UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TEMA:

Implementación de dos robots autónomos y uno controlado mediante tecnología Bluetooth para las categorías seguidor de línea con obstáculos, laberinto y balancín.

AUTORES:

Caiza García, Víctor Daniel

Verdezoto Cedeño, Cristhian Enrique

Trabajo de Titulación previo a la obtención del grado de

INGENIERO EN TELECOMUNICACIONES

TUTOR:

Palacios Meléndez, Edwin Fernando

Guayaquil, Ecuador

13 de septiembre del 2016

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

CERTIFICACIÓN

Certificamos que el presente trabajo fue realizado en su totalidad por los Sres. **Caiza García, Víctor Daniel** y **Verdezoto Cedeño, Cristhian Enrique** como requerimiento para la obtención del título de **INGENIERO EN TELECOMUNICACIONES**.

TUTOR

_____

Palacios Meléndez, Edwin Fernando

DIRECTOR DE CARRERA

_____

Heras Sánchez, Miguel Armando

Guayaquil, a los 13 del mes de septiembre del año 2016

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

Nosotros, **Caiza García, Víctor Daniel** y **Verdezoto Cedeño, Cristhian Enrique**

DECLARAMOS QUE:

El trabajo de titulación "**Implementación de dos robots autónomos y uno controlado mediante tecnología Bluetooth para las categorías seguidor de línea con obstáculos, laberinto y balancín.**" previo a la obtención del Título de **Ingeniero en Telecomunicaciones**, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de nuestra total autoría.

En virtud de esta declaración, nos responsabilizamos del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, a los 13 del mes de septiembre del año 2016

LOS AUTORES

_____	_____
CAIZA GARCÍA, VÍCTOR DANIEL	VERDEZOTO CEDEÑO, CRISTHIAN

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

AUTORIZACIÓN

Nosotros, **Caiza García, Víctor Daniel** y **Verdezoto Cedeño, Cristhian Enrique**

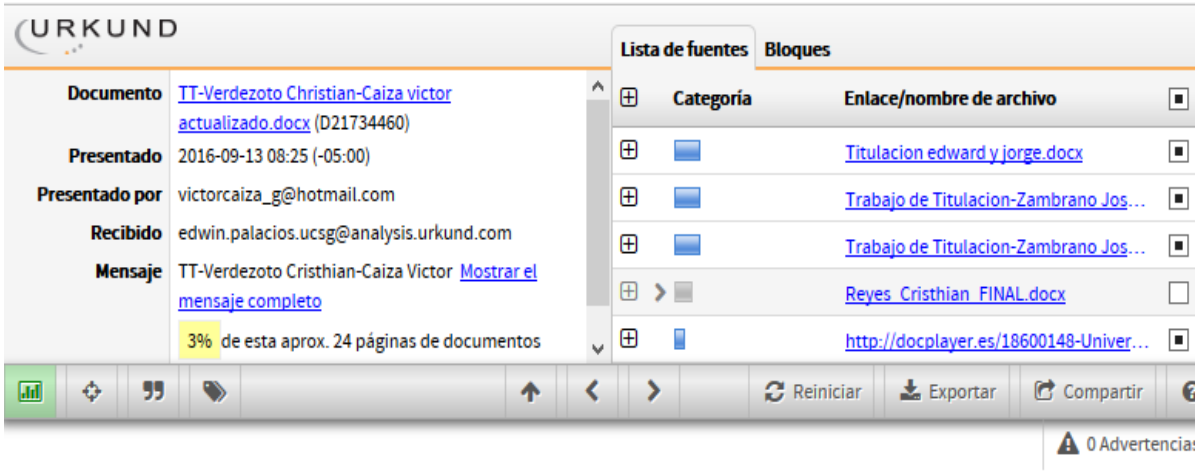
Autorizamos a la Universidad Católica de Santiago de Guayaquil, la publicación, en la biblioteca de la institución del Trabajo de Titulación: "**Implementación de dos robots autónomos y uno controlado mediante tecnología Bluetooth para las categorías seguidor de línea con obstáculos, laberinto y balancín**", cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

Guayaquil, a los 13 del mes de septiembre del año 2016

LOS AUTORES

_____	_____
CAIZA GARCÍA, VÍCTOR DANIEL	VERDEZOTO CEDEÑO, CRISTHIAN

REPORTE DE URKUND



UNIVERSIDAD CATÓLICA DE SANTIAGO DE GUAYAQUIL FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO CARRERA

DE INGENIERÍA

EN TELECOMUNICACIONES

TEMA:

Implementación de dos robots autónomos y uno controlado mediante tecnología Bluetooth para las categorías seguidor de línea con obstáculos, laberinto y balancín. AUTORES: Cristhian Enrique Verdezoto Cedeño Víctor Daniel Caiza García

Trabajo de Titulación previo a

la obtención del grado de INGENIERO EN TELECOMUNICACIONES TUTOR: Fernando, Palacios

DEDICATORIA

A Dios por brindarme la oportunidad de poder llegar a estas instancias. A mi familia por apoyarme siempre desde el inicio de la carrera hasta el final de la misma, siendo ellos los partícipes de cada etapa de mi carrera.

A mis profesores que aportaron con su paciencia y sabiduría. Fueron determinantes en cada proceso de nuestra carrera universitaria y a lo largo del trabajo de titulación.

Cristhian Verdezoto Cedeño

A Dios por ser el patrocinador principal de este proyecto de vida, siendo él la razón del mismo. A mi familia por su consejo y apoyo en cada proceso de esta etapa.

Finalmente, a mis maestros por brindarme parte de sus conocimientos, los cuales se proyectan como resultado en este trabajo académico y personalmente.

Víctor Caiza García

AGRADECIMIENTO

Primero a Dios, por permitirme llegar a culminar con éxitos esta etapa de mi vida.

A mi padre por su constante apoyo, a mi hermana, a Vicky porque ellos como familia fueron los que siempre me incentivaron a seguir adelante en mi camino.

A mis profesores por su enseñanza dentro y fueras de las aulas que contribuyeron para que alcance esta instancia de mi vida.

Cristhian Verdezoto Cedeño

A Dios quien me ofrece cada día la oportunidad de ser mejor.

A mi padre por su respaldo y consejo a cada momento, a mi madre por educarme con sabiduría y a mis hermanos siendo ellos los partícipes principales de mi vida.

A mi familia espiritual por estar conmigo en todo momento.

Víctor Caiza García

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TRIBUNAL DE SUSTENTACIÓN

f. _____

EDWIN FERNANDO PALACIOS MELÉNDEZ

TUTOR

f. _____

MIGUEL ARMANDO HERAS SÁNCHEZ

DIRECTOR DE CARRERA

f. _____

NESTOR ARMANDO ZAMORA CEDEÑO

COORDINADOR DE ÁREA

Índice General

Índice de Figuras

Capítulo 2

Capítulo 3

RESUMEN

Este trabajo de titulación tiene como meta centralizar el estudio de la tecnología robótica móvil, motivando al lector en el área de investigación. Indagar en ámbitos tecnológicos tanto de la parte física como programable (software y hardware) para la implementación de un robot en categoría seguidor de línea con obstáculos, laberinto y balancín. Presentamos a los microcontroladores y su impacto desde años atrás, estándares y procesos que se deben tener en cuenta para la implementación de un robot y como ha sido su impacto en la sociedad.

En la fundamentación teórica se observa la información que nos llevara a tener de una manera más clara y concisa que engloba los microcontroladores, memorias, tipos de microcontroladores, tecnologías inalámbricas. Se resalta la tecnología autónoma y de bluetooth necesarias para nuestros robots; revisamos de manera general, robots dentro de estas categorías que han sido importantes para la investigación en el mundo.

En este trabajo detallamos a cabalidad todos los elementos sobresalientes requeridos para la implementación de los robots presentados, su programación y diseño como tal de las tres categorías.

Palabras claves: MICROCONTROLADORES, BLUETOOTH, RADIO CONTROL, ROBÓTICA MÓVIL, AUTÓNOMO, ARDUINO.

ABSTRACT

The goal of this final Project is to centralize the exploring of technology of mobile robotics, encouraging the reader at the investigation area. Inquire in technological areas such as physical part like programmable (software and hardware) for the robot implementation at category line follower with obstacles, maze, balancing. We present the microcontrollers and the impact that they caused years ago, standards and processes that are important for the robot implementation and how has been its impact in our society.

At the theoretical part we observe the information that will take us to have it clearer and include the microcontrollers, memory, microcontroller's types, wireless technologies. It highlights the autonomous technology and bluetooth technology necessary for our robots; we check in a general way, robots within of these categories that are been important for world investigation.

In this Project, we detail very well all the outstanding elements required for the robot implementation presented, their design and programming such as all the three categories.

Key words: MICROCONTROLLERS, BLUETOOTH, RADIO CONTROLLER, MOBILE ROBOTIC, AUTONOMOUS, ARDUINO.

CAPÍTULO 1: INTRODUCCIÓN

1.1. Introducción.

En este capítulo vamos a conocer un breve preámbulo sobre los 3 robots: seguidor de línea con obstáculos, laberinto y balancín. Tanto el seguidor de línea como el de laberinto operarán de forma independiente, mientras que el balancín trabajará mediante tecnología bluetooth.

En un mundo de constante evolución, la tecnología crece a pasos agigantados, y dentro de ella la robótica es esencial para su desarrollo en todo ámbito; no obstante, el afán del ser humano por satisfacer sus necesidades y sobre todo hacer su vida mucho más fácil, ha sabido sobrellevar este proceso tan acelerado, al mismo ritmo. Es por ello, la necesidad de incorporar novedosas tecnologías a los laboratorios de la Facultad Técnica para el Desarrollo de la Universidad Católica de Santiago de Guayaquil, específicamente para los estudiantes de la Carrera de Ingeniería en Telecomunicaciones.

La construcción de robots permite a los estudiantes de nuestra facultad participar en concursos de robótica ya sea a nivel nacional, como internacional consiguiendo logros importantes en los diferentes ámbitos en los que se requiera participar, que dejan en lo alto el nombre de nuestras carreras, y de la UCSG. El objetivo principal de los robots autónomos y el controlado por tecnología bluetooth es vencer a los demás participantes. Las opciones de mando que controlarán a los robots se realizarán utilizando tecnología autónoma y bluetooth respectivamente.

