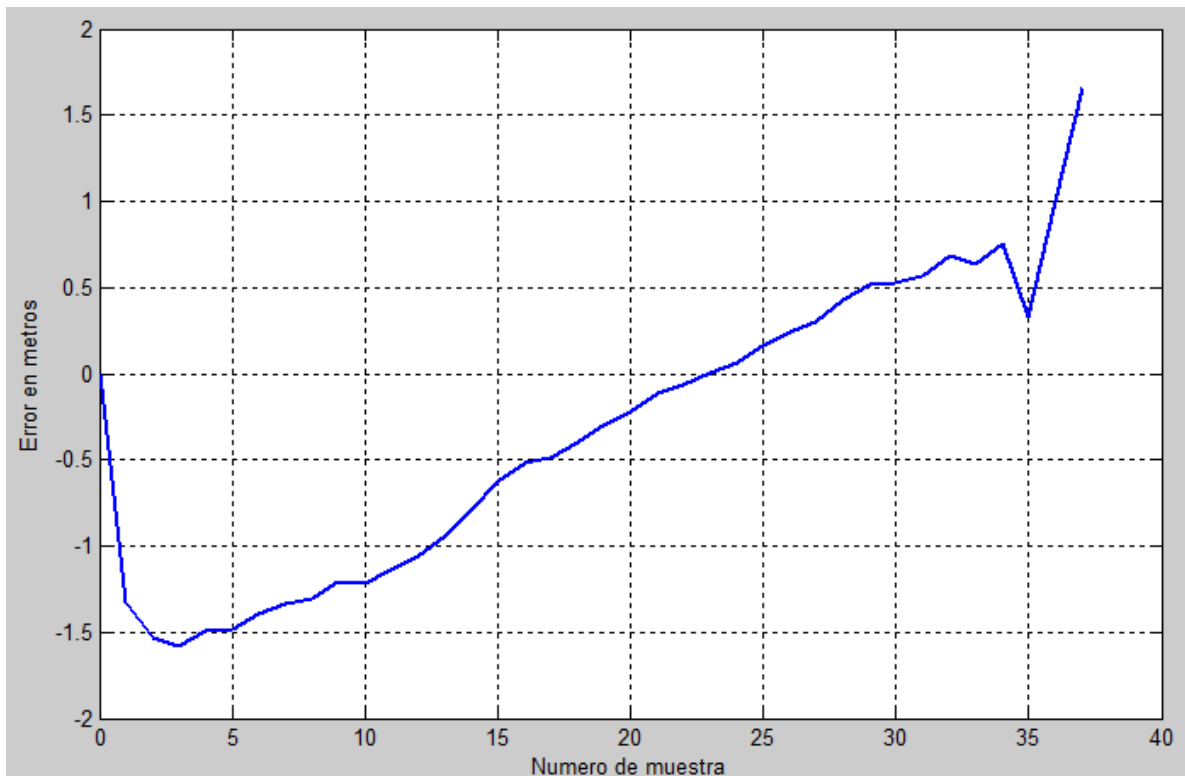


Figura 4.10. Error de seguimiento de la ruta de referencia del robot.

4.3.2. Trayectoria definida por más de dos puntos

Enseguida se realizó una prueba consistente en el seguimiento de una trayectoria definida por más de dos puntos. Dicha prueba sirve para comprobar el funcionamiento lógico del algoritmo empleado, hay que recordar que la trayectoria a seguir, que se obtiene de "Google Maps", está definida por una serie de puntos geodésicos o geográficos, es decir, están dados en coordenadas de latitud y longitud, al ir uniéndolos, con líneas rectas por ejemplo, es posible obtener una trayectoria, entonces, al ir alcanzando cada uno de estos puntos de forma sistemática, se estaría recorriendo la trayectoria deseada, por esta razón, la prueba realizada en la sección 4.3.1 es fundamental, ya que, básicamente, si se tiene una trayectoria definida por " n " puntos tendríamos una trayectoria formada por " $n-1$ " rectas unidas, de esta manera, el robot móvil debe ir siguiendo las rectas de manera sistemática una tras de otra.

En esta prueba, el robot móvil se colocó orientado hacia su primer objetivo, con la finalidad de analizar su comportamiento ante una situación diferente, posteriormente se realizó un recorrido con la trayectoria formada por "Google Maps", para evaluar el desempeño en general del robot móvil, dicha trayectoria se forma conforme se ha explicado en las secciones anteriores.

En las figuras 4.11 y 4.12, se observan las trayectorias de referencia (azul) y las descritas por el robot móvil (rojo) en el plano de trabajo y en coordenadas de latitud y longitud, respectivamente; en este caso la trayectoria está formada por 4 puntos, el primero corresponde a la ubicación del robot, el segundo y el tercero definen un cambio de dirección y el último es la meta, seleccionada por el usuario.

De nuevo es posible observar que tanto la trayectoria obtenida de "Google Maps" y la del robot inician aproximadamente en el mismo punto. En las figuras 4.11 y 4.12, es posible observar como la trayectoria descrita por el robot, representada en color rojo, trata de llegar a su primer objetivo, se puede notar como el robot sigue de manera fiel la primer recta que conforma a la trayectoria de referencia, haciendo el error en esta zona muy próximo a cero, este comportamiento seguramente es debido a que el robot no tuvo que realizar correcciones grandes en su orientación, sin embargo, al llegar al punto número dos el error crece alrededor de 1 metro, provocando un error al inicio del recorrido de la segunda línea, comprendida entre los puntos dos y tres de la trayectoria, en esta sección el robot se traslada casi en línea recta reduciendo poco a poco su error, al llegar al punto número tres existe un error de nuevo que se encuentra dentro de 3 metros, debido a los datos que se obtienen del sensor GPS, por último se recorre la recta número tres y se observa un comportamiento similar al realizado en la recta anterior, sin embargo, al final de su trayectoria se detecta un error mayor en la latitud que en la longitud, este error se encuentra alrededor de 2 metros.

La figura 4.13, muestra el error de seguimiento entre la referencia y la trayectoria real del robot, la gráfica en rojo representa el error en la latitud, la gráfica en color violeta representa el error observado en la longitud y por último, la gráfica en color negro representa el error más grande que se observó comparando los errores detectados entre la latitud y la longitud.



Figura 4.11. Trayectoria de referencia en azul y trayectoria realizada por el robot en rojo para más de dos puntos.

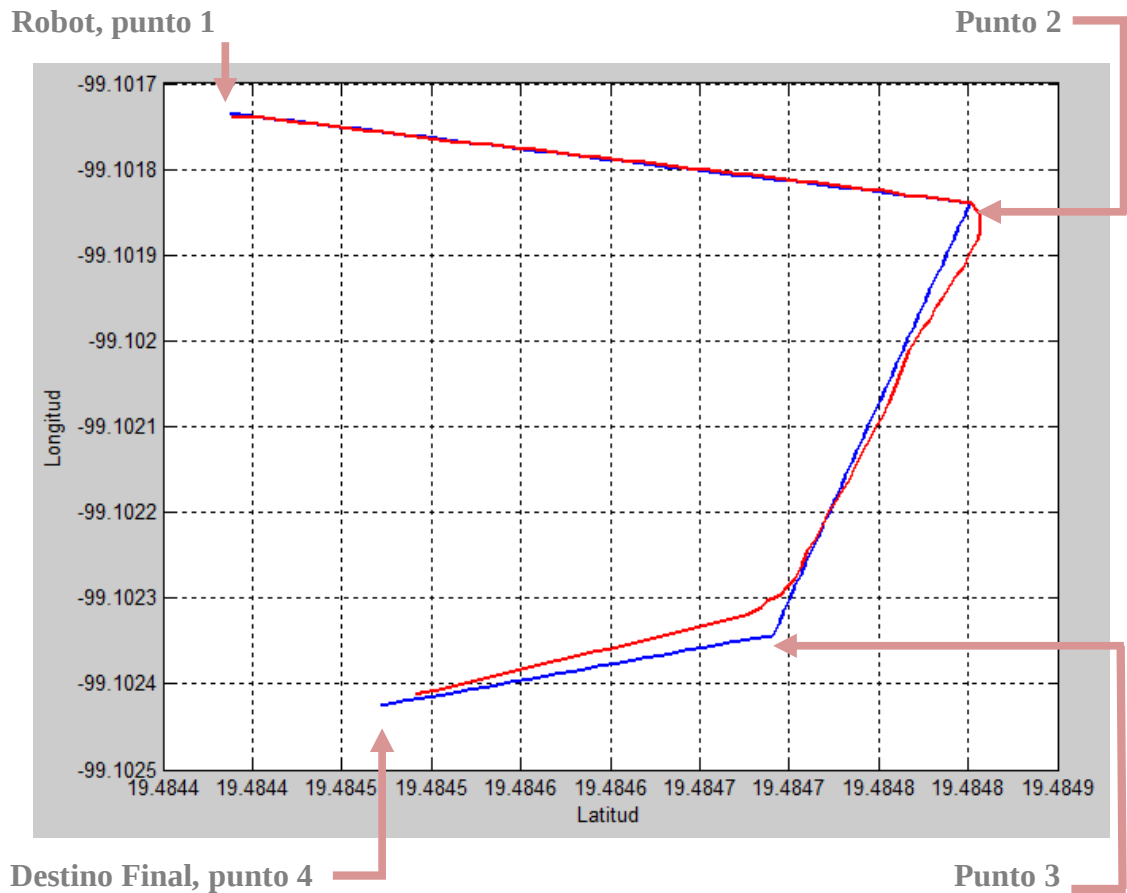


Figura 4.12. Comparación entre la trayectoria deseada a seguir (azul), con la realizada por el robot móvil (rojo) para más de dos puntos.

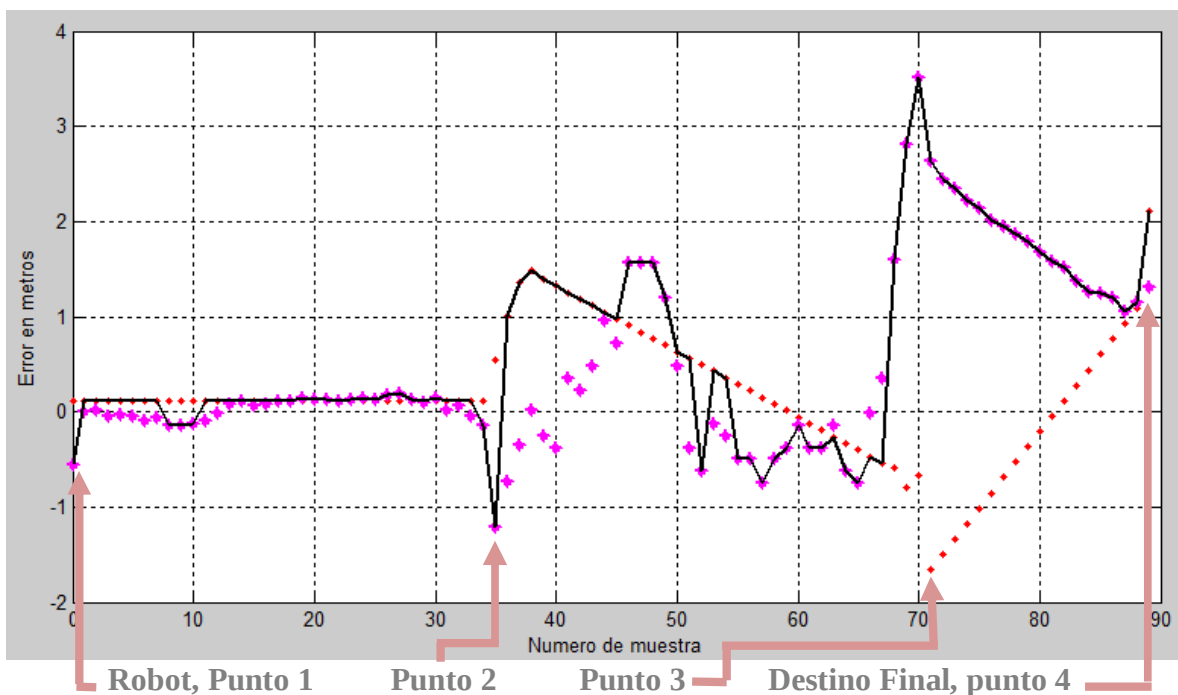


Figura 4.13. Error de seguimiento de la ruta de referencia del robot para más de dos puntos.

Una vez mostrados los datos obtenidos en la evaluación de las tecnologías involucradas y del prototipo en general, resulta evidente como pueden ser aprovechadas las tecnologías móviles, tanto los "*smartphone*", como medios de procesamiento y comunicación, y los mapas digitales, en este caso "Google Maps", que además es gratuito, para poder desarrollar tecnología de fácil acceso para la automatización de los automóviles.

Finalmente es posible afirmar, que los objetivos de este trabajo fueron cumplidos. Se logró un sistema de rastreo, basado en GPS y sensores de navegación que los sistemas móviles poseen, que permita obtener la posición y orientación del robot autónomo dentro de un área urbana, se utilizó la información obtenida del sistema de rastreo para alimentar el algoritmo de navegación conocido como "*campos potenciales*" [5], con la finalidad de seguir una trayectoria obtenida por medio de la tecnología de "Google Maps" y se comprobó la capacidad que los teléfonos inteligentes tienen actualmente para ser utilizados en sistemas robóticos, proveyendo de una unidad de procesamiento pequeña y potente, capaz de interactuar no solo con el usuario, sino con sistemas electrónicos, tecnologías de internet y con otros "*smartphone*".

5. CONCLUSIONES Y TRABAJO A FUTURO

El presente trabajo muestra el desarrollo de un sistema de rastreo para conocer la ubicación y la orientación de un vehículo autónomo, en este caso se empleó un robot móvil como prototipo, utilizando tecnologías de fácil acceso y en algunos casos gratuitas, proveyendo al sistema de una interfaz de fácil comprensión para el usuario, además de conectividad con otras tecnologías y con la red de internet para obtener información importante para su funcionamiento, reduciendo costos y facilitando su desarrollo sin reducir su desempeño, sino incrementándola, utilizando tecnologías novedosas y emergentes como son el GPS, "Google Maps" y las tecnologías contenidas dentro de los "smartphone".

Al finalizar las pruebas, resulta claro que se lograron cumplir los objetivos propuestos. El hecho de que el usuario sólo tenga una interfaz de usuario sencilla para operar al robot, señalando un punto como destino sobre un mapa, y de que el instrumento utilizado como interfaz entre el robot y el usuario fuera un dispositivo pequeño y, que además puede ser utilizado para diversos fines, es decir, un teléfono inteligente, tiene una gran importancia ya que los usuarios no necesitan tener conocimientos diferentes a los de una persona normal para poder operar la tecnología desarrollada. Esto le da un valor de carácter social ya que al acercar la tecnología a la sociedad de una manera sencilla, ésta tiene una mayor probabilidad de ser aceptada y sufrir un mayor desarrollo en menos tiempo.

Normalmente los robots móviles se encuentran dentro de un entorno académico, utilizando herramientas de laboratorio y ambientes controlados, la navegación efectuada resultó ser una combinación de tecnologías existentes, cada una aportando una característica importante ya sea portabilidad, instrumentación de fácil acceso o creación de rutas de manera adaptable, dependiendo de la ubicación del vehículo y del destino, permitiendo ubicar a un robot móvil en un plano urbano, orientarlo e indicarle una trayectoria a seguir, obteniendo errores de posicionamiento que representan menos del 10% de la longitud final de la trayectoria.

Una de las tecnologías utilizadas es el GPS, considerando que desde el año 2000 se tiene acceso a una precisión mucho mejor, parecida a la utilizada por los militares, puede ser considerada una tecnología novedosa, sin embargo, el depender de ésta a la vez es una desventaja ya que en cualquier ocasión, por motivos de seguridad, Estados Unidos puede considerar degradar la señal de GPS a la que tenemos acceso y, por tal motivo, provocar errores en la navegación bastante considerables, mientras esto no ocurra esta tecnología representa un servicio de localización confiable y que ayuda al desarrollo de nuevas tecnologías, como es el caso de vehículos autónomos.

Otra tecnología utilizada fue la red de telefonía celular para la transmisión de instrucciones entre el usuario y el robot móvil, su uso permitió comandar al robot a largas distancias con un retardo de 2 a 10 segundos aproximadamente, los dispositivos utilizados para el usuario y el robot, es decir, los "smartphone" poseen la tecnología propia de un celular y permiten tener acceso a la tecnología conocida como SMS, "Short Message Service", de forma programática, es decir, es posible acceder a las cadenas de caracteres contenidas en los mensajes SMS de forma automática, de esta manera se logró establecer un medio de comunicación de forma sencilla y confiable, pues el mismo sistema operativo del celular se encarga de administrar toda esta tecnología y sólo es necesario acceder a ella con las herramientas que se ofrecen, con la única restricción que exista señal GSM para los dos dispositivos.

A la vez que se llevaron a cabo las pruebas, se evidenció que el trabajo realizado podría incorporar mejoras a futuro, permitiendo así tener un mejor funcionamiento, ellas son:

Evasión de obstáculos: El prototipo empleado contiene sensores ultrasónicos, sin embargo no son capaces de detectar obstáculos de poco volumen, pero que pueden representar razón suficiente como para evadirlos, como podría ser una persona ó un poste de luz, por esta razón es útil incorporar sensores capaces de detectar objetos delgados o no sólidos, de tal manera que se logre una deformación en la trayectoria con la finalidad de evitar una colisión.

Incorporación de un Sistema de Visión: Para reducir los errores, sobre todo en los cambios de dirección, se sugiere realimentar el algoritmo por medio de visión artificial, reconociendo de esta manera los carriles, semáforos e intersecciones, que es donde se presentan errores más grandes.

Tomando como base el trabajo realizado en esta tesis, estas recomendaciones serán útiles para mejorar la tecnología descrita e incursionar en la automatización de los automóviles, que en mi opinión es el futuro de los robots móviles y el robot más inteligente que en el futuro las personas tendrán en su hogar.

El algoritmo de navegación, como se ha descrito, sólo emplea la orientación y la ubicación del "smartphone" para orientar y ubicar al robot; para realizar el desplazamiento sólo manda a la electrónica del robot cuatro datos, el ángulo de las llantas delanteras y las velocidades angulares de las llantas traseras, la electrónica del robot toma estos valores y se encarga de ejecutarlas, sin embargo, es posible crear una nueva interfaz electrónica que tome estos cuatro valores y que controle el eje de la dirección del automóvil, la aceleración y la caja de velocidades para obtener los parámetros deseados, dicho en otras palabras, el sistema desarrollado es totalmente adaptable a un vehículo real realizando

modificaciones pequeñas a un vehículo convencional, otra opción sería remplazar la tradicional dirección Ackerman y el diferencial de las llantas traseras por sistemas que permitan controlar al vehículo utilizando señales eléctricas en lugar de mecanismos, estos sistemas son conocidos como "*Drive by Wire*" y "*Differential Drive*" respectivamente, sin embargo, esto implicaría realizar modificaciones más profundas, que permitirían una mayor capacidad de automatización.

6. APÉNDICES

Apéndice A, código programado en el microcontrolador

```
#include <16F877A.h>
#device adc=10

#fuses HS,NOWDT
#use delay (clock = 20000000)
#use rs232(baud=9600, xmit=pin_c6, rcv=pin_c7, bits=8, parity=N)
#priority timer1, timer0, rb, rda

#BYTE TRISB=0x86
#BYTE PORTB=0x06
#BYTE TRISD=0x88
#BYTE PORTD=0x08

//Lo que hace ésta función es recibir un carácter tipo char y restarle el carácter 0,
#define toint(c) ((int)((c)-'0'))

int ch[13]={'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x'};
int manda[3];
int manda2[6];
int distancia[6]={0, 0, 0, 0, 0, 49};
int i=0;
int servomotorA = 1;
int servo1 = 0;
int servo2 = 0;

//Destinadas al control de la velocidad angular de las llantas de tracción traseras.
//Destinadas al conteo de los pulsos.
int vmotord=0;
int32 pulsosmd=0;
int vmotori=0;
int32 pulsosmi=0;

//Destinadas a contener la velocidad angular de las llantas traseras.
int16 velWd=0;           //En rad/s
int16 velWi=0;           //En rad/s

int velWdd=0;
int velWii=0;

//Destinada al cálculo de las revoluciones de las llantas en rad/s.
int tiempoRev=0;
float operacionesd=0.0;
```

```

float operacionesi=0.0;
int16 revolucionesd=0;           //En rad/s
int16 revolucionesi=0;           //En rad/s

//Variables para la señal de PWM que controla la velocidad angular de las llantas de tracción.
int32 valord=0;
int32 valori=0;
int banderai=0;
int banderaii=0;
int banderad=0;

//Variables para la lectura analógica del sensor ultrasónico.
int16 q;
float p;
int16 tiempo=0;

//Cada 0.5 ms
//int timer0 = 0x1B;

//Variables encargadas del ángulo de las llantas de los servomotores.
//Variables encargadas de la lectura serial del ángulo de las llantas delanteras.
unsigned long angulod=90;
unsigned long anguloi=90;

//Variables destinadas al cálculo del ciclo de trabajo para controlar el PWM de los servomotores.

int16 pulsoAltod = 58036;
int16 pulsoBajod = 23036;
int16 pulsoAltoi = 58036;
int16 pulsoBajoi = 23036;

//Variables encargadas del ángulo del sensor de proximidad.
//Variables encargadas de la lectura serial del ángulo del sensor.
unsigned long angulos=180;

//Variables destinadas al cálculo del ciclo de trabajo para controlar el PWM del servomotor.
int16 pulsoAltos = 58036;
int16 pulsoBajos = 23036;

//Pin D7 para llanta derecha
//Pin D6 para llanta izquierda.
//Pin D5 para Sensor.
#INT_TIMER1
void timer1_isr()                 //Interrupción para mover los servomotores.
{
    if(bit_test(PORTD,4)==0)

```

```

{
    bit_set(PORTD,4);
}
else
{
    bit_clear(PORTD,4);
}

if(servo2==0 && servo1==0)                //Llanta Derecha.
{
    if(bit_test(PORTD,7)==0)
    {
        set_timer1(pulsoAltod);
        bit_set(PORTD,7);
    }
    else
    {
        set_timer1(pulsoBajod);
        bit_clear(PORTD,7);
        servo1=1;
        servo2=0;
    }
}
if(servo2==0 && servo1==1)                //Llanta izquierda.
{
    if(bit_test(PORTD,6)==0)
    {
        set_timer1(pulsoAltoi);
        bit_set(PORTD,6);
    }
    else
    {
        set_timer1(pulsoBajoi);
        bit_clear(PORTD,6);
        servo1=0;
        servo2=1;
    }
}
if(servo2==1 && servo1==0)                //sensor
{
    if(bit_test(PORTD,5)==0)
    {
        set_timer1(pulsoAltos);
        bit_set(PORTD,5);
    }
    else