En este proceso se analiza y recopila la información necesaria sobre la tecnología autónoma y bluetooth. En esencia estos elementos contendrán dispositivos electrónicos acoplados a sus cuerpos mecánicos; así estas

máquinas necesitan de apropiados medios sensoriales para resistir el entorno al que se coloquen, una sofisticada forma mecánica con propiedades, como cierta habilidad física de traslación y manejo, y de mecanismos de control que sean capaces de llevar acciones correctivas cuando sea el caso.

En esta forma, el trabajo sobre el robot seguidor de líneas con obstáculos se basará en desarrollar mecanismos propios de la tecnología autómata para que nuestro robot sea capaz de recorrer un camino señalado, con su principal característica de esquivar obstáculos en el entorno en que se encuentre. La función del laberinto será similar a la del seguidor, este será capaz de desplazarse a lo largo de un laberinto, propiamente dicho, formado por paredes. Con esto se conseguirá satisfacer los principios básicos de este campo multidisciplinario como lo es la tecnología autómata. Por otro lado, el robot balancín, controlado por medio de un dispositivo móvil mediante bluetooth logrará mantenerse equilibrado.

Cabe recalcar que la gradual necesidad de trabajos de investigación de las instituciones de tercer nivel, dentro de sus facultades de tecnologías, sobre los proyectos de diseño e implementación de robots, siempre ha sido de interés, pero esta investigación presenta un peculiar reto al estudiante, como es el presentar una información clara, y en español para que, nuestros compañeros posteriores se valgan de la misma y mejorarla en lugar de recurrir a libros que, generalmente están en otro idioma, o que simplemente no satisfacen lo que se busca.

La implementación de dos robots autónomos y uno controlado por tecnología bluetooth (seguidor de línea con obstáculos, laberinto y balancín) facilitará el proceso de enseñanza de fundamentos de robótica en las diferentes carreras de nuestra facultad y beneficiará no solo a los estudiantes de los

primeros ciclos, sino que servirá de pauta para futuros proyectos de investigación, en las tres carreras técnicas.

1.2. Antecedentes.

Desde el estudio de la robótica se ha logrado alcanzar campos que tiempos atrás eran inaccesibles, pues hoy en día lo palpamos en nuestro entorno, desde un recolector de basura hasta un dron capaz de captar imágenes en tiempo real. La robótica cubre numerosas áreas ya sea medicina (operaciones vías online, brazo robótico, órganos artificiales, etc.), informática, ayudando al mantenimiento de la industria tecnológica. Y sobre todo en las famosas batallas de robots que son muy frecuentas a nivel universitario y profesional.

El estudio de la robótica ha influido mucho para poder llevar a cabo este proceso, por ello se han implementado diversos planes y proyectos en universidades con carreras de tecnología cuyo objetivo es continuar evolucionando el nivel académico.

De esta manera la Universidad Católica de Santiago de Guayaquil está optando por involucrarse a los diversos concursos de robótica existentes a nivel universitario por medio de la introducción de un club de robótica orientado al avance de dichas investigaciones.

1.3. Justificación del Problema.

Actualmente, a pesar de que nuestra facultad ya posee un club de robótica dedicado al avance del mismo, éste no es suficiente para satisfacer las necesidades del estudiantado. Es así que en los concursos realizados por otras universidades, nuestra facultad como tal no consigue participar en todas las categorías en comparación con otras universidades que han dedicado recursos para alcanzar los primeros lugares.

Razón por la cual se está desarrollando múltiples investigaciones y si se diera el caso diseños relacionados al campo de la robótica. Dentro de este proceso nuestra universidad ha conseguido grandes avances respecto a la integración de un grupo de estudiantes interesados en el diseño e implementación de robots para diversas categorías.

1.4. Definición del problema.

Con el fin de una aportación didáctica en la Facultad de Educación técnica para el desarrollo, se implementará la construcción de dos robots autónomos y uno por medio de tecnología bluetooth, cuya finalidad es participar en las categorías que en ediciones pasadas no se disponían por falta de recursos. Adicional a esto este material proporcionará información importante para compañeros de futuras promociones.

1.5. Objetivos del Problema de Investigación.

1.5.1. Objetivo General.

Implementar dos robots autónomos y uno controlado mediante tecnología bluetooth para las categorías seguidor de línea con obstáculos, laberinto y balancín.

1.5.2. Objetivos Específicos.

- Describir el estado del arte de los diferentes dispositivos electrónicos que son utilizados en el ámbito de la robótica y del medio de transmisión inalámbrico que requiere el móvil no autónomo.
- Realizar el diseño electrónico de los robots móviles tanto autónomo como el controlado por la tecnología bluetooth.
- Diseñar los algoritmos de programación que nos ayudará a establecer la autonomía del robot móvil y la comunicación bluetooth con el robot no autónomo.

1.6. Metodología de Investigación

En el presente trabajo de titulación, se describe los tipos de metodología investigativa que se usará a lo largo de su desarrollo. La investigación descriptiva de acuerdo a (Ávila B., 2016) nos permitirá recolectar la información suficiente para analizarla detalladamente, partiendo de una manera generalizada convirtiéndola en ideas muy específicas, es decir, se buscará de esta manera cumplir con los objetivos trazados.

La investigación explicativa según (Hernández, Fernández, & Baptista, 2006) está orientada a la búsqueda de soluciones de variables que se puedan presentar en el desarrollo del problema. Basándose en el estudio de varias características que contribuirá a una investigación eficiente.

La investigación exploratoria de acuerdo a (Nieves C., 2016) se enfoca generalmente a la elaboración de conclusiones, si quedó alguna duda posterior al estudio realizado. Genera un gran impacto en nuestra investigación ya que nos ayuda a tener de una manera mucho más clara las incógnitas que se quieren despejar. Por ello aportará al desenvolvimiento a una investigación más sobresaliente.

CAPÍTULO 2: FUNDAMENTOS TEÓRICOS DE LA IMPLEMENTACIÓN DE ROBOTS

2.1. Sistemas Microcontroladores

Imaginemos una computadora con su sistema de mandos, hardware y software obedeciendo los comandos que el usuario requiere, y es que, al hablar de ésta, claramente estamos tomando un ejemplo clásico de microprocesadores. Hoy en día las necesidades del ser humano en su búsqueda por hacer la vida más sencilla, dio lugar a una mejora de estos sistemas digitales. Es aquí donde nace el microcontrolador, que no es más que un conjunto de elementos dentro de un circuito integrado, como se representa en la Figura 2. 1 a continuación.

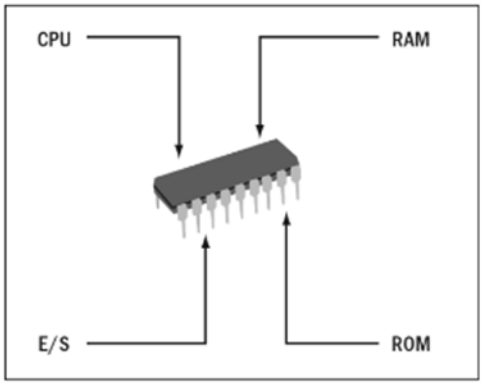


Figura 2. 1: Un microcontrolador es un conjunto de elementos dentro de un circuito integrado.
Fuente: (Rossano, 2009)

Los microcontroladores tienen como función principal, administrar sus labores. Además, sus memorias están dentro de un mismo circuito. Económicamente hablando los microcontroladores son mucho más accesibles que un microprocesador. Presenta bajos niveles de interferencia debido a su arquitectura. Consecuentemente el momento de su despliegue será mucho más

rápido. Sin mencionar que el ahorro de energía es mucho más accesible. (Molina C., 2016)

Para hacer uso del microcontrolador se debe tener en cuenta conocimientos sobre lenguajes de programación avanzado. Es así que estos lenguajes resultan ser herramientas útiles aliadas para el usuario. Sus usos, múltiples, pues los encontramos tanto en nuestros hogares desde nuestra computadora de escritorio, los electrodomésticos hasta los frenos (Antilock Brake system) de un automóvil. (Ayovi & Mero, 2011)

En la Figura 2.2 se ilustra el esquema típico de un microcontrolador.

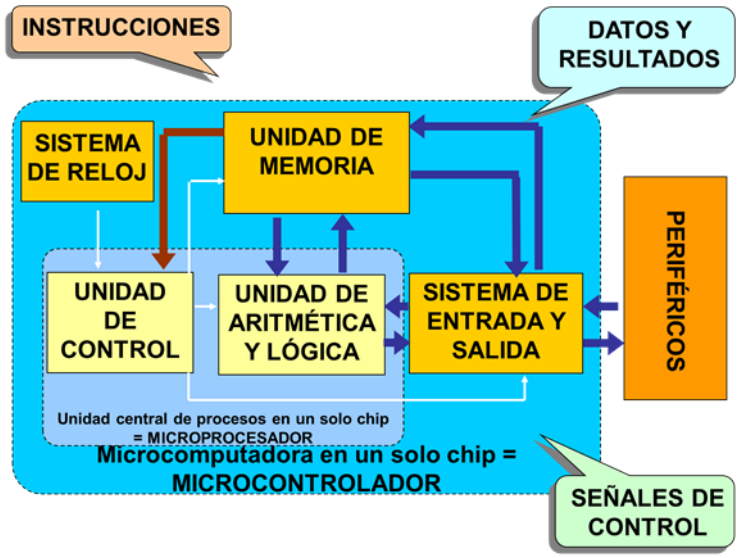


Figura 2. 2: Esquema de un microcontrolador.
Fuente: (Rodríguez, 2016)

En un sistema microcontrolador es importante hablar sobre la memoria que se la puede dividir en dos grupos: la externa y la interna. En la memoria interna se almacenan los datos que no se pierden al desconectar la fuente; en la memoria externa pasa lo contrario, pues se almacenan los datos que si pueden ser borrados al conectarse a la fuente.

2.1.1. Tipos de Memorias

Memoria RAM

Random Access Memory, alcanza una media de 256 bytes, esto puede variar dependiendo el dispositivo, es considerada una memoria volátil por su propiedad de perdida de datos cuando se la desconecta de la fuente.

Memoria ROM

Read Only Memory, debido a que es programada en su fabricación, la información que guarda es constante y no se puede alterar.