```

```

    {
        set_timer1(pulsoBajos);
        bit_clear(PORTD,5);
        servo1=1;
        servo2=1;
    }
}
if(servo2==1 && servo1==1)                //auxiliar
{
    if(servomotorA==0)
    {
        set_timer1(pulsoAltos);
        servomotorA=1;
    }
    else
    {
        set_timer1(pulsoBajos);
        servomotorA=0;
        servo1=0;
        servo2=0;
    }
}
}

//Pin B7 para llanta derecha.
//Pin B6 para llanta izquierda.
#INT_RB
void RB_isr(void)                          //función de interrupción de los bits 4 al 7, del puerto B.
{
    if(bit_test(PORTB,7)==1&&vmotord==0)
    {
        vmotord=1;
        pulsosmd=pulsosmd+1;
    }
    else
    {
        if(bit_test(PORTB,7)==0&&vmotord==1)
        {
            vmotord=0;
        }
    }
}
}

```

```

#INT_TIMER0
void timer0_isr()                                //Se cuenta cada medio milisegundo aproximadamente.
{
    disable_interrupts(INT_TIMER0);              //Se deshabilita la interrupción de tiempo.
    i=i+1;
    if(i==2)                                     //Se ha contado un milisegundo.
    {
        tiempo=tiempo+1;
        i=0;
        if(tiempo==25)                          //Se cuentan 25 milisegundos.
        {
            //Revoluciones del motor de las llanatas de traccion
            tiempoRev = tiempoRev+1;

            if(tiempoRev==10)                    //Se contaron 250 ms.
            {
                if(pulsosmd > valord)
                {
                    bit_set(PORTD,3);

                    if(velWdd > 1)
                    {
                        if(((pulsosmd/valord)>0.5)&&((pulsosmd/valord)<1))
                        {
                            velWdd = velWdd-2;
                        }
                        else
                        {
                            velWdd = velWdd-((int)((pulsosmd/valord)));
                        }
                    }

                    if(velWdd<2)
                    {
                        velWdd=1;
                    }
                }
            }
            else
            {
                bit_clear(PORTD,3);
                if(velWdd<250)
                {
                    if(pulsosmd<=1)
                    {
                        velWdd = velWdd+1;
                    }
                }
            }
        }
    }
}

```

```

        else
        {
            velWdd = velWdd+((int)((valord/pulsosmd)));
        }
    }
    if(velWdd >= 250)
    {
        velWdd = 250;
    }
}
if(pulsosmi > valori)
{
    bit_set(PORTD,2);

    if(velWii > 1)
    {
        if(((pulsosmi/valori)>0.5)&&((pulsosmi/valori)<1))
        {
            velWii = velWii-2;
        }
        else
        {
            velWii = velWii-((int)((pulsosmi/valori)));
        }
    }

    if(velWii<2)
    {
        velWii=1;
    }
}
else
{
    bit_clear(PORTD,2);
    if(velWii<250)
    {
        if(pulsosmi<=1)
        {
            velWii = velWii+1;
        }
        else
        {
            velWii = velWii+((int)((valori/pulsosmi)));
        }
    }
}
if(velWii >= 250)

```

```

    {
        velWii = 250;
    }
}

if(velWdd > 250)
{
    velWdd=250;
}
if(velWdd <= 2)
{
    velWdd=1;
}
if(velWii > 250)
{
    velWii=250;
}
if(velWii <= 2)
{
    velWii=1;
}

if(velWd==0)
{
    set_pwm1_duty(0);
}
else
{
    set_pwm1_duty(velWdd);
}

if(velWi==0)
{
    set_pwm2_duty(0);
}
else
{
    set_pwm2_duty(velWii);
}

pulsosmi=0;
pulsosmd=0;
tiempoRev = 0;
}
tiempo = 0;
}

```

//Se cuenta el número de pasos.
//Se reinicia el conteo.

```

    }
    set_timer0(0x67);           //Se reinicia el timer 0.
    enable_interrupts(INT_TIMER0); //Habilita la interrupción del timer0.
}

//Interrupción del puerto serial.
#INT_RDA
void rda_isr()
{
    //Para Char
    //Para string, lee hasta encontrar un enter, 13 en código ascii
    gets(ch);                  //Se almacena los datos provenientes del smartphone.

    //El celular se conecta con el robot
    if(ch[0]=='p')
    {
        manda[0]='c';
        manda[1]='o';
        manda[2]='n';
        puts(manda);
        bit_set(PORTD,1);
    }
    //El celular se desconecta del robot.
    else if(ch[0]=='a')
    {
        manda[0]='a';
        manda[1]='p';
        manda[2]='a';
        puts(manda);
        bit_clear(PORTD,1);
    }

    //Se pide la distancia número 0
    else if(ch[0]=='z')
    {
        if(distancia[0]<100)
        {
            putc('0');
        }
        if(distancia[0]<10)
        {
            putc('0');
        }
        printf("%u",distancia[0]);
    }
}

```

```

//Se pide la distancia número 1.
else if(ch[0]=='u')
{
    if(distancia[1]<100)
    {
        putc('0');
    }
    if(distancia[1]<10)
    {
        putc('0');
    }
    printf("%u",distancia[1]);
}
//Se pide la distancia número 2.
else if(ch[0]=='d')
{
    if(distancia[2]<100)
    {
        putc('0');
    }
    if(distancia[2]<10)
    {
        putc('0');
    }
    printf("%u",distancia[2]);
}

//Ángulos de las llantas direccionales centenas.
else if(ch[0]=='0' | |ch[0]=='1' | |ch[0]=='2' | |ch[0]=='3')
{
    //Ángulo de las llantas direccionales.
    angulod=(toint(ch[0])*100)+(toint(ch[1])*10)+(toint(ch[2]));
    anguloi=(toint(ch[3])*100)+(toint(ch[4])*10)+(toint(ch[5]));

    //Velocidad angular de las llantas de tracción.
    velWd=(toint(ch[6])*100)+(toint(ch[7])*10)+(toint(ch[8]));
    velWi=(toint(ch[9])*100)+(toint(ch[10])*10)+(toint(ch[11]));

    //Límites de los ángulos entre 0 y 180 grados.
    if(angulod>180)
    {
        angulod=180;
    }
    if(angulod<0)
    {
        angulod=0;
    }
}

```

```

    }
    if(anguloi>180)
    {
        anguloi=180;
    }
    if(anguloi<0)
    {
        anguloi=0;
    }

    //Límites de las velocidades angulares entre 0 y 300 grados.
    if(velWd > 300)
    {
        velWd = 300;
    }
    if(velWd < 0)
    {
        velWd = 0;
    }
    if(velWi > 300)
    {
        velWi = 300;
    }
    if(velWi < 0)
    {
        velWi = 0;
    }

    valord=(velWd*17/540);/*253/300;
    valori=(velWi*17/540);/*253/300;

    velWdd = (int)velWd;
    velWii = (int)velWi;
    banderai = 0;

    pulsoAltod = (65535-((angulod*(6000/180))+4500));/*+60;
    pulsoBajod = (65535-(25000-((angulod*(6000/180))+4500)));/*-60;

    pulsoAltoi = (65535-((anguloi*(6000/180))+4500));/*+60;
    pulsoBajoi = (65535-(25000-((anguloi*(6000/180))+4500)));/*-60;
    printf("%lu",velWi);
    printf("%u",velWii);
}
}

```

```

#INT_EXT
ext_isr()
{
    if(bit_test(PORTB,0)==1&&vmotori==0)
    {
        vmotori=1;
        pulsosmi=pulsosmi+1;
        ext_int_edge(H_TO_L);
    }
    else
    {
        if(bit_test(PORTB,0)==0&&vmotori==1)
        {
            vmotori=0;
            ext_int_edge(L_TO_H);
        }
    }
}

void main()
{
    TRISB=0xFF;                //Cuatro bits más altos como entrada.
    PORTB=0x00;
    TRISD=0x00;
    PORTD=0x00;
    bit_set(PORTD,0);

    /* PWM PARA PICS DE 8-BIT */

    setup_ccp1(CCP_PWM);        // Configuración de CCP1 como PWM.
    setup_ccp2(CCP_PWM);        // Configuración de CCP2 como PWM.
    setup_timer_2(T2_DIV_BY_16, 255,1);

    //Interrupción del Timer 0
    setup_timer_0(RTCC_INTERNAL | RTCC_DIV_16);    //Configuración del timer 0.
    set_timer0(0x0F);
    //Interrupción del Timer 1
    setup_timer_1(T1_INTERNAL | T1_DIV_BY_1);        //Configuración del timer 0.
    set_timer1(58036);

    setup_adc_ports(AN0_AN1_AN2_AN3_AN4);
    setup_adc(ADC_CLOCK_INTERNAL);

    enable_interrupts(INT_TIMER0);    //Habilita la interrupción del timer0 para mover el
                                     //motor a pasos.
}

```

```

enable_interrupts(INT_TIMER1);           //Habilita la interrupción del timer1 para mover los
                                           //servomotores.
enable_interrupts(INT_RB);               //Habilita interrupción por cambio de estado de los bits
                                           //RB4 al RB7 para lectura del encoder de las llantas.
enable_interrupts(INT_RDA);              //Habilita la interrupción del puerto serie para leer
                                           //datos del bluetooth.

enable_interrupts(INT_EXT);
ext_int_edge(L_TO_H);
enable_interrupts(GLOBAL);               //Habilita interrupción global.

pulsoAltod = (65535-((angulod*(6000/180))+4500));//+60;
pulsoBajod = (65535-(25000-((angulod*(6000/180))+4500)));//-60;

pulsoAltoi = (65535-((anguloi*(6000/180))+4500));//+60;
pulsoBajoi = (65535-(25000-((anguloi*(6000/180))+4500)));//-60;

set_pwm1_duty(0);
set_pwm2_duty(0);

while(1)
{
    if(velWdd >= 250)
    {
        velWdd = 250;
    }
    if(velWdd <= 2)
    {
        velWdd = 1;
    }
    if(velWii >= 250)
    {
        velWii = 250;
    }
    if(velWii <= 2)
    {
        velWii = 1;
    }

    //Sensor lado izquierdo viendo de frente al carro.
    set_adc_channel(0);
    delay_ms(1);
    q=read_adc();                       //Se toma el cero de la medición.
    p=5.0*q/1024.0;                     //Se convierte a voltaje.
    distancia[0]= p*2.54*512/5;          //Se mide la distancia.

```

```

//Sensor de enfrente.
set_adc_channel(1);
delay_ms(1);
q=read_adc();
p=5.0*q/1024.0;
distancia[1]= p*2.54*512/5;

//Sensor lado derecho viendo de frente al carro.
set_adc_channel(2);
delay_ms(1);
q=read_adc();
p=5.0*q/1024.0;
distancia[2]= p*2.54*512/5;

}
}

```

Apéndice B, Instalación de Motodev

En este apéndice se muestra el proceso necesario para la instalación del entorno de programación Motodev, de tal manera que sea posible, para la persona que le interese, programar para el sistema operativo Android utilizando la herramienta proporcionada por la compañía "Motorola".