Memoria EPROM

Erasable Programmable Read Only Memory, su función es muy similar a la de la memoria ROM excepto porque esta puede ser programable con el dispositivo adecuado. (Gonzalez, 2013)

Memoria EEPROM

Electrically Erasable Programmable Read Only Memory, este tipo de memoria es más amigable con el usuario grabar y programar datos. Debido a sus propiedades complejas EEPROM tiende a ser lenta normalmente. (Gonzalez, 2013)

Memoria EEPROM Flash

Es una versión mejorada de EEPROM ya que a más de poder grabar y programar datos lo hace con una velocidad alta; también nos posee la capacidad de reprogramar datos. (Gonzalez, 2013)

2.1.2. Arquitecturas de los microcontroladores

Normalmente se manejan dos tipos: Arquitectura Von Neumann y Arquitectura Harvard.

Arquitectura Von Neumann

Al comienzo era una arquitectura muy utilizada en los sistemas computacionales, pero a medida que va avanzando la tecnología se vio desplazada por otra arquitectura. Sostuvo su propuesta de que el lenguaje binario reduciría considerablemente posibles fallos en las operaciones de los sistemas. Y guarda en su memoria las instrucciones que han sido dadas anteriormente. (Trujillo, C.; Candelo, A.; Zipaquira, J., 2014)

Arquitectura Harvard

Este tipo de arquitectura generalmente la encontramos en los PIC, tiene por separado los datos de las instrucciones, pero cada una conectada a un sistema de buses para tener fácil acceso a las diferentes operaciones que se tengan que realizar al mismo tiempo. (Condor & Paredes, 2009)

2.2. Microcontroladores PIC

El microcontrolador PIC pertenece a una gran gama de modelos fabricadas principalmente por la empresa **MICROCHIP TECHNOLOGY INC.** que cubre gran parte del mercado. Cuando hablamos de microcontroladores PIC debemos mencionar características generales como: mayor velocidad, su valor es accesible, es amigable con el usuario a la hora de programar, y su consumo de potencia es mínima. (Reyes, 2008)

2.2.1. Características generales de un microcontrolador PIC

Como vimos, gran parte de los microcontroladores fueron diseñados con una memoria con capacidad de desarrollo mucho mayor a la de datos “La memoria de programa está organizada en palabras de 12, 14 o 16 bits mientras que la memoria de datos está compuesta por registros de 8 bits” (Valdéz & Pallas, 2007).

Adicionalmente en los microcontroladores PIC poseen una pila con una profundidad limitada dependiendo del modelo del PIC. Esta no tiene relación en cuanto a la ubicación con la memoria de datos (Valdéz & Pallas, 2007). Todos los microcontroladores PIC poseen diversas categorías, divididas en gama baja y gama alta. En los microcontroladores de gama baja no sufren ningún tipo de interrupción. Mientras que en los de gama alta tienen acceso a los registros de la memoria de datos, en esta memoria de datos se encuentran los bits que pueden ser requeridos para seguir alguna instrucción asignada. (Valdéz & Pallas, 2007)

Es importante resaltar que los microcontroladores PIC también son llamados microcontroladores RISC y que todas sus instrucciones son del mismo tamaño con palabras que oscilan entre 12 a 16 bits. Además, constan de dos tipos de registros: el primero corresponde al registro de trabajo y el segundo al registro de memoria, según el programador. Absolutamente todos los microcontroladores PIC cuentan con un dispositivo de alarma cuya función es proteger al microcontrolador de cualquier intento de reproducción ilícito. Existen dos gamas de microcontroladores PIC: de gama alta y gama baja (Valdéz & Pallas, 2007).

“Los microcontroladores PIC disponen de las siguientes opciones para el oscilador principal: oscilador de cristal (de cuarzo), oscilador RC y oscilador externo. Algunos dispositivos disponen de un oscilador RC interno de unos 4 MHZ” (Valdéz & Pallas, 2007). Al momento de configurar un microcontrolador debemos de tomar en cuenta el número de bits que posee, sin embargo, una vez que los bits sean programados ya no se podrán hacer cambios., los bits de configuración sirven para adaptar de una mejor manera las instrucciones que el usuario desee para su microcontrolador (Valdéz & Pallas, 2007).

2.2.2. Conceptos básicos

"La memoria es un conjunto de celdas o localizaciones que se identifican por su dirección. En cada celda se almacena una palabra. Una palabra es la unidad lógica de información almacenada en una celda de la memoria" (Valdéz & Pallas, 2007).

Las palabras pueden ser conformadas por 8, 12, 14, 16 bits. A continuación, ilustraremos lo mencionado en la Figura 2.3.

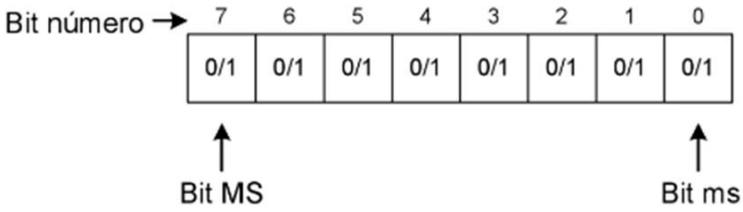


Figura 2. 3: Representación de un octeto (palabra de 8 bits). En donde se especifica los bits más significativos hasta los menos significativos.
Fuente: (Valdéz & Pallas, 2007)

Como se lo mencionaba anteriormente la memoria de los microcontroladores posee un papel fundamental y es que esta memoria debe ser organizada linealmente. La misma que es llamada también "páginas". (Valdéz & Pallas, 2007)

2.3. Microcontroladores Atmel AVR

Pertenecen a un nuevo grupo con configuraciones RISC, están compuestos tanto de memoria EEPROM para los datos, como memoria flash para el programa, fue fabricada con altos estándares de calidad para un arduo rendimiento. Fueron elaborados en base al lenguaje C y ensamblador.

Al utilizar un lenguaje de tan alto engranaje, nació la necesidad del diseño del AVR con 32 registros de 8 bits capaz de desempeñarse con eficacia respecto a la ejecución del lenguaje C. Nada más basta compararlos con microcontroladores PIC para darnos cuenta de la ventaja de estos dispositivos,

ya que la relación de los registros tanto de entrada como de salida y los datos trabajan con unanimidad formando así un espacio único de memoria.

Recordemos que un microcontrolador no es más que un conjunto de elementos dentro de un circuito integrado, de tal manera que pueda ser procesado y programado para desarrollar la información que el usuario requiera. Por eso la necesidad de resaltar los microcontroladores Atmel y la evolución de esta gran familia, con su facilidad para poder ejecutarse en distintos lenguajes utilizados en configuraciones complejas.

Su continuo uso en la vida cotidiana es debido a su ahorro de energía, a que puede interactuar con otros dispositivos y económicamente hablando es de muy fácil acceso. También son sencillamente programables.

2.3.1. Memoria de programación AVR

“Cada programa que se desarrolla para los AVR, se almacena en una región de la memoria no volátil (es decir, permanece al pagar el dispositivo), además es programable con el procedimiento de carga (flash). La primera sección de esta región es la sección de carga (flash) de la aplicación y es donde se almacena el programa que se escribe para el AVR. La segunda sección se llama: ‘Boot Flash Section´, o sección de carga del inicio y se puede configurar para que funcione una vez que el dispositivo (sistema ensamblado), se prende o se enciende.” (Salguero, 2009)

A continuación, la Figura 2.4 muestra las secciones que forman parte de la memoria de los microcontroladores.

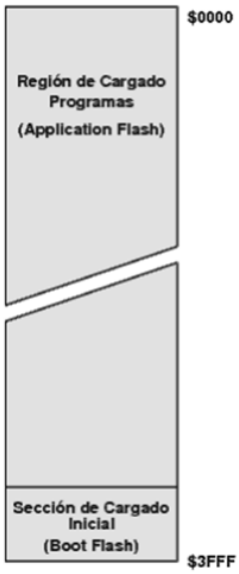


Figura 2. 4: Representación de las secciones de cargado.
Fuente: (Salguero, 2009)

Cada código programable que se inserte, se plasme o se emplea debe ser ajustado al código hexadecimal que no es más que un conjunto de números hexadecimales que al ser introducidos para las operaciones con microcontroladores, el programador ya no deberá preocuparse por los datos en la hoja del fabricante. Todo esto gracias a la propiedad de operar con lenguajes de alto nivel, uno de ellos Lenguaje C. Procederemos a nombrar algunos de los microcontroladores AVR más sobresalientes en el medio.

- AVR UC3 de 32 bit.
- AVR XMEGA MCU.
- MEGA AVR MCU.
- TinyAVR MCU de 8 bit.

2.3.2. AVR UC3 de 32 bit

Además de su eficiente comportamiento y ahorro de energía, consta de un soporte de doble puerto, numerosos buses de datos. Algunos elementos UC3

tienen integradas las unidades FPU (unidad de punto flotante), que ajusta la precisión y un mejor dinamismo en el rango.

2.3.3. AVR XMEGA MCU

Poseen características específicas como lo son los cifrados DES y AES. Estos dispositivos cuentan con la capacidad de poder desarrollar funciones lógicas que hoy en día son muy usadas. (Vera & Alejandro, 2016)

2.3.4. MEGA AVR MCU

Sirve como un impulsador, cuando vemos que los diseños necesitan de una ayuda extra nos referimos a estos microcontroladores. A la hora de hablar de un extenso número de códigos; con una capacidad de hasta 20 MIPS. Sus actualizaciones son confiables y no solo eso sino que su memoria pueden ser actualizados mientras se esté ejecutando alguna operación. (Vera & Alejandro, 2016)

2.3.5. TinyAVR MCU de 8 bit.

Permite su configuración en cualquier lenguaje ensamblador. Podría decirse que gracias a su alto rendimiento analógico TinyAVR MCU de 8 bit. es el más compacto de la familia de los microcontroladores AVR pudiendo operar con 0.7 v y es altamente compatible con los dispositivos pertenecientes al mismo grupo.

2.4. Microcontrolador ATMEGA 164P.

Este dispositivo pertenece a la familia de los microcontroladores Atmega. A continuación, en la Figura 2. 5 mostraremos las características más importantes de este microcontrolador:

Frecuencia máxima	20 MHz
Rango de voltaje	2.7 - 5.5 V
Corriente máxima entre sus pines	40 mA
Memoria de programa (flash)	16 KB
Posiciones RAM de datos	1 Kbytes
Posiciones EEPROM de datos	512 bytes
Puertos E/S	4
Timers	3
Comunicación serial	2 USART
Convertidor A/D	10 bits
Arquitectura	Harvard
Instrucciones de tipo RISC	131

Figura 2. 5: Características del microcontrolador Atmega 164p.
Fuente: (Ante & Espinel , 2014)

2.4.1. Microcontroladores ATMEGA 168

Este microcontrolador AVR está dentro del grupo de dispositivos con propiedades no volátiles, en su hoja de fabricante se detallan 131 instrucciones. Tiene una sección de código opcional.

Cuenta con 512 bytes de memoria EEPROM capaz de procesar 100.000 períodos de escritura y/o borrado. Tiene la capacidad de bloqueo, que actúa como cerradura y sirve para tener una mejor seguridad del software que se esté utilizando. Tiene actualizaciones muy accesibles con alta potencia y mínimo consumo de energía. (Salazar, 2014)

2.4.2. Características de los periféricos

Normalmente son elementos auxiliares que le permiten completar varias funciones al microcontrolador, detallaremos los comúnmente usados.

- Temporizadores
- Fuente de Alimentación
- Conversores

- Puertos de E/S
- Protección Brownout
- Puertas de Comunicación

2.4.3. Temporizadores

Tiene el control de los periodos de tiempo, así como también lo que ocurre afuera de él. Cuando este ha sobrepasado el límite de su capacidad el usuario tiene la facilidad de extraer la información cargada hasta el momento mientras se solucionan los posibles inconvenientes.

2.4.4. Fuente de Alimentación

Todos los tipos de los microcontroladores son capaces de operar con valores de voltaje que oscila entre 2.5V y 6V, esto dependerá de la hoja del fabricante que posee cada dispositivo de la cual se podrá valer el usuario. Y así este podrá verificar la capacidad de tensión del microcontrolador.

2.4.5. Conversores

Se pueden diferenciar dos clases de conversores, analógico - digitales y digitales - analógicos. Hay un gran número de microcontroladores que utilizan este tipo de conversores, ya que este les permite procesar la información analógica entrante que se encuentra en diversas aplicaciones mediante un multiplexor, estos son los analógico – digitales.

Por otra parte, los digitales - analógicos se encargan de la conversión de la información entrante con señal analógica. A través de otros dispositivos que trabajan con señales analógicas. (Vera & Alejandro, 2016)

2.4.6. Puertos de E/S

Todo microcontrolador posee cuatro puertos, los cuales están distribuidos de tal manera que dos de ellos reciben información, el otro par de puertos está compuesto de cristal de cuarzo el cual se encarga de estandarizar las frecuencias de operación; adicionalmente posee un terminal que servirá para la operación de reinicio. El resto de terminales servirán como medios de comunicación entre el entorno externo y el procesador a través de varios medios. La función característica de estos puertos es dar soporte a las señales de control que pueden atravesar estos puertos.

2.4.7. Protección "Brownout"

La función de este circuito es resetear al microcontrolador en caso de que este se encuentre sometido a un valor inferior de un voltaje mínimo.
Cuando este voltaje de alimentación se mantenga en ese valor, el microcontrolador mantendrá la función de reseteo. (Gordillo, 2011)

2.4.8. Puertas de Comunicación

Con el objetivo de mejorar al microcontrolador de manera que este pueda comunicarse con otros dispositivos electrónicos y poder asociarlo con protocolos y otros estándares, se ha introducido algunos elementos que hacen posible esta labor:

- USB

  Un bus serial utilizados en los computadores.
- UART

  Transmisor/receptor asíncrono universal
- USART

  Transmisor/receptor síncrono/asíncrono universal
- CAN

Un bus serial de los microcontroladores generalmente usados en aplicaciones de control.

- Puerta paralela capaz de vincularse con buses de otros microprocesadores.

2.5. Introducción a la Robótica

La robótica es un avance de la ingeniería basado en la arquitectura de un diseño e implementación de máquinas que cumplen una función específica ya sea doméstica o en el campo laboral según la necesidad del ser humano.

"Las ciencias y tecnologías de las que deriva podrían ser: el álgebra, los autómatas programables, las máquinas de estados, la mecánica, la electrónica y la informática. En los últimos tiempos la robótica ha jugado papeles muy importantes dentro del avance de la tecnología, como él envió de estos a misiones espaciales." (Sánchez & Peña , 2012)

La idea de la robótica se refiere a una arquitectura basada en sistemas digitales y mecánicos universales capaces de cumplir mandos similares a las del hombre. En la actualidad ha sabido adaptarse a esta tecnología ya que con el paso del tiempo nosotros lo observamos desde las fábricas hasta sitios de entretenimiento.

Cuando pensamos en robótica necesariamente se nos viene una idea a la cabeza sobre la definición de un robot, y es que hoy en día la palabra "robot" ya no nos parece tan descabellada como lo era en épocas pasadas debido a que el cine siempre se ha adelantado a la tecnología real. Pues basta con recordar películas donde visualizábamos este tipo de tecnología; nuestros padres, nuestros abuelos quizás veían este avance tecnológico muy lejano y ficticio. Por que quien imaginaría que robots como "TERMINATOR" "WALLE" entre otros iban

a terminar siendo hoy en día elementos que se exponen en las grandes ferias tecnológicas.

Hoy en día incluso se habla de robots enviados al espacio para realizar nuevos descubrimientos científicos, como lo es el robot conocido como “CURIOSITY”. Este es el encargado de recoger muestras durante años del suelo de otro planeta para así enviarlas a nuestro planeta, dejando así a los científicos para que realicen su respectivo análisis sobre la composición del mismo. Y al parecer la relación entre la máquina y el humano ha sido positiva, en función al trabajo en equipo ya que el robot satisface la necesidad del ser humano y a su vez este se vale del robot para hacer de su vida más sencilla.

2.5.1. La Robótica

Es Isaac Asimov quien en unas de sus obras menciona como tal el término “robot”. Es a él a quien se le atribuye este suceso y no es para menos es de su misma obra que el instituto Norteamericano de Robótica muestra el siguiente concepto: “Manipulador, multifuncional y reprogramable, diseñado para mover materiales, piezas, herramientas o dispositivos especiales, mediante movimientos programados y variables que permiten llevar a cabo diversas tareas”. (Pilaquingua, 2009)

Hoy por hoy el ser humano ha avanzado de manera considerable con las investigaciones, implementación y fabricación de robots. Lógicamente los robots necesitan de los mandos del ser humano para completar sus tareas asignadas, es así que no se ha logrado alcanzar el objetivo general, que es construir un robot totalmente inteligente, pero es a lo q se apunta. Mientras tanto el enfoque va dirigido a la fabricación de los robots antropomorfos, que participen en la fabricación de sí mismos. Actúan de manera tal que asisten al técnico en oficios

donde, propiamente dicho sería riesgoso y/o poco accesibles. A continuación, nombraremos las leyes de la robótica que (Pilaquingua, 2009) señala:

1) “Un robot no debe dañar a un ser humano o, por su inacción, dejar que un ser humano sufra daño.”
2) “Un robot debe obedecer las órdenes que le son dadas por un ser humano, excepto si estas órdenes entran en conflicto con la Primera Ley.”
3) “Un robot debe proteger su propia existencia, hasta donde esta protección no entre en conflicto con la Primera o la Segunda Ley.”

Adicional a las leyes citadas anteriormente, Asimov habla sobre una cuarta denominada “ley cero” que se refiere a que un robot no debe atentar contra la integridad humana o en su defecto dejar que la misma reciba algún tipo de daño. Estas leyes surgieron, asegura Asimov, por la gran incógnita de qué pudiera pasar cuando los robots tuvieran la capacidad de decisión propia, y pudieran de alguna manera no obedecer a sus creadores. Así estas leyes serian insertadas y programadas en el robot, y si ocurriere que estos no siguieran dichos principios la maquina se tornaría dañada.

2.5.2. Generaciones de Robots

(Vera & Alejandro, 2016) en su trabajo de titulación Diseño E Implementación De Dos Robots Seguidores De Línea Modalidad Velocista Y Destreza Para Participaciones En Concursos De Robótica, 2016, mencionan que a los robots en 3 grandes generaciones.

- 1ra generación

“Desde 1960 hasta 1980; comprende los robots industriales que realizan tareas repetitivas. Se desarrollan las funciones de manipulación fundamentalmente.”

- 2da generación

"Entre 1980 y 1985; incluye a los robots dotados de sensores que interactúan con el entorno. El robot puede modificar su accionar en tiempo real en función de los cambios en su entorno. Aquí se desarrollan funciones perceptivas y planificativas"

- 3ra generación

"Comprende la época desde 1985 hasta la actualidad; construcción de robots móviles lo que deriva en inteligencia artificial 56 aplicada a la robótica. Constantes mejoras en percepción y planificación." En la Figura 2. 6 se mostrará los robots de distintas generaciones.

Figura 2. 6: Robot de Primera Generación, Robot de Segunda Generación, Robot de Tercera Generación.
Fuente: (Vera & Alejandro, 2016)

Posteriormente menciona la naturaleza de un robot, ya la subdivide en tres sistemas mecánicos, los accionadores y los de control:

- Sistemas Mecánicos: Son toda la parte rígida del robot, incluye todo lo referente a extremidades, ganchos, ruedas, etc.
- Sistemas Accionadores: Se encargan de la función del movimiento, su propiedad determinada es brindar la movilidad a las extremidades o miembros que así lo requieran y varían dependiendo la clase de movimiento.

- Sistemas de control: Es aquel que ejecuta su función a partir de darse la orden, a raíz de esta el proceso se ejerce por sí solo, sin la intervención del usuario.

(Vera & Alejandro, 2016) Afirman que una vez examinado la arquitectura, así como también su funcionalidad, es posible dividir a los robots por la labor que realizan, o para lo que fueron diseñados:

- Domésticos: Dispositivos que comúnmente encontramos en la comodidad de nuestro hogar como, secadoras, aspiradoras y microondas.
- Móviles: Estos cumplen necesidades en la cual tengan que recorrer distancias programadas por el usuario.
- Industriales: A este tipo de robot se los reconoce como “industriales” ya que son usados en grandes sectores de manufactura que requieran de múltiples operaciones en los procesos de producción.

2.5.3. Robot Móvil

Podríamos definir al robot móvil como:

“Los robots móviles son dispositivos de transporte automático, es decir, una plataforma mecánica dotada de un sistema de locomoción capaz de navegar a través de un determinado ambiente de trabajo, dotado de cierto nivel de autonomía para su desplazamiento portando cargas. Sus aplicaciones pueden ser muy variadas y siempre están relacionadas con tareas que normalmente son riesgosas o nocivas para la salud humana, en áreas como la agricultura, en el transporte de cargas peligrosas o en tareas de exploración solitarias o cooperativas junto a otros vehículos no tripulados.” (Andaluz, 2011)

2.5.4. Datos generales

Podría decirse que el robot móvil surgió a partir de 1966 cuando el hombre el hombre se vio en necesidad de realizar estudios espaciales, iniciando así una

nueva era de robots móviles que podían cumplir misiones especificas en lugares obviamente inaccesibles por el hombre. Es así que en entre los años 60-70 se enviaron dos robots, cuya misión, era recolección de datos en lugares altamente nocivos. En la siguiente Figura 2. 7 se muestra los robots de investigaciones espaciales.