1. Descargar e instalar el JDK (Kit de Desarrollo de Java)

- 1.1. Acceder a <http://www.oracle.com/technetwork/es/java/javase/downloads/index.html> y descargar el JDK apropiado de acuerdo a la configuración del equipo que se emplee, las siguientes figuras muestran la información contenida en el enlace anterior al momento de su escritura.

JDK 7 and JavaFX Demos and Samples
Demos and samples of common tasks and new functionality available on JDK 7. The source code provided with samples and demos for the JDK is meant to illustrate the usage of a given feature or technique and has been deliberately simplified.

Demos and Samples
[DOWNLOAD](#)

Java SE 6u45
This release brings in security features and bug fixes. Oracle strongly recommends that all Java SE 6 users upgrade to this release. [Learn more](#)

JDK
[DOWNLOAD](#)

JRE
[DOWNLOAD](#)

JDK 6 Docs

- Installation Instructions
- ReadMe
- Release Notes
- Oracle License
- Java SE Products
- Third Party Licenses
- Certified System Configurations

JRE 6 Docs

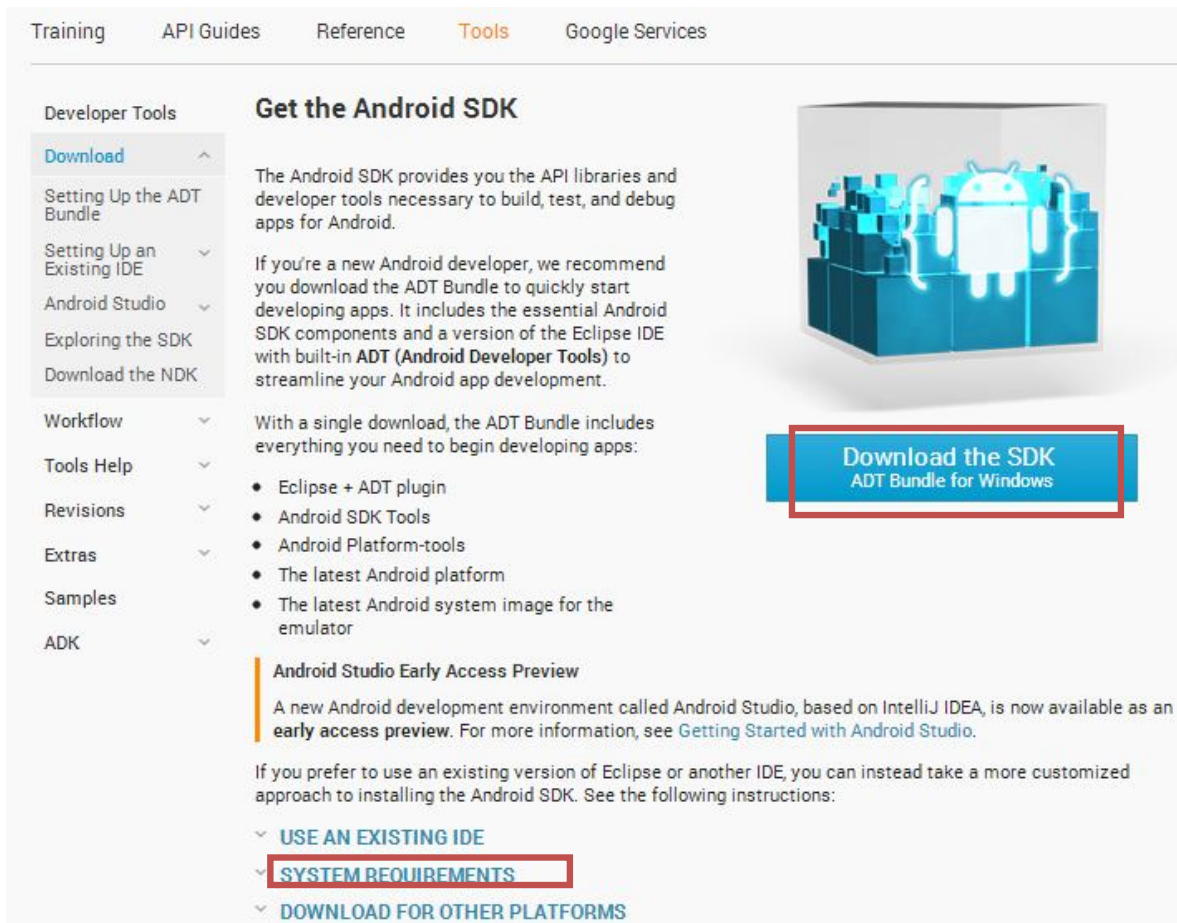
- Installation Instructions
- ReadMe
- Release Notes
- Oracle License
- Java SE Products
- Third Party Licenses
- Certified System Configurations

JDK 6 Demos and Samples
Demos and samples of common tasks and new functionality available on JDK 6. The source code provided with samples and demos for the JDK is

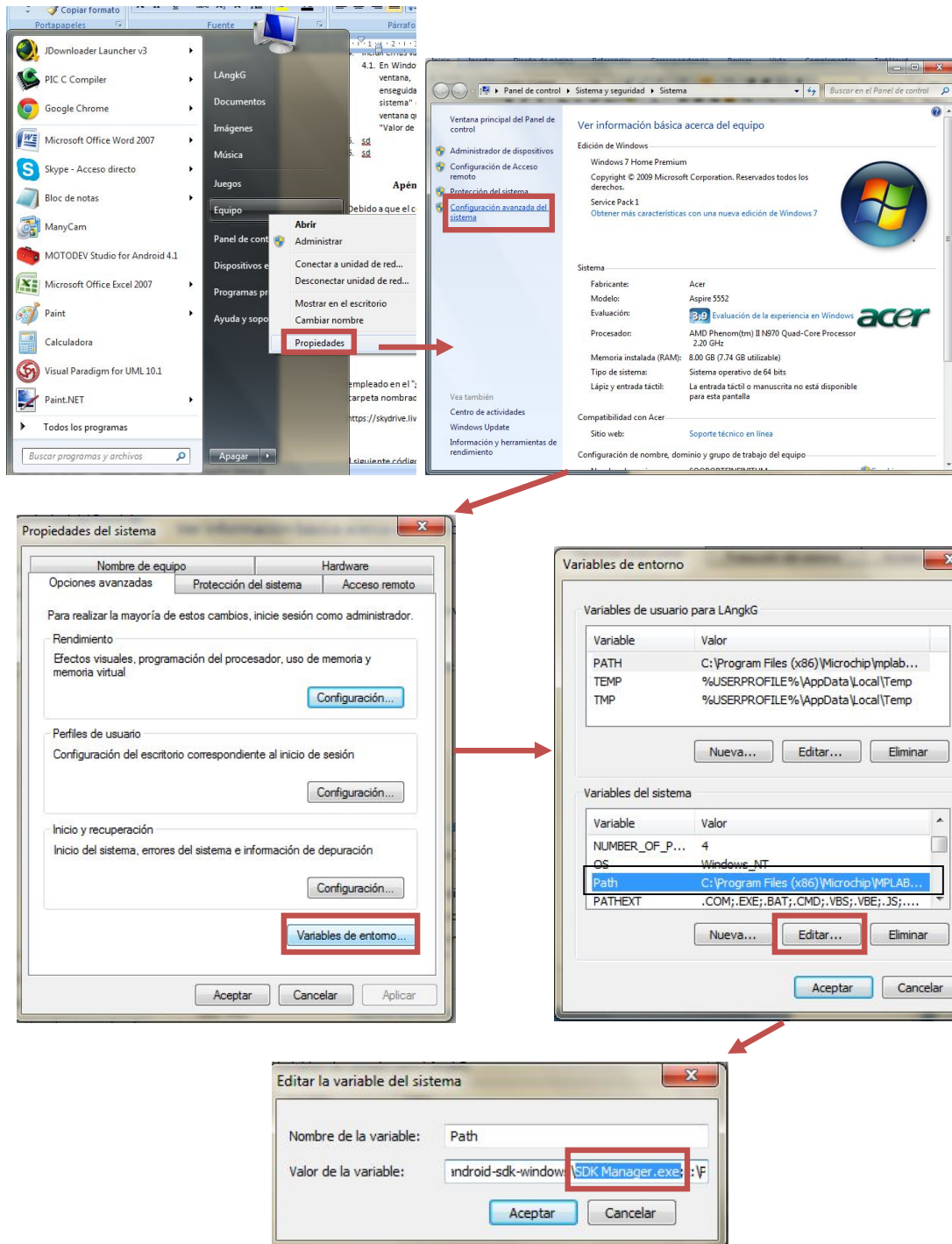
JDK Demos and Samples
[DOWNLOAD](#)

Java SE Development Kit 6 Update 45		
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.		
☐ Accept License Agreement ☒ Decline License Agreement		
Product / File Description	File Size	Download
Linux x86	65.46 MB	jdk-6u45-linux-i586-rpm.bin
Linux x86	68.47 MB	jdk-6u45-linux-i586.bin
Linux x64	65.69 MB	jdk-6u45-linux-x64-rpm.bin
Linux x64	68.75 MB	jdk-6u45-linux-x64.bin
Solaris x86	68.38 MB	jdk-6u45-solaris-i586.sh
Solaris x86 (SVR4 package)	120 MB	jdk-6u45-solaris-i586.tar.Z
Solaris SPARC	73.41 MB	jdk-6u45-solaris-sparc.sh
Solaris SPARC (SVR4 package)	124.74 MB	jdk-6u45-solaris-sparc.tar.Z
Solaris SPARC 64-bit	12.19 MB	jdk-6u45-solaris-sparcv9.sh
Solaris SPARC 64-bit (SVR4 package)	15.49 MB	jdk-6u45-solaris-sparcv9.tar.Z
Solaris x64	8.5 MB	jdk-6u45-solaris-x64.sh
Solaris x64 (SVR4 package)	12.23 MB	jdk-6u45-solaris-x64.tar.Z
Windows x86	69.85 MB	jdk-6u45-windows-i586.exe
Windows x64	59.96 MB	jdk-6u45-windows-x64.exe
Linux Intel Itanium	53.89 MB	jdk-6u45-linux-ia64-rpm.bin
Linux Intel Itanium	60.5 MB	jdk-6u45-linux-ia64.bin
Windows Intel Itanium	57.8 MB	jdk-6u45-windows-ia64.exe

2. instalar Motodev
 - 2.1. Ingresar a <http://xplicando.blogspot.mx/2013/02/motodev-studio-for-android.html> y seleccionar la opción que corresponda de acuerdo a la configuración del equipo a emplear.
3. Instalar el SDK de Android (Kit de Desarrollo de Software)
 - 3.1. Acceder a <http://developer.android.com/sdk/index.html> y descargar el SDK adecuado, de acuerdo al sistema operativo que contenga el equipo a emplear para programar.