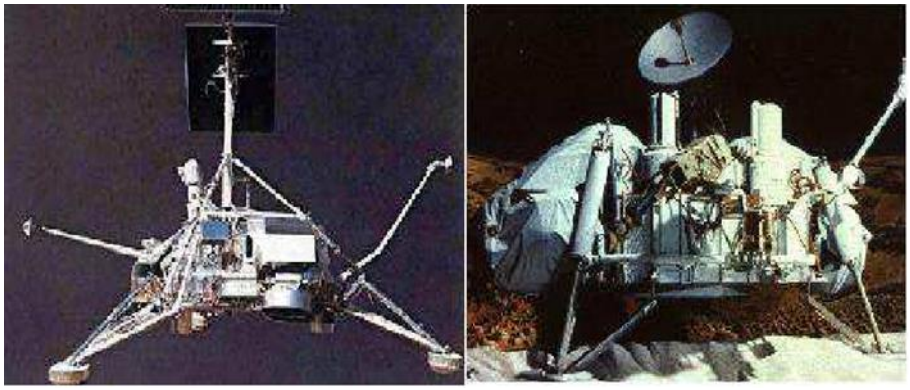


Figura 2. 7: Robots de investigaciones espaciales: Robot Surveyor y Robot Viking
Fuente: (Andaluz, 2011)

2.5.5. Tipos de Sistemas de Locomoción

(Andaluz, 2011) Mencionó que todo robot móvil se clasifica o se divide según la clase de locomoción por movimiento.

- Locomoción por ruedas
- Locomoción por patas
- Locomoción por orugas

Como podemos Observar que el desarrollo de los robots móviles, más se adapta a los de locomoción por ruedas que los otros. También existen los tipos de locomoción por el entorno en el que se desenvuelven:

- Flotantes
- Acuáticos
- Submarinos
- Aéreos

Es necesario mencionar que el robot móvil con ruedas es el más adquirido en el mercado puesto que, a diferencia de los demás, este se desenvuelve fácilmente en lugares remotos, su diseño es mucho más fácil, requiere menos materiales y por ende será de menor costo. (Andaluz, 2011)

2.5.6. Tipos de ruedas

Básicamente los robots de locomoción por ruedas se subdividen en los siguientes grupos:

- Rueda Fija
- Rueda Orientable Centrada
- Rueda Orientable no-centrada

2.5.7. Rueda Fija

El eje de la rueda está sujeto al esqueleto de robot; está directamente relacionada con la tracción del mismo (Andaluz, 2011). En la Figura 2. 8 tenemos una representación de lo mencionado anteriormente.

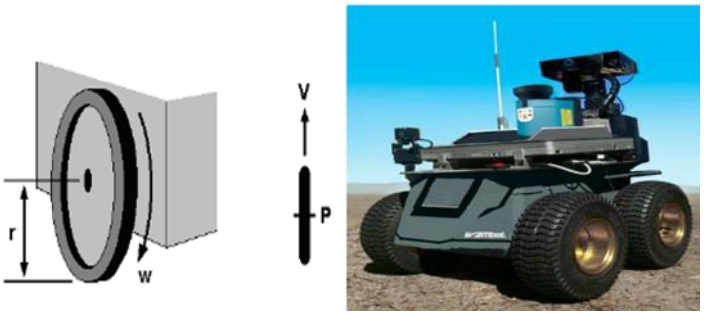


Figura 2. 8: Ejemplo de rueda fija.
Fuente: (Andaluz, 2011)

2.5.8. Rueda Orientable centrada

En este tipo se cumplen funciones como la de tracción y dirección (Andaluz, 2011). En la Figura 2. 9 expondremos un ejemplo de rueda orientable centrada.

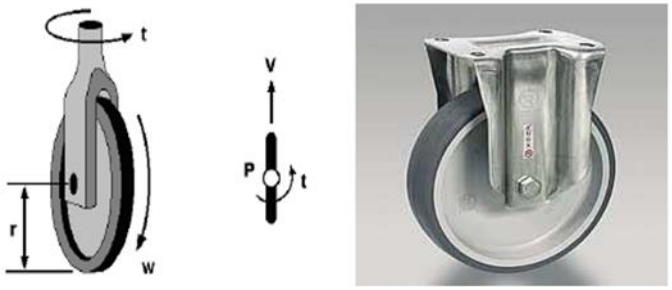


Figura 2. 9: Ejemplo de rueda orientable centrada.
Fuente: (Andaluz, 2011)

2.5.9. Rueda Orientable no centrada

Su funcionalidad principal es proporcionar firmeza a la estructura mecánica del robot. A continuación, en la Figura 2. 10 presentaremos un ejemplo de Rueda Orientable no centrada.

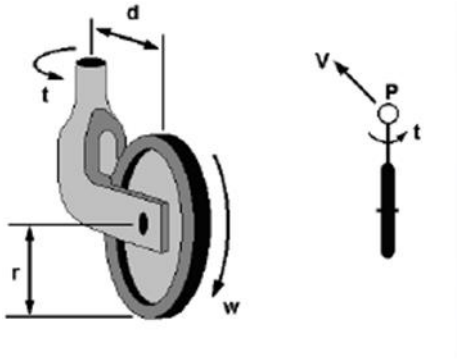


Figura 2. 10: Rueda Orientable no centrada.
Fuente: (Andaluz, 2011)

2.6. Robots Autómatas

La autonomía es la propiedad del robot para poder operar según sus órdenes, todo esto es gracias al programa que lleva dentro de sí mismo. No es necesario que el operador este al pie del robot para que este cumpla sus funciones. Cabe recalcar que un robot no es inteligente del todo, si así fuera el caso no necesitaría que lo programen para alcanzar sus objetivos,

probablemente esto pueda ser factible en un futuro gracias al constante avance de la tecnología. (Samaniego, 2009)

"Los campos de aplicación de los autómatas programables son hoy día extremadamente extensos debido a sus posibilidades en cuanto a flexibilidad en su programación, así como a la factibilidad de ampliación mediante módulos adicionales o incluso con más autónomas conectados en red, etc." (Lindao & Quilambaqui , 2014)

2.7. Niveles de Autonomía

A continuación, se muestran los distintos niveles de autonomía de un robot:

- Primer Nivel

A este se le denomina "Interacción directa" el robot es manejado hapticamente, ya que la persona posee el control casi total sobre las operaciones que vaya a ejecutar el robot, interviene en todas sus operaciones.

- Segundo Nivel

Son los tipos de ejecución que tienen que ver con comandos de un nivel intermedio respecto a la intervención del hombre; este se encarga solo de asistirlo, posteriormente el robot lo interpretara y lo ejecutara.

- Tercer Nivel

Podría decirse que son los menos humano-dependientes, ya que pueden operar relativamente solos, por un largo ciclo. No necesitan de patrones complejos, una maquina embazadora está totalmente autónoma y trabaja con un mismo comando.

Adicional a la información anterior existe otra clasificación, que tiene que ver con el manejo del hombre y los mandos de la máquina.

- Teleoperación: El operador, o sea el hombre maneja cada cambio de ritmo del robot, todos los movimientos los procesa el usuario.
- Supervisión: el ser humano previamente detalla mandos habituales o cambios de ritmo y el robot está en capacidad de decidir los movimientos de sus actuadores.

Básicamente el nivel de autonomía de un robot dependerá de la necesidad del ser humano para realizar un trabajo.

2.8. Fundamentos básicos de transmisión de las ondas

Toda transmisión necesariamente se realiza por ondas electromagnéticas, estas están formadas por ondas tanto eléctricas, como magnéticas, es en la fusión de estas dos, que se genera el tipo de onda que puede ser transmitida por el espacio a la velocidad de la luz. Las ondas electromagnéticas poseen 3 parámetros:

- Frecuencia: Define el número de ondas que se transmiten en un segundo.
- Velocidad: Es siempre la misma ya que es independiente de la frecuencia. Esta velocidad es igual a la velocidad de la luz (300.000 kilómetros por segundo).
- Longitud de onda: es el resultado de dividir la velocidad de propagación (la velocidad de la luz) por la frecuencia. El resultado viene expresado en metros."

2.9. Comunicaciones Inalámbricas

(Delgado, 2011) Llamó comunicaciones inalámbricas, a toda comunicación que carece de cables que unan tanto al emisor y como al receptor, más bien aquí interviene la modulación de las ondas electromagnéticas en el espacio. Así, los medios físicos solo están presentes en la emisión y la transmisión. Entre estos

podemos señalar los principales medios de comunicación inalámbrica en los que se puede lograr la comunicación en relación a la robótica, entre una máquina y un Smartphone:

- Punto a punto
- De darse el caso, debe haber comunicación en ambos sentidos
- La comunicación debe ser lo más sencilla posible

Generalmente las comunicaciones inalámbricas poseen varias clasificaciones ya sea según el área donde se ubica, el usuario se conecta a la red la denominamos “Área de cobertura” (Delgado, 2011). En la Figura 2. 11 se mostrará la clasificación de redes inalámbricas.

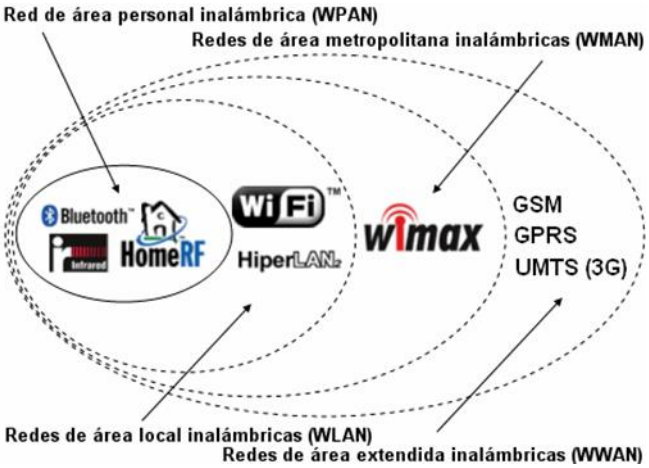


Figura 2. 11: Clasificación de redes inalámbricas.
Fuente: (Delgado, 2011)

2.9.1. Redes inalámbricas WPAN

Cubre un determinado sector limitado incluyendo decenas de metros. Su objetivo es ayudar con la conexión de aparatos tecnológicos y/o de todo tipo como celulares, impresoras, mouse, etc. Dirigidos a un ordenador base inalámbrico se puede realizar una conexión con 2 dispositivos no tan lejanos entre algunos modelos de tecnologías que abarca las WPAN podemos nombrar el “Bluetooth” (Delgado, 2011).

2.9.2. Zigbee

Más conocida IEE 802.15.4, la ventaja de esta tecnología es que tiene un costo muy accesible y consume un mínimo porcentaje de energía, es muy práctico y adecuado puesto que se adapta bien a los dispositivos electrónicos mayormente pequeños funciona en la frecuencia de 1.4 GHz y contiene 16 canales con una velocidad media de 250 Kbps y un alcance de hasta 100m (Delgado, 2011).

2.9.3. Infrarrojo:

Puede decirse que esta tecnología es la menos favorable para un usuario ya que a pesar de utilizar conexiones inalámbricas se lo debe hacer a muy pocos metros y su velocidad de transferencia por segundo no es muy conveniente. A pesar de eso aparatos en nuestros hogares necesitan de ello, así como un control remoto o el control de alarma de un vehículo. Infrared Data Association Actualmente tiene alrededor de 150 socios (Delgado, 2011).

2.10. Comunicación por Radiofrecuencia

La transferencia de datos sin la necesidad de guías de onda, se está haciendo cada vez más común en nuestra época. Además de ventajas como la fácil instalación ya que no requieren del uso de cables y por lo tanto no hace falta mucha inversión en este tipo de comunicación (Pazmiño & Sánchez, 2010).

Los elementos disponibles en la comunicación por radiofrecuencia han ido evolucionando constantemente, su objetivo es el de facilitar el proceso de transmisión y recepción. Esto se debe al surgimiento de circuitos transmisores totalmente integrados que pueden desempeñar ambas funciones como las de emisor y receptor. Estos dispositivos pueden ser utilizados en distintos sectores: comercial o industrial (Pazmiño & Sánchez, 2010).

El mayor aliado de estos sistemas es el espacio, pues, es su medio de TX. Puesto que la información que se transmite y se recepta contiene ondas electromagnéticas guiadas pero la información que viaja por el espacio debe contener ondas electromagnéticas no guiadas, es necesario un tercer componente, es aquí donde interviene la antena, que se encarga de hacer la conversión entre estas ondas ya sea de ondas no guiadas a guiadas como en el receptor, o por el contrario con el transmisor.

Estos sistemas tienen la propiedad de traspasar paredes y otros obstáculos. Que estos sistemas sean capaces de transmitir a corto o a largo plazo depende de varios aspectos: de la frecuencia de trabajo, a mayor frecuencia menor alcance (Pazmiño & Sánchez, 2010).

No olvidemos también que la antena influye mucho en el comportamiento la TX ya que la potencia de salida y la sensibilidad del receptor están en función al tipo de antena y de sus cualidades. Por ultimo existe un cuarto factor influyente en este proceso, y es el entorno en que el que se transmite, recordemos que hay mucha diferencia entre transmitir información en el aire libre, en la ciudad o dentro de un edificio.

Entonces la fuerza o intensidad con la que viajará una señal de un receptor a un transmisor, será directamente proporcional a la frecuencia, la fuerza de salida, la sensibilidad del receptor, las propiedades de la antena, y el medio de desenvolvimiento (Pazmiño & Sánchez, 2010).

De acuerdo a (Pino & Vallejo , 2010) existen dos tipos de módulos RF, estos dispositivos no lineales ocasionan una señal de transmisión con frecuencia mayor que la de recepción, poseen un mando de transmisión determinado y tienen la propiedad de reducir ruidos indeseables en la señal transmitida. Estos dos son:

- **RF activos**

Generan una señal en la transmisión con un nivel de fuerza más grande que el de la señal que llega al receptor.

- **RF pasivos**

Generan una señal que llega al receptor con un nivel de fuerza más pequeña que el de la que llega al transmisor. Analicemos algunas especificaciones de rendimiento sobre los módulos de radiofrecuencia que (Pino & Vallejo , 2010) mencionan:

- Sensibilidad: “Es la mínima señal de entrada necesaria para producir una señal de salida que tenga una determinada relación señal-ruido (S/N).”
- Potencia de salida: “Es el máximo de potencia de señal que se puede transmitir.”
- Frecuencia de funcionamiento: “Es el rango de transmisión de las señales recibidas.”
- Medición de resolución: “Es el mínimo de resolución digital.”
- Distancia máxima de transmisión: “Es la mayor distancia que un transmisor y un receptor pueden estar separados.”

2.10.1. Tipos de Comunicaciones Inalámbricas por RF

(Pazmiño & Sánchez, 2010) Destacan los siguientes tipos de comunicaciones por RF, estas son:

- Las que no cumplen ningún protocolo estándar.
- Las que cumplen un protocolo estándar.
- Las frecuencias de trabajo conocida como menores a 1GHz.
- Las frecuencias de trabajo de 2.4 GHz.

2.10.2. Bandas ISM

Estas bandas conocidas como Bandas: Industriales, Científicas y médicas. Actualmente los módulos inalámbricos están creados para poder operar en la banda de 2.4 GHz, esto significa que pueden ser usados en cualquier rincón del mundo ya que la banda descrita es útil a nivel global. Hay diversas tecnologías que operan en la banda de 2.4 GHz, una de ellas el conocido “Bluetooth” (Pazmiño & Sánchez, 2010).

2.10.3. Home Radiofrecuency

Fue publicada en el 98 por empresas COMPAQ, HP, INTEL, SIEMENS, MOTOROLA, MICROSOFT y otros. Con una velocidad de hasta 10 Mbps y un alcance no mayor a los 100m. Cinco años después este proyecto fue abandonado a pesar de tener el respaldo de INTEL. Puesto que los diseñadores de otros dispositivos comenzaron a utilizar WIFI Centrino (Delgado, 2011).

2.11. Tipos de Radiaciones

Existen dos tipos de radiación, respecto a la energía son:

- **Radiaciones Ionizantes:** disponen de una carga energética muy elevada, su característica principal es que estas radiaciones chocan con las células de cuerpo.
- **Radiaciones no Ionizantes**: Por el contrario, estas ondas tienen como propiedad proveer una ración insuficiente de carga energética en la alteración del organismo humano. Son aquellas las ondas en las que la frecuencia va conforme a las Radiaciones ultravioleta, infrarroja, microondas, ondas de radio y radiofrecuencia.

(Borobia, 2012) Menciona que, surgen diferentes radiaciones a los que el ser humano se expone a diario, como:

- Manipulación y flujo de energía eléctrica (ELF)

- En los ordenadores suele ocurrir una clase de calentamiento provocado por inducción de magnetismo.
- Transmisiones de radio por modulación de frecuencia.
- El mismo teléfono y los microondas en casa.

2.12. Comunicación Bluetooth

Un protocolo de comunicación muy importante en estos días es el "Bluetooth". Son varias las aplicaciones en las que este protocolo está involucrado y que hacen más fácil la vida a las personas. Su origen fue en uno de los países escandinavos como lo es Suecia, desarrollado por la empresa "Ericsson". Y así procederemos a exponer su funcionamiento, conceptos básicos y aplicaciones que se presentan actualmente (Martínez, 2014). En la Figura 2. 12 se enseña la simbología de la tecnología Bluetooth.

Figura 2. 12: Símbolo de la tecnología Bluetooth.
Fuente: (Martínez, 2014)

2.12.1. Protocolos

Esta tecnología consta de cuatro capas que incluyen protocolos, los protocolos que son indispensables para los dispositivos Bluetooth son los que se encuentran en el núcleo de Bluetooth y el de radio de Bluetooth. En la siguiente Figura 2. 13: se presentan los diferentes protocolos del Bluetooth.

Figura 2. 13: Grupo de protocolos en Bluetooth.
Fuente: (Pazmiño & Sánchez, 2010)

2.12.2. Especificaciones

Esta tecnología forma parte de las comunicaciones inalámbricas, tiene corto alcance y su labor principal es la de facilitar la comunicación de datos entre dispositivos móviles y otros dispositivos de mano. (Martínez, 2014) Afirma que Bluetooth se divide en 3 clases.

- Clase 1: Posee un alcance de cobertura aproximado de 100m.
- Clase 2: Posee un alcance de cobertura aproximado de 10m.
- Clase 3: Posee un alcance reducido de 1m.

No obstante (Pazmiño & Sánchez, 2010) exponen especificaciones técnicas las cuales se mencionan a continuación:

- Banda de Frecuencia: 2.4 GHz
- Canales máximos de voz: 3 por piconet (64 kbps bidireccional)
- Canales máximos de datos: 7 por piconet
- Velocidades de datos: hasta 721 kbps asimétrico o 433.9 kbps simétrico.
- Seguridad: Si, en la capa de enlace
- Interferencia: Bluetooth minimiza la interferencia potencial al emplear saltos rápidos en frecuencia (1600 veces por segundo).

Esta tecnología permite establecer comunicaciones aun en presencia de obstáculos, distancias como por ejemplo de 100m. Una aplicación podría ser reproducir archivos de música desde un teléfono móvil y de modo inalámbrico en un estéreo para carro. También sirve para conectar dispositivos como teclados y mouse hacia una máquina de escritorio (Martínez, 2014).

2.12.3. Ventajas y Desventajas de Bluetooth

Esta tecnología fue diseñada para facilitar la vida del hombre es así que surgieron muchas ventajas como las conexiones entre dispositivos con ordenadores y dispositivos con dispositivos si todos, si fueron diseñados con esta tecnología. Podemos observarlo cuando enlazamos nuestros dispositivos una cámara sin conexión física alguna. La transferencia de archivos de cualquier tipo nunca fue más fácil.

Por otro lado, esta tecnología contaba con una desventaja que más tarde la haría ser reemplazada por nuevas mejoras. La distancia de conexión no era la más favorable, ya que los dispositivos necesitaban estar relativamente cerca. Y si existía algún obstáculo dependiendo del material, este se convertía en un inconveniente impidiendo el enlace (Martínez, 2014).