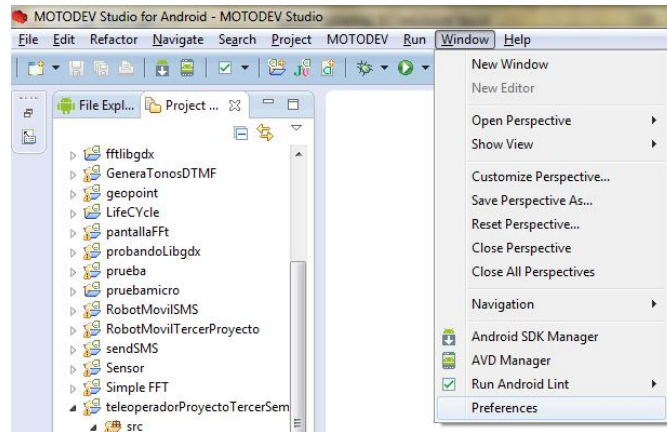


4. Incluir en las variables "Path" del sistema operativo la ruta donde se instaló el SDK de Android.
 - 4.1. En Windows dar click derecho sobre el icono "equipo", posteriormente propiedades, se abrirá una ventana, en ella seleccionar la opción de "Configuración Avanzada del Sistema", enseguida presionar el botón llamado "Variables de entorno", en la sección "Variables del sistema" dar click en la fila con nombre "CLASSPATH", pulsar "Editar", por último en la ventana que aprese agregar la ruta del SDK Manger.exe al final de la sección con nombre "Valor de la variable", aceptar todas las ventanas que se abrieron.

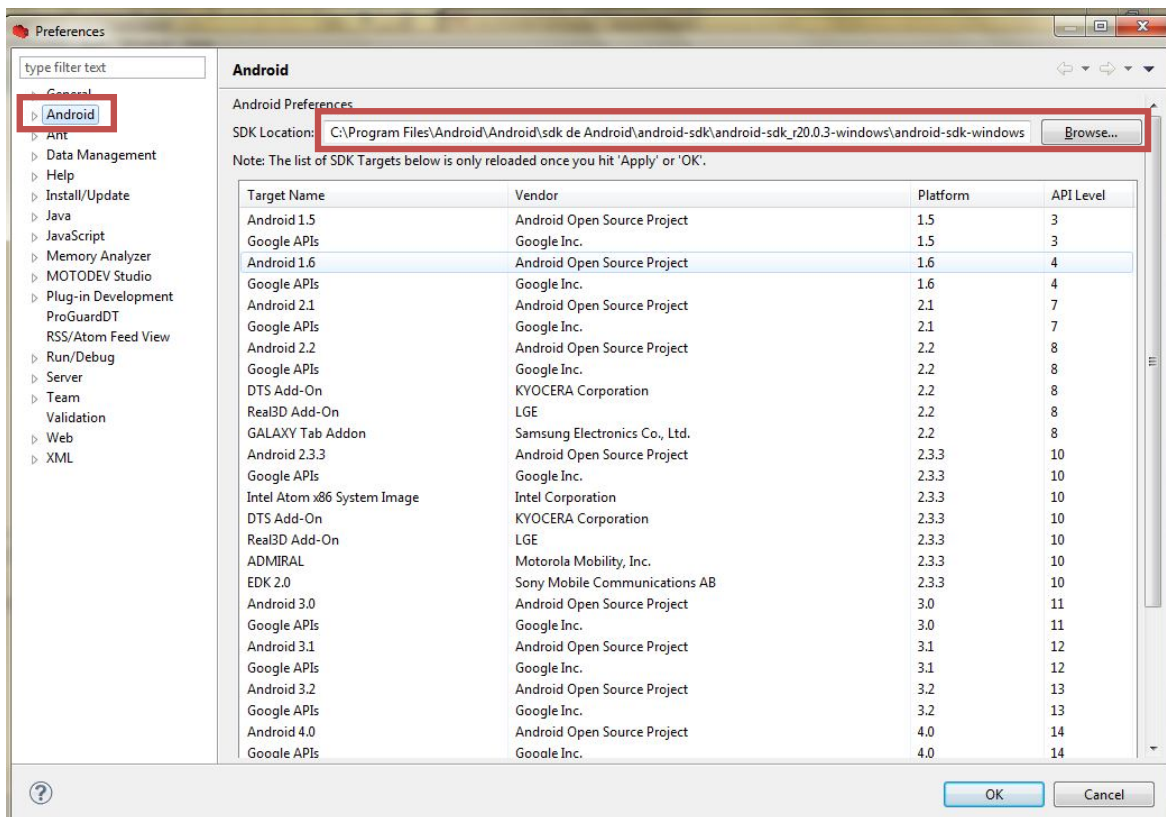


5. Configuración del entorno

5.1. Abrir Motodev, ir al menú de "Window" y seleccionar "Preferencias".



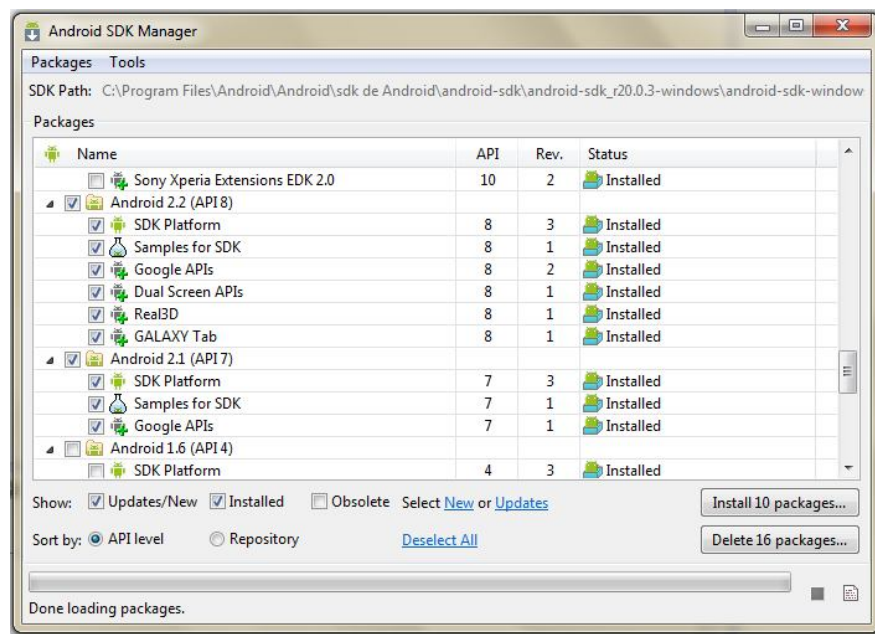
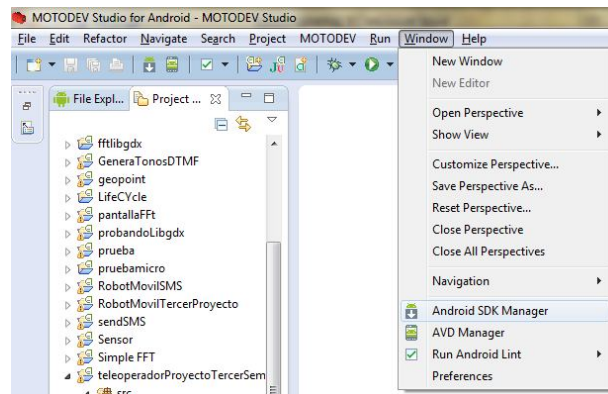
5.2. En la ventana que aparece, seleccionar "Android" y verificar que se encuentre la ubicación del SDK de Android en el campo de edición, sino es así, apretar el botón "Browse" y seleccionar la ubicación del SDK de Android.



6. Instalación de paquetes disponibles

6.1. Abrir Motodev, ir al menú de "Window" y seleccionar "Android SDK Manager"

6.2. Seleccionar los paquetes que se necesitan para el desarrollo, no es necesario instalar todos, sino los que se ocuparan, por ejemplo el API de Google con el cual es posible utilizar los mapas de Google.