2.12.4. La Seguridad en Bluetooth

(Pazmiño & Sánchez, 2010) Señalaron que, para mantener una privacidad en el envío y recepción de información, esta tecnología ofrece componentes de encriptación. La clave soporta longitudes de 64 bits o menos, por si alguna persona no autorizada quiere acceder a la información y editarla, entonces existen 3 niveles de enlace entre equipos:

- Nivel de seguridad A: Inseguro.
- Nivel de seguridad B: Servicio de seguridad aplicada.
- Nivel de seguridad C: Conexión de seguridad aplicada.

2.13. Redes WI-FI

Certificada por IEE 802.11 y respaldada por Wireless Ethernet Compatibility Alliance al igual que HiperLAN 2.0. Puede conectarse hasta 54 Mbps (Delgado, 2011). Es una herramienta orientada a la interconexión de dispositivos inalámbricos. En el presente muchos equipos caseros y portátiles como Smart TV, Smart Phone, PC, etc. Dejando a un lado la conexión por cable. Para lograr esto existen un de modos de acceso: el Access Point y Ad-hoc (Lalama, 2014).

2.13.1. Access Point

Tienen un encaminador denominado Router, quien es el mediador para el enlace entre el la red inalámbrica e internet (Lalama, 2014).

2.13.2. Ad-hoc

También facilita el traspaso de información inalámbricamente la diferencia es que éste no necesita de un encaminador que realice el trabajo de enlazar las redes, sino que cada es posible la interconexión directamente entre los nodos (Lalama, 2014).

CAPÍTULO 3: DISEÑO E IMPLEMENTACIÓN

3.1. Diseño e Implementación del Robot Laberinto.

El objetivo del robot Laberinto como su nombre lo indica, consiste en resolver el laberinto en el menor tiempo posible. El robot es ubicado en el punto de partida, y éste, utilizando su autonomía deberá encontrar la salida opuesta. A continuación, se describe un robot diseñado de forma casi redonda para evitar los atascos entre las paredes del laberinto, que a su vez se apoya sobre dos ruedas loca que le permite girar fácilmente sobre su propio eje con la finalidad de avanzar, y no requiere de la participación de ningún dispositivo o persona en cuanto a su sistema de control y alimentación se refiere.

En la Figura 3.1 se muestra el diagrama de bloques de las partes que integrarán al robot laberinto, que son: microcontrolador Atmega 328A, driver, sensor infrarrojo (sirve para saber la distancia del objeto al uC Atmega 328A), motores Pololu (cuya relación es 30:1), ruedas, batería de lipo 7.4

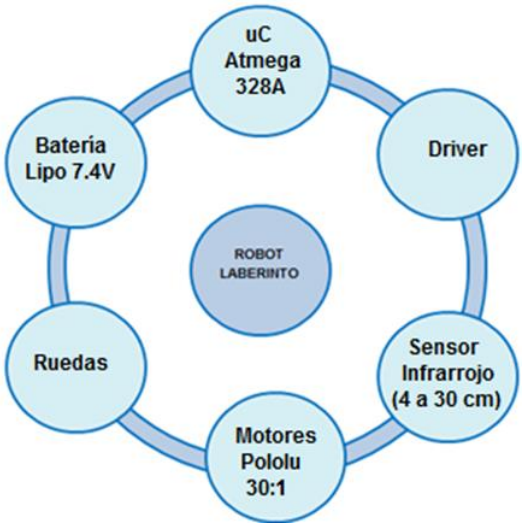


Figura 3. 1: Partes del robot Laberinto.
Elaborado por: Autores.

3.1.1. Arquitectura del robot laberinto.

En la Figura 3. 2 se muestra la arquitectura del robot, se diseñó así, con el propósito de evitar que el robot quede atrapado entre las paredes del laberinto. Es importante mencionar que, para esto, se ayuda mediante dos ruedas locas que giran sobre su propio eje hacia cualquier dirección. Además, el Laberinto tiene dos ruedas de tracción que mejoran el agarre del mismo sobre la superficie, las cuales tienen un diámetro de 22mm y están ubicadas en la placa superior de la tarjeta.

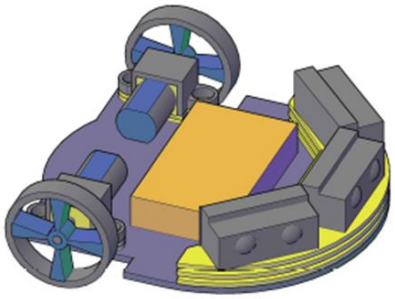


Figura 3. 2: Arquitectura del robot laberinto.
Elaborado por: Autores.

Además, para el desarrollo y construcción del robot se va a utilizar el microcontrolador Atmega 328A que será colocado en el diseño de la placa electrónica y que a su vez será el chasis del robot, tal como se muestra en la Figura 3. 3. El diseño de la placa fue realizado en el programa Altium, y que permite la conexión de los múltiples elementos que forman el robot laberinto (véase la Figura 3. 3).

Entre los demás componentes también se puede mencionar el puente H (modelo Tb6612), el mismo que nos permite controlar los motores, es decir, para realizar giros en ambos sentidos; otro componente es el microcontrolador Atmega 328A sirve para controlar y procesar la información proveniente de los 3 sensores infrarrojos GP2Y0A41SK0F (véase Figura 3. 4) con los que se ha dotado al mismo.

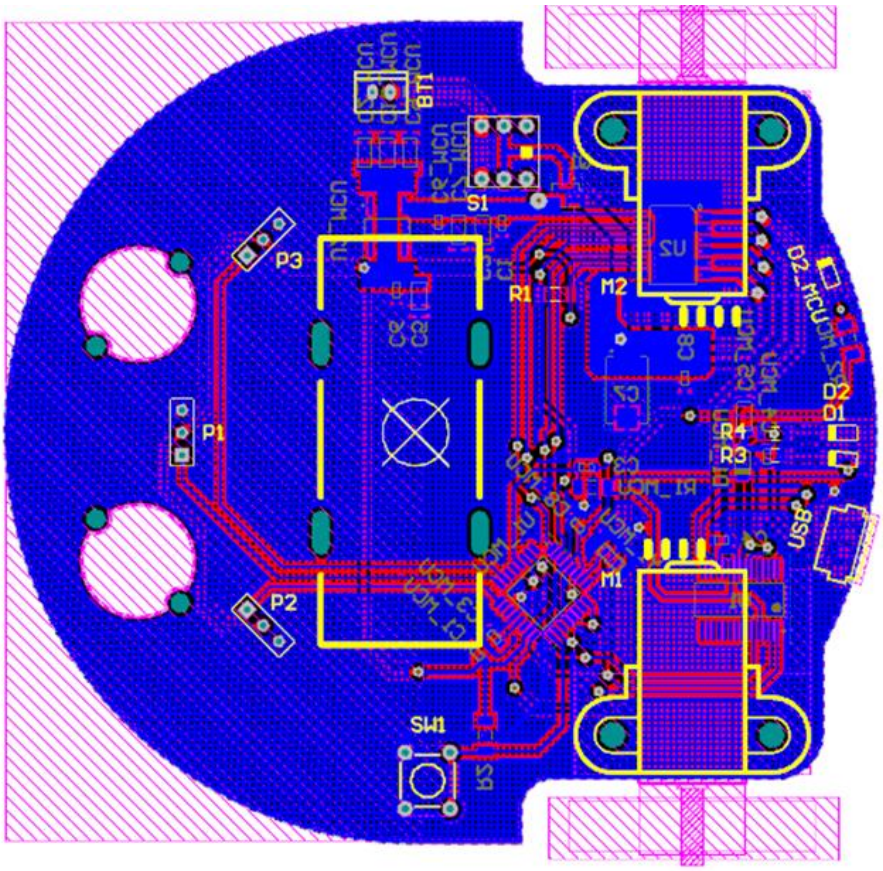


Figura 3. 3: Diseño de la placa del robot laberinto.
Elaborado por: Autores.

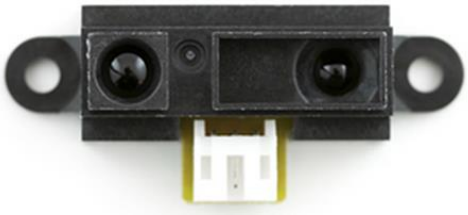


Figura 3. 4: Sensor infrarrojo GP2Y0A41SK0F de distancia.
Elaborado por: Autores.

3.1.2. Simulación en MatLab del motor dc 30:1.

Para el caso de los motores dc, estos cumplen la función de darle movimiento al robot. Dependiendo de la función que realice, la superficie sobre la cual se mueva, el tamaño, el peso o inclusive la precisión de los motores que

necesite el robot existen varias clases, entre los cuales se mencionan: motores de corriente continua, motores paso a paso o servomotores.

Para el robot Laberinto se han utilizado dos micromotores DC 30:1 HP cuyas dimensiones están entre 10x12x26 mm, los cuales son de alta potencia y operan con corriente continua. Estos motores funcionan con 6V, aunque en general pueden utilizarse voltajes más altos o bajos del voltaje nominal. Otra característica es que la caja de engranajes (cambios) tiene un largo de 0.365'', 3mm de diámetro en forma de D del eje de salida de metal y su peso es 9.5g.

Las características del motor 30:1 a 6 V con rapidez de 1000 rpm, se encuentran en la página www.pololu.com o buscando el Datasheet del motor. Aunque, para esto, se realizó un pequeño programa en MatLab para ingresar los parámetros del motor 30:1 y así obtener las gráficas de: corriente vs torque, velocidad vs torque, potencia de salida y eficiencia de energía.