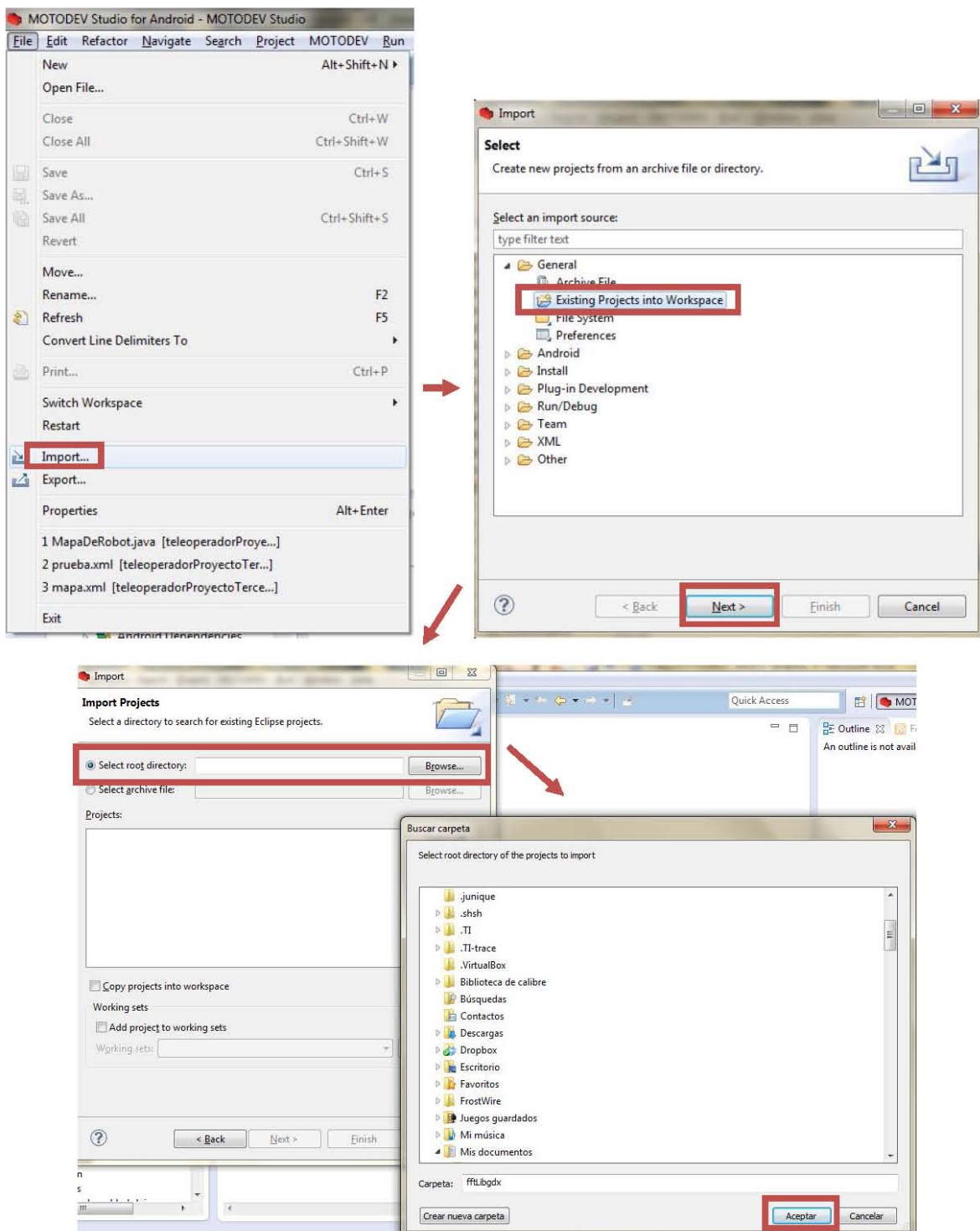


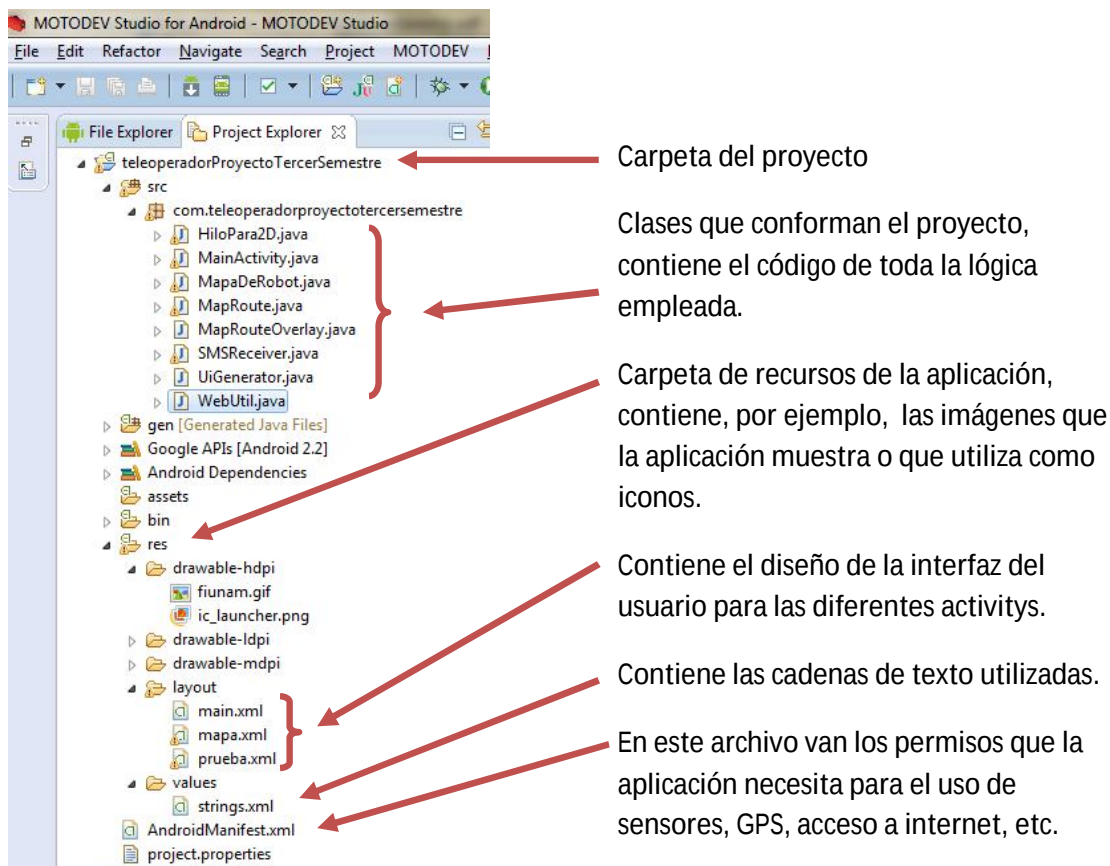
7. Con estos pasos se tiene todo listo para empezar a desarrollar aplicaciones para el sistema operativo Android.

Apéndice C, abrir un archivo de un proyecto de Android

En esta sección se presenta el método para abrir un proyecto previamente realizado y poder visualizar todo el código que conforma a la aplicación.

1. Ir al menú "File" del entorno de programación, ya sea Motodev o de Eclipse, seleccionar la opción "Importar".
2. Seleccionar el directorio en donde será almacenado el proyecto y apretar siguiente.
3. Seleccionar "Select root directory" y apretar el botón "Browser" para seleccionar el archivo que contiene el proyecto.
4. Apretar finalizar.
5. El proyecto se agregará al "Explorador de Proyectos", con esto es posible ver todo el código e imágenes que conforman a la aplicación.





Apéndice D, código programado en el smartphone del usuario y del robot móvil

Debido a que el código empleado contiene un gran número de líneas se consideró factible subir el proyecto a internet, de tal manera que se tenga acceso a él. Para acceder al código empleado en los "smartphone" se debe acceder a la siguiente liga, para el código del usuario, descargar la carpeta nombrada como "**usuario**", para el código del robot, descargar la carpeta llamada "**robot**":

<https://skydrive.live.com/?lc=2058#cid=49DCC946BF6FC4D9&id=49DCC946BF6FC4D9%21105>

Para visualizar su contenido siga las instrucciones del apéndice C.

TECNOLOGÍAS MÓVILES APLICADAS AL SEGUIMIENTO DE RUTAS EN VEHÍCULOS AUTÓNOMOS

Luis Angel Cabral Guzmán, Víctor J. González-Villela

Departamento de Mecatrónica, Facultad de Ingeniería, UNAM. Cd. Universitaria, MÉXICO,
04510, D.F.

luisangelcg007@gmail.com; vjgv@unam.mx

RESUMEN

Se presenta un prototipo de vehículo autónomo utilizando tecnología móvil, aprovechando los avances tecnológicos que estos dispositivos tienen. Para obtener una navegación autónoma se requiere conocer el área de trabajo, en este caso la ciudad, estos datos son adquiridos por medio de mapas digitales, adicionalmente es necesario un algoritmo capaz de generar la ruta a seguir, la propuesta de este trabajo es aprovechar no solo las nuevas tecnologías y sensores que los teléfonos inteligentes tienen, como procesos GSM y rastreo vía GPS, si no también las bases de datos existentes, en este caso "*Google Maps*", para la obtención automática de la ruta a seguir por el robot autónomo.

La eficiencia de esta tecnología y del algoritmo empleado, es validada realizando un recorrido autónomo y midiendo el error que se tiene comparando la trayectoria realizada por el robot con la ruta deseada.

1. INTRODUCCIÓN

Obtener la posición y orientación de un vehículo, así como determinar la mejor ruta a seguir dependiendo de la ubicación, orientación y destino del automóvil tiene una importancia especial para muchas aplicaciones en Sistemas de Transporte Inteligente, estas tareas se han intentado solucionar aplicando diversos dispositivos tecnológicos en forma discreta, como cámaras y diversos sensores, con el afán de proporcionar información útil, de manera común, a una computadora, ya sea de escritorio o personal. Sin embargo, dispositivos no discretos pueden proporcionar mucha información útil para tal fin a un costo menor, considerando que los dispositivos deben de viajar dentro del vehículo. Los celulares inteligentes, con sistemas de navegación, rastreo, comunicación GSM, y conexión a datos, entre otras características, proveen una solución potencial para resolver el problema de navegación autónoma en vehículos, que a primera vista parece ser costoso y complicado debido a la gran cantidad de dispositivos que actualmente se consideran para resolver este problema.

En el presente trabajo se presenta una forma novedosa de aprovechar las tecnologías emergentes para el desarrollo de un prototipo de vehículo autónomo. La fusión de las tecnologías descritas a continuación tales como "*Google Maps*", "*Smartphone*" y GPS son herramientas útiles que no están limitadas solo para propósitos de recreación, si no que pueden trasladarse a aplicaciones en sistemas Mecatrónicos y de forma particular a Sistemas de Transporte Inteligentes, resumiéndose todo al proceso simple de escribir una aplicación para el teléfono inteligente, aprovechando los recursos en éstos. A continuación se muestra el desarrollo de un prototipo para un vehículo autónomo capaz de llegar de un punto a otro utilizando la tecnología móvil, "*Smartphone o Tablet*", aprovechando sus sensores y medios de comunicación incorporados en éstos.

PALABRAS CLAVE: Google Maps, GPS, GSM, Smartphone, Vehículo Autónomo, Robot.

2. DESCRIPCIÓN DEL SISTEMA

La Figura 1, muestra el sistema en general, se compone de dos "Smartphone", uno para el usuario y otro para el robot móvil.

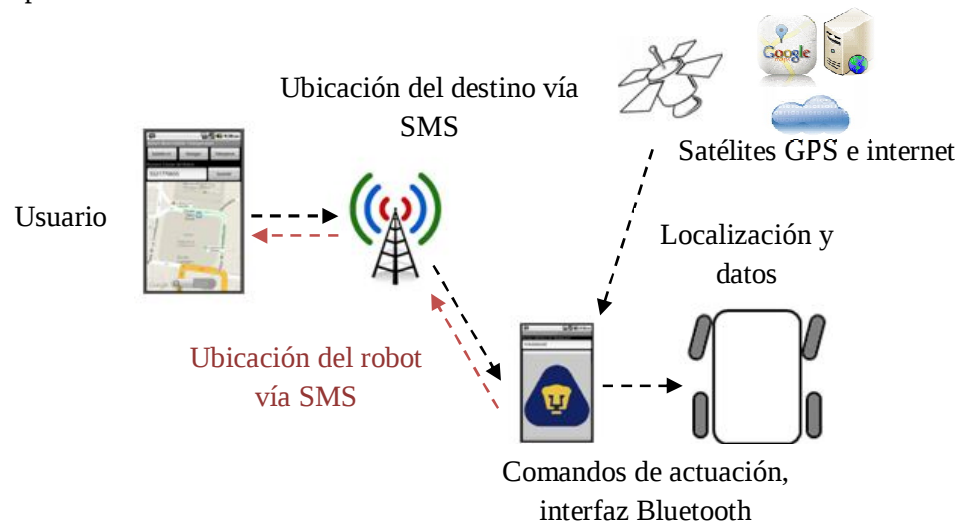


Figura 1. Arquitectura general propuesta para sistema.

Se desarrollaron dos aplicaciones para el sistema operativo "Android", una para el usuario y otra para el robot móvil. Las aplicaciones transfieren instrucciones entre ellas por medio de mensajes SMS, utilizando las redes de telefonía celular comerciales, la Figura 2, muestra los componentes de cada "Smartphone", y se utilizan como unidades de procesamiento principal y como interfaz de comunicación para el usuario.



Figura 2. Pantalla principal del "Smartphone" del usuario (a) y del robot móvil (b).

En la Tabla 1, Se muestra una descripción del funcionamiento de cada elemento de los dispositivos móviles.

Tabla 1. Componentes de las aplicaciones desarrolladas

Interfaz del usuario	
Elemento	Descripción
Botón Satélite	Permite cambiar la vista del mapa de modo satélite a mapa, mostrando una vista de la ciudad desde arriba.
Botón Navegar	Permite navegar sobre el mapa arrastrando el dedo sobre él, al tocar el botón se cambia esta función, permitiendo seleccionar una meta para el robot.
Botón Operar	Seleccionado un destino para el robot sobre el mapa, este botón transfiere las instrucciones hacia el robot vía SMS.
Campo de Edición	Al tocarlo se visualiza el teclado numérico para colocar el número celular del robot a 10 dígitos.
Botón Guardar	Verifica la estructura del número colocado en el campo de edición y lo guarda.
Mapa	Se visualiza un mapa obtenido desde "Google Maps".
Ubicación del robot	Punto que ubica al robot sobre el mapa.
Ubicación de la meta	Punto que ubica el destino final del robot sobre el mapa.
Trayectoria	Línea que dibuja la trayectoria estimada que el robot debe de seguir.
Interfaz del robot móvil	
Elemento	Descripción
Campo de Edición	Al tocarlo se visualiza el teclado numérico para colocar el número celular del usuario a 10 dígitos.
Botón Guardar	Verifica la estructura del número colocado en el campo de edición y lo guarda, posteriormente inicia la conexión "bluetooth" con el robot para controlar sus motores.

En la pantalla del usuario se presenta un mapa de "Google Maps" en donde se ubica al robot móvil, por medio de esta es posible seleccionar el punto de destino del robot y con tan solo apretar el botón llamado "**Operar**", las coordenadas de latitud y longitud de la meta seleccionada son transferidas al prototipo por medio de un mensaje SMS, generado de manera automática por la aplicación desarrollada. El "Smartphone" del robot, al recibir las intenciones del usuario, crea una trayectoria hasta su objetivo con ayuda de "Google Maps", dicha trayectoria estará compuesta de un cierto número de submetas, en coordenadas de latitud y longitud, de tal manera que al ir alcanzando cada una de ellas se estaría recorriendo la trayectoria a seguir. Empleando un algoritmo conocido como "*Campos Potenciales*" [2] el robot móvil empieza a recorrer la trayectoria obtenida, basándose en la información de un compás electrónico y GPS embebidos en el sistema móvil celular. El algoritmo de control empleado se encarga de realizar los cálculos correspondientes para las velocidades angulares de sus llantas traseras y el ángulo de dirección de las llantas delanteras del robot, de tal manera que se logre dirigir al punto determinado como meta.

2.1. Navegación

El algoritmo empleado es conocido como "*Campos Potenciales*"[2], en la Figura 3, es posible observar los parámetros que se involucran para llevar al robot de una ubicación a otra, para ello, el primer paso es calcular la distancia que separa al robot de su meta, esto se realiza aplicando la ecuación (1), calculado este parámetro, que se representa por el vector d_g y simboliza una fuerza

virtual de atracción que dirige al robot, se procede a calcular el ángulo θ_g , que es el ángulo formado por el eje de latitud y el vector d_g , esto se realiza empleando la ecuación número (2).

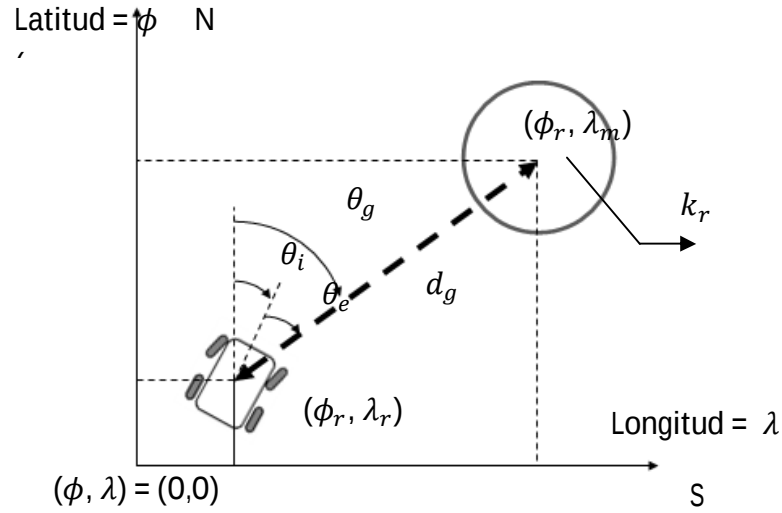


Figura 0. Posición del robot y de la meta a alcanzar, adaptado de [2]

$$d_g = \sqrt{(\phi_m - \phi_r)^2 + (\lambda_m - \lambda_r)^2} \quad (1)$$

$$\theta_g = \text{Arctan}\left(\frac{\lambda_m - \lambda_r}{\phi_m - \phi_r}\right) \quad (2)$$

Sin embargo, el robot tiene un ángulo θ_i , el cual está medido desde el eje de latitud, obtenido gracias al sensor de orientación del "Smartphone", por lo tanto se debe de calcular el ángulo que realmente debe de girar desde su orientación actual hasta orientarse a su objetivo, este ángulo es θ_e y está dado por la ecuación (3).

$$\theta_e = \theta_g - \theta_i \quad (3)$$

Con las variables dadas por las ecuaciones (1) y (3), proseguimos a calcular la velocidad lineal y angular del robot móvil. Para la velocidad lineal:

$$V_R = \begin{cases} V_{max} & \text{si } |d_g| > k_r \\ \frac{V_{max}}{k_r} d_g & \text{si } |d_g| \leq k_r \end{cases} \quad (4)$$

Donde:

- V_{max} : Velocidad máxima del robot.
- V_R : Velocidad lineal del robot.
- k_r : Es el radio del área para que el robot frene.

Para la velocidad angular:

$$W_R = W_{max} * \text{Sen}(\theta_e) \quad (5)$$

Donde:

- W_{max} : Velocidad angular máxima del robot.
- W_R : Velocidad angular del robot.

2.2. Prototipo de Vehículo

El prototipo es un robot móvil, se puede apreciar en la Figura 4, tiene una arquitectura *Ackerman* tal como un automóvil convencional, dicho robot móvil cuenta con dos llantas traseras centradas fijas motorizadas y dos llantas delanteras centradas orientables comandadas por servomotores. A partir de la Figura 5, se realizó un análisis cinemático, en éste se involucran las dimensiones del robot móvil, como su largo " l ", su ancho " $2b$ " y el radio de sus llantas " r_{llx} ", además de las posiciones angulares de las llantas delanteras " α_x " y diversas velocidades lineales " V_x " y angulares " W_x ".

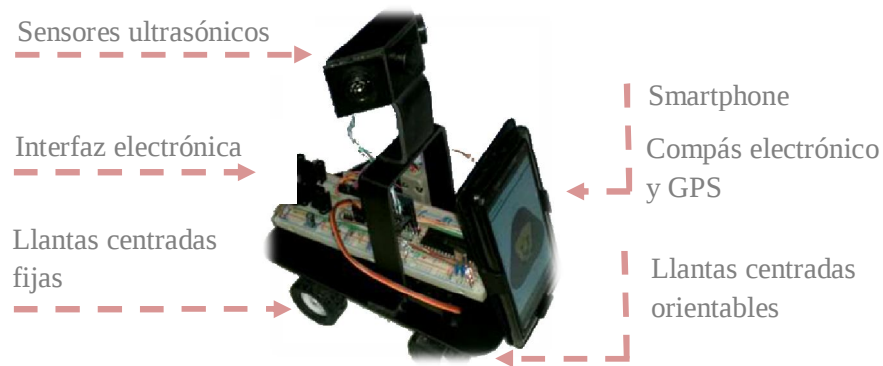


Figura 4. Robot móvil autónomo empleado

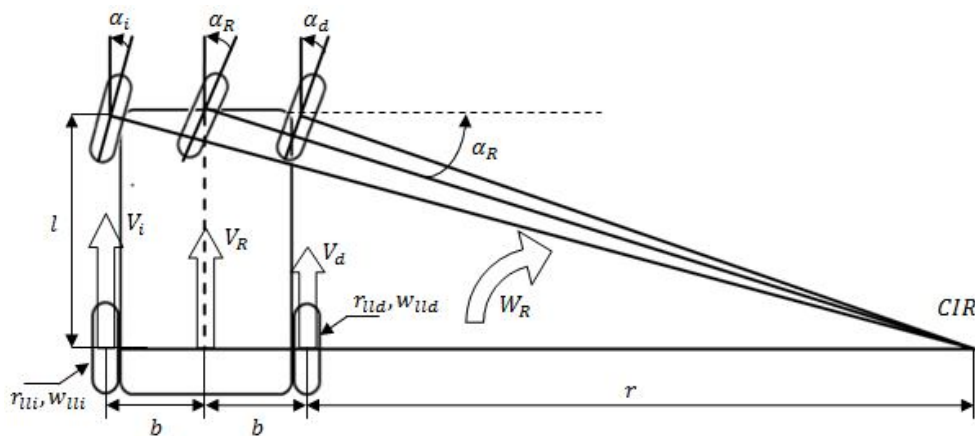


Figura 5. Cinemática de un robot móvil en configuración Ackerman

Las ecuaciones (6) - (7), obtenidas del análisis cinemático, proporcionan información de las velocidades angulares de las llantas traseras derecha e izquierda y la posición angular de las llantas

delanteras derecha e izquierda, respectivamente, dependiendo de la velocidad lineal " V_R " y angular " W_R " que el robot necesita, dichas velocidades están dadas por (4) y (5), respectivamente.

$$w_{ld} = \frac{1}{r_{ld}} (V_R - W_R b) \quad (6)$$

$$w_{li} = \frac{1}{r_{li}} (V_R + W_R b) \quad (7)$$

$$\alpha_d = \text{atan}\left(\frac{lW_R}{V_R - W_R b}\right) \quad (8)$$

$$\alpha_i = \text{atan}\left(\frac{lW_R}{V_R + W_R b}\right) \quad (9)$$

El algoritmo de control, conocido como "*Campos Potenciales*", y el cálculo de las ecuaciones (6) - (9) se computarizan dentro del "*Smartphone*" del robot. En la Tabla 2, se aprecian los valores que toman los parámetros involucrados en las ecuaciones (6) - (9) para el caso específico del robot que se emplea.

Tabla 2. Parámetros y sus valores de la cinemática del robot móvil

Parámetro	Descripción	Valor
V_R	Velocidad lineal del robot	Valor máximo = 0.401 m/s
W_R	Velocidad angular del robot	Valor máximo = 4.41 rad/s
b	Un medio del ancho del robot móvil	0.05 m
l	Largo del robot móvil	0.15 m
r_{ld}	Radio de la llanta trasera derecha	0.02 m
r_{li}	Radio de la llanta trasera izquierda	0.02 m

2.3. Interfaz Electrónica

Entendido como es que se realiza la navegación, debe diseñarse la interfaz electrónica que permita conectar el robot móvil al "*Smartphone*". Se utiliza un módulo "*bluetooth*" BT0417C como interfaz entre el teléfono inteligente y un microcontrolador PIC16F877A, la Figura 6, muestra la electrónica empleada.

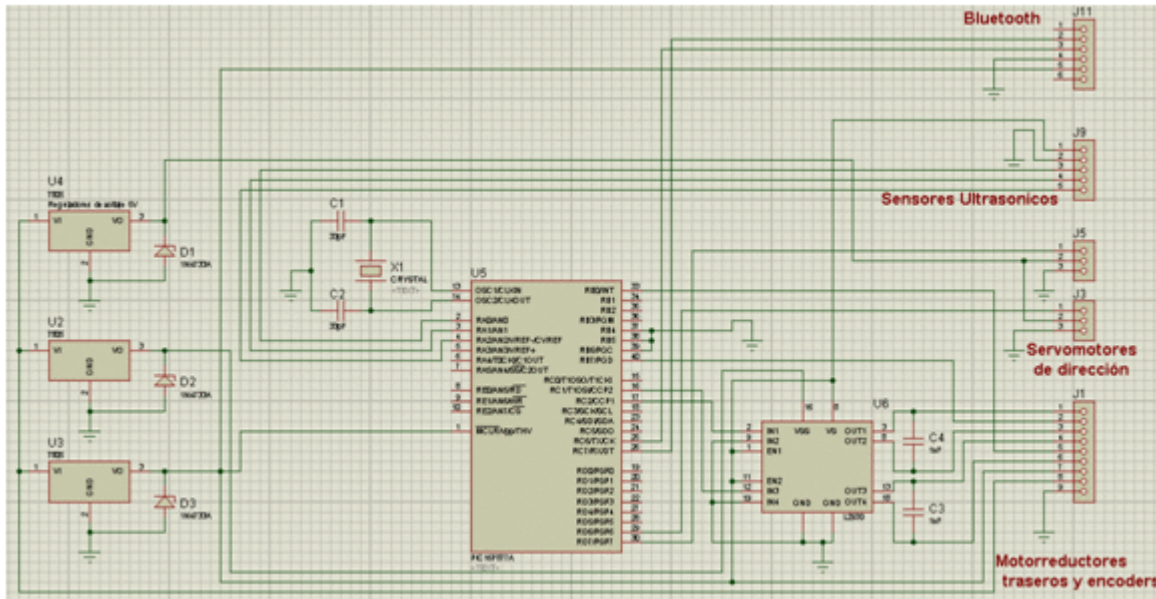


Figura 6. Diagrama de la interfaz electrónica del robot móvil

El PIC16F877A recibe cuatro datos del "Smartphone" del robot; el ángulo de las llantas centradas orientables derecha e izquierda y las velocidades angulares de la llantas centradas fijas traseras derecha e izquierda, la lógica empleada dentro del microcontrolador se encarga de orientar los servomotores delanteros de acuerdo a los ángulos indicados y con ayuda de los encoders colocados en las llantas traseras, puede supervisar que sus velocidades angulares sean las correctas.

3. RESULTADOS

Se lograron diseñar dos aplicaciones para los dispositivos móviles capaces de interactuar entre ellas por medio de mensajes SMS, una de ellas, la aplicación del robot móvil, logró controlar de manera aceptable su recorrido, trasladándose de un punto a otro con la lectura del sensor GPS y del sensor de orientación embebidos en el mismo "Smartphone". La Figura 7, muestra la trayectoria realizada por el prototipo, este se colocó en un lugar fijo, "punto 1", y se realizó un recorrido con una trayectoria formada por "Google Maps" para evaluar su desempeño, en dicha figura se muestra el trayecto realizado por el prototipo de vehículo autónomo en puntos rojos, en conjunto con la trayectoria deseada en línea azul.



Figura 7. Trayectoria de referencia en línea azul y trayecto realizado por el robot en puntos rojos

La Figura 8, muestra el error que se obtuvo en el experimento, es posible observar que en el punto 3 es donde se tiene un error más grande, llegando a ser de más de 3 metros.

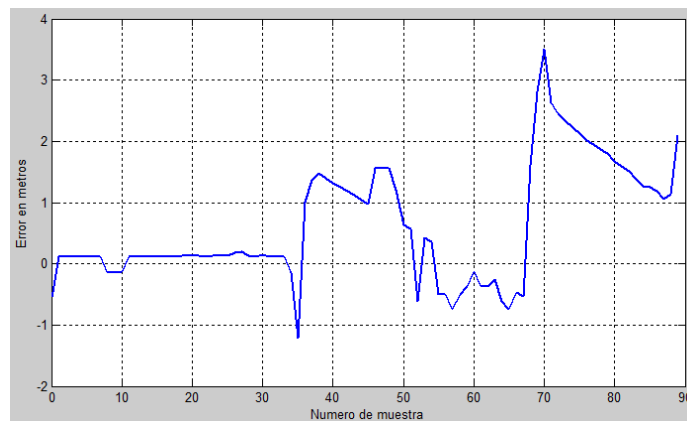


Figura 8. Error de seguimiento de la ruta de referencia obtenido

4. CONCLUSIONES

El algoritmo de control ha sido probado en plataformas de escala pequeña, en donde la posición del robot y de la meta son detectadas por medio de visión artificial, sin embargo el presente trabajo ha servido para demostrar que un sistema con características similares funciona, empleando un sensor GPS, comunicación GSM, sensor de orientación, entre otras tecnologías embebidas en un dispositivo móvil, de tal manera que es posible experimentar con otros robots móviles más complejos, como un automóvil, y así crear Sistemas de Transporte Inteligentes con tecnologías fáciles de conseguir y de uso común, como lo son los "Smartphone".

La aplicación de un "Smartphone" como panel de control o computadora de procesamiento, ofrece la posibilidad de crear sistemas con enorme potencial tanto de control, proceso y almacenaje de información. Se cuenta también con un alto nivel de autonomía, movilidad y portabilidad del software, así como instrumentación de sensores de navegación, entre otros, además, se utilizan las

redes telefónicas de las empresas particulares, reduciendo el costo y haciendo más sencilla la supervisión de la navegación del prototipo.

5. AGRADECIMIENTOS

El primer autor agradece a todas las personas que contribuyeron con su tiempo, amistad y paciencia para la realización de este proyecto, especialmente a sus padres, hermanos, novia, amigos y profesores.

Los autores agradecen el apoyo brindado por la DGAPA, UNAM, a través del proyecto PAPIIT IN115811, con título: "Investigación y desarrollo en sistemas mecatrónicos: robótica móvil, robótica paralela, robótica híbrida y teleoperación" durante la realización de este trabajo.

6. REFERENCIAS

Libro: [1] W. F. Ableson, S. Robi y C. King, Android guía para desarrolladores, 2nd ed. España: Anaya Multimedia, 2011.

Resúmenes de Congresos: [2] V. J. G. Villela, A wheeled mobile robot with obstacle avoidance capability, *SOMIM*, vol. 1, n° 5, pp. 159-166, 2004.

Libro: [3] P. Correia, Guía práctica del GPS, España: Marcombo, 2002.

Sitio Web: [4] National Geographic Education Programs. Available at: http://education.nationalgeographic.com/education/media/selecting-map-projection/?ar_a=1. (Accesed on: Mar 7, 2013).

Resúmenes de Congresos: [5] A. Klimaszewski-Patterson, SMARTPHONES IN THE FIELD: PRELIMINARY STUDY COMPARING GPS CAPABILITIES BETWEEN A SMARTPHONE AND DEDICATED GPS DEVICE, *Papers of the Applied Geography Conferences*, vol. 33, pp. 270-279, 2010.

Sitio Web: [6] T. S. P. & C. Gorman, IEE SPECTRUM. Available: http://spectrum.ieee.org/video/computing/software/moveas-data-fusion-transforms-sensors-into-indoor-navigation/?utm_source=computerwise&utm_medium=email&utm_campaign=040313. (Accesed on: April 03, 2013).

Sitio Web: [7] Google, Android Developers. Available: <http://developer.android.com/guide/components/index.html>. (Accesed on: Mar 25, 2013).

Bibliografía

- [1] R. Fenton., «R.E. Fenton. IVHS/AHS: Driving into the future. Control Systems,» *IEEE*, vol. 14, nº 6, pp. 13-20, 1994.
- [2] A. M. Khamis, Interacción Remota con Robots Móviles Basada en Internet, Madrid (España): "Tesis Doctoral", Universidad Carlos III de Madrid, 2033.
- [3] M. F. Villa, «Vehículo robótico: autónomo y teleoperador con una PDA,» *SOMIM*, vol. 3, nº 2, pp. 46-54, 2009.
- [4] T. Menard, «Comparing the GPS Capabilities of the Samsung Galaxy S, Motorola Droid X, and the Apple iPhone for Vehicle Tracking Using FreeSim_Mobile,» de *International IEEE Conference on Intelligent Transportation Systems*, Washington, DC, USA, 2011.
- [5] V. J. G. Villela, «A wheeled mobile robot with obstacle avoidance capability,» *SOMIM*, vol. 1, nº 5, pp. 159-166, 2004.
- [6] D.Sivaraj, «Development of Navigation System for Autonomous Vehicle,» de *International Conference on Machine Vision and Image Processing, MVIP*, Coimbatore, Tamil Nadu, India, 2012.
- [7] A. B. S. Saman, «Implementation of autonomous vehicle navigation algorithms using event-driven programming,» de *4th International Conference on Intelligent and Advanced Systems, ICIAS*, Kuala Lumpur, Malaysia, 2012.
- [8] P. Correia, Guía práctica del GPS, Barcelona (España): Marcombo, 2002.
- [9] J. B.-Y. Tsui, Fundamentals of Global Positioning System Receivers A Software Approach, Canada: Wiley-Interscience, 2005.
- [10] R. B. Langley, «The Orbits of GPS Satellites,» *GPS World*, vol. 2, nº 3, pp. 50-53, 1991.
- [11] M. Shaw, «Modernization of the Global Positioning System,» *GPS World*, vol. 11, nº 9, pp. 36-44, 2004.
- [12] S. M. GMV, «El impacto de la ionósfera en la navegación por satélite,» de *Jornada Técnica sobre Clima Espacial II*, Madrid, 2012.
- [13] Y. Georgiadou, «The Issue of Selective Availability,» *GPS World*, vol. 1, nº 5, pp. 53-56, 1990.
- [14] A. M. Furones y G. Departamento de Ingeniería Cartográfica, 2010. [En línea]. Available: <http://www.upv.es/unigeo/index/docencia/etsigct/astronomía/teoria/astronomía210.pdf>.

[Último acceso: 16 03 2012].

- [15] S. P. O'Connor, «National Geographic Education Programs,» 2012. [En línea]. Available: http://education.nationalgeographic.com/education/media/selecting-map-projection/?ar_a=1. [Último acceso: 7 3 2013].
- [16] O. D. Hernández, Experimentación con una configuración de robot móvil de llantas frontales independientes es su direccionabilidad, Tesis Maestría, México D.f: UNAM, 2010.
- [17] L. I. G. Calandín, Modelado Cinemático y Control de Robots Móviles con Ruedas Tesis Doctoral, Valencia, España: Universidad Politécnica de Valencia, 2006.
- [18] B. Siciliano, Robotics Modelling, Planning and Control, Sprienger, 2010.
- [19] «NASA's Smartphone Nanosatellite,» [En línea]. Available: http://www.nasa.gov/directorates/spacetech/small_spacecraft/phonesat.html. [Último acceso: 27 04 2013].
- [20] W. F. Ableson, S. Robi y C. king, Android guía para desarrolladores, España: Anaya Multimedia, 2011.
- [21] «International Data Corporation (IDC),» [En línea]. Available: <http://www.idc.com/getdoc.jsp?containerId=prUS23946013>. [Último acceso: 10 04 2013].
- [22] A. Klimaszewski-Patterson, «SMARTPHONES IN THE FIELD: PRELIMINARY STUDY COMPARING GPS CAPABILITIES BETWEEN A SMARTPHONE AND DEDICATED GPS DEVICE,» *Papers of the Applied Geography Conferences*, vol. 33, pp. 270-279, 2010.
- [23] Google, «Android Developers,» [En línea]. Available: <http://developer.android.com/guide/components/index.html>.
- [24] T. S. P. & C. Gorman, «IEEE SPECTRUM,» 1 04 2013. [En línea]. Available: http://spectrum.ieee.org/video/computing/software/moveas-data-fusion-transforms-sensors-into-indoor-navigation/?utm_source=computerwise&utm_medium=email&utm_campaign=040313. [Último acceso: 3 04 2013].
- [25] E. F. Clavijo, C. Torrent Cuevas y R. Martí Mateu. [En línea]. Available: <http://www.coopvvgg.com.ar/alumnado-gomara/Files/gsm.pdf>. [Último acceso: 30 05 2013].
- [26] Barbara Maranzani. [En línea]. Available: <http://www.history.com/news/bertha-benz-hits-the-road-125-years-ago> [Último acceso: 10 06 2013]