A continuación, se muestra el código del programa en la Figura 3. 5.

```
% De la página web pololu tomar los valores para dibujar curvas del
motor
% de corriente continua.
StallTorque=input ('Ingrese el torque máximo del motor en oz-inch: ');
StallCurrent=input ('Ingrese la máxima corriente en mA: ');
RatedVoltage=input ('Ingrese el voltaje nominal en Voltios: ');
FreeRunCurrent=input ('Ingrese la corriente de funcionamiento en mA:
');
FreeRunSpeed=input ('Ingrese la velocidad del motor dc en RPM: ');
Resistance=RatedVoltage / StallCurrent;

torque1=0;
torque2=StallTorque; % oz-inch * (1/141.611932278) = N-m;

current1=FreeRunCurrent; % / 1000; %mA to Amps
current2=StallCurrent; %/ 1000; %mA to Amps

speed1=FreeRunSpeed; %RPM = 1/9.5493 radianes por segundos
speed2=0;

line ([torque1 torque2], [current1 current2]);
line ([torque1 torque2], [speed1 speed2]);
 totSamples=200; % total de muestras para finalizar la simulación
```

```
   redlineY = zeros (1, totSamples);
   redlineX = zeros (1, totSamples);
   bluelineY = zeros (1, totSamples);
   bluelineX = zeros (1, totSamples);
   greenlineY = zeros (1, totSamples);
   greenlineX = zeros (1, totSamples);
   orangelineY = zeros (1, totSamples);
   orangelineX = zeros (1, totSamples);

   % a = (torque2 - torque1) /totSamples;
   % looper = 1, torque = torque1, looper = 2, torque = torque1 + (t2-t1)
   /FS
   Index=1;
   maxPwrIndex=1;
   maxEffIndex=1;
   lastEff=0;
   lastPwr=0;

   for T = torque1:(torque2 - torque1) /totSamples:torque2
      bluelineY(Index)=(((speed2-speed1) /(torque2-torque1))*(T-
   torque1))+(speed1);
      bluelineX(Index)=T;

      redlineY(Index)=(((current2-current1) /(torque2-torque1))*(T-
   torque1))+(current1);
      redlineX(Index) = T;

      orangelineY(Index)=bluelineY(Index)*T; % Curva de la potencia de
   salida
      orangelineX(Index)=T;
      if orangelineY(Index)>lastPwr
          maxPwrIndex=Index;
          lastPwr=orangelineY(Index);

   end

greenlineY(Index)=orangelineY(Index)/((redlineY(Index))*RatedVoltage);
% Potencia de salida.
   greenlineX(Index)=T;
   if greenlineY(Index)>lastEff
       maxEffIndex=Index;
       lastEff=greenlineY(Index);
   end

   Index=Index+1;
end


fprintf ('Resistencia del motor = %f\n', Resistance)
fprintf ('Potencia máxima de salida %f @ Corriente = %f, Torque =
%f\n', orangelineY(maxPwrIndex), redlineY(maxPwrIndex),
orangelineX(maxPwrIndex))
```

```
fprintf ('Eficiencia máxima %f @ Corriente = %f, Torque = %f\n',
greenlineY(maxEffIndex), redlineY(maxEffIndex),
orangelineX(maxEffIndex))

dim1 = 2;
dim2 = 2;
subplot (dim1, dim2, 1);
plot (redlineX, redlineY);
title ('Corriente vs. Torque');
legend ('Corriente');
xlabel ('Torque (oz-in)');
ylabel ('Corriente mA');
subplot (dim1, dim2 ,2);
plot (bluelineX, bluelineY);
title ('Velocidad vs. Torque');
legend('Velocidad');
xlabel ('Torque (oz-in)');
ylabel ('Velocidad (RPM)');
subplot (dim1, dim2, 3);
plot (orangelineX, orangelineY);
title ('Potencia de Salida');
legend ('Potencia de Salida');
ylabel ('Potencia de Salida (Velocidad*Torque)');
xlabel ('Torque (oz-in)');
subplot (dim1, dim2, 4);
plot (greenlineX, greenlineY);
title ('Eficiencia de energía');
legend ('Eficiencia de energía');
ylabel ('Eficiencia de energía (Psalida /Pentrada)');
xlabel ('Torque (oz-in)');
```

Figura 3. 5 : Código de programación para el motor dc 30:1.
Elaborado por: Autores.

Una vez terminado el programa que se mostró en la figura 3.5 (código fuente, que se coloca como imagen debido a que la mayoría de instrucciones son las mismas que utiliza MatLab y pueden tener coincidencias en análisis de urkund) se procede a compilar y sin errores el mismo se ejecuta directamente desde el prompt de MatLab. Aquí, es donde se ingresarán los datos de las especificaciones para cualquier motor dc, solo hay que considerar que, para nuestro trabajo, el motor dc tiene la relación 30:1. Los parámetros solicitados se muestran en la Figura 3. 6.

```
>> simulacionpololu
Ingrese el torque máximo del motor en oz-inch : 9
Ingrese la máxima corriente en mA : 1.6
Ingrese el voltaje nominal en Voltios : 6
Ingrese la corriente de funcionamiento en mA : 0.120
Ingrese la velocidad del motor dc en RPM : 1000
Resistencia del motor = 3.750000
Potencia máxima de salida 2250.000000 @ Corriente = 0.860000, Torque = 4.5
Eficiencia máxima 577.732770 @ Corriente = 0.438200, Torque = 1.935000
```

Figura 3. 6: Ejecución de la simulación del motor dc 30:1.
Elaborado por: Autores.

En la figura 3.7 se muestra la gráfica de la corriente vs torque, después en la figura 3.8 vemos la gráfica de velocidad vs torque, posteriormente, en la figura 3.9 se observa la gráfica de la potencia de salida del motor 30:1, y finalmente, en la figura 3.10 se ilustra la gráfica de la eficiencia de la energía consumida por el motor.

De la Figura 3.7 se observa que el comportamiento de la corriente en el motor es que mientras más aumenta el torque en la pista del laberinto, la corriente también se incrementa, hasta llegar a su corriente máxima.

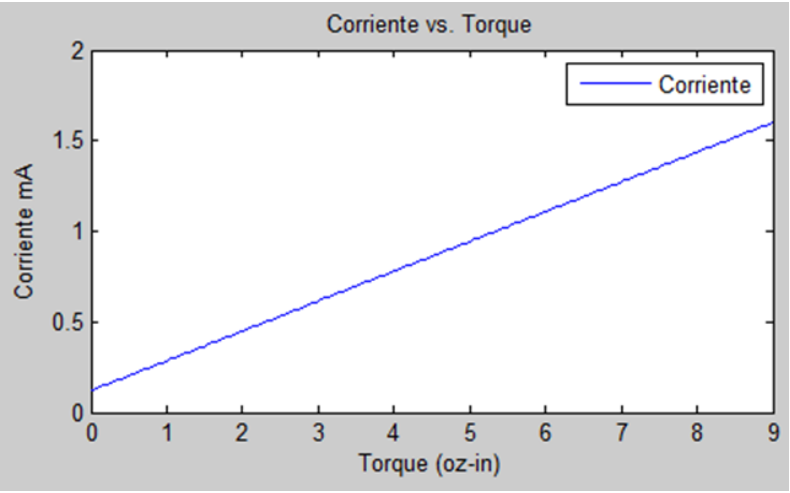


Figura 3. 7: Gráfica de la corriente versus torque para el motor dc 30:1.
Elaborado por: Autores.

En la Figura 3.8 se observa que la velocidad tiende a disminuir a medida que el torque se incrementa, aunque para el robot laberinto no va a disminuir mucho su velocidad debido a que no es un robot de pelea. La figura 3.9 se muestra la potencia de salida (multiplicación de velocidad y torque) versus torque.

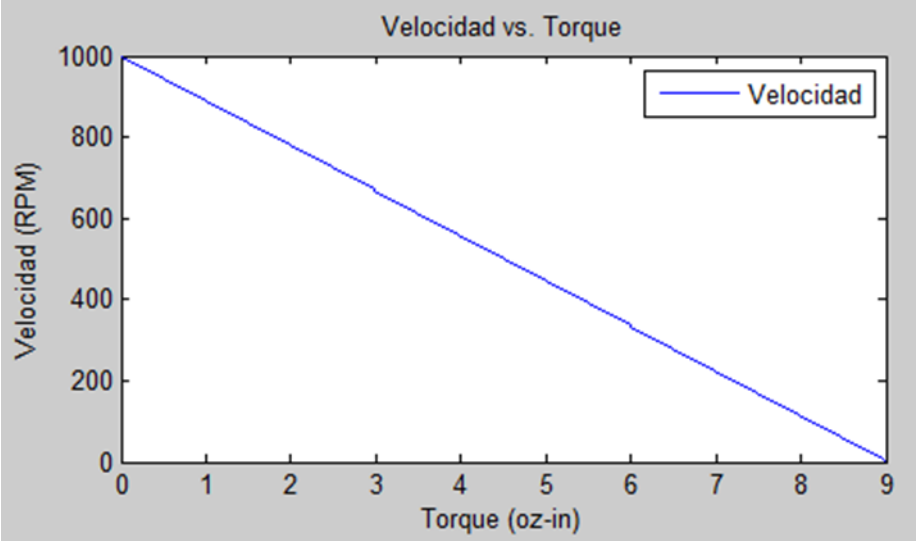


Figura 3. 8: Gráfica de la velocidad versus torque para el motor dc 30:1.
Elaborado por: Autores.

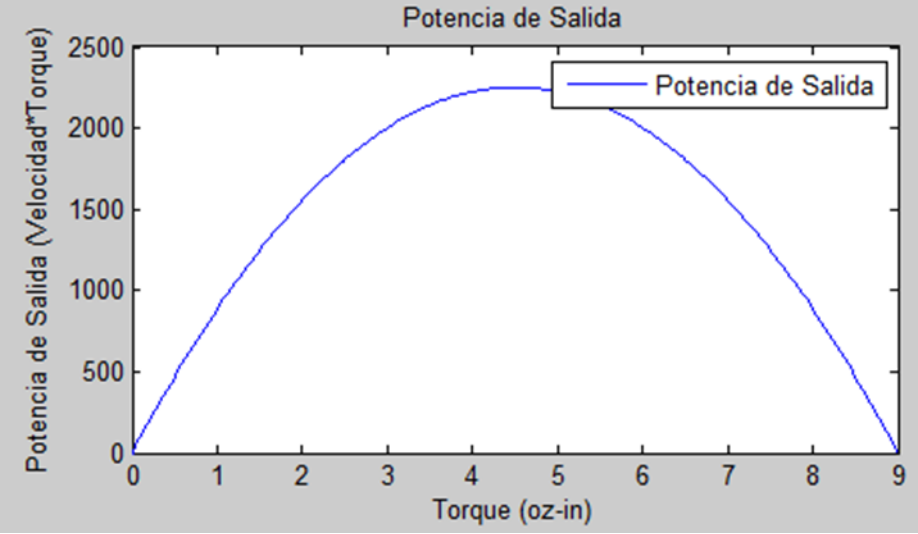


Figura 3. 9: Gráfica de la potencia de salida versus torque para el motor dc 30:1.
Elaborado por: Autores.

La Figura 3. 10 muestra la eficiencia energética del motor, esto nos permite que el motor pueda responder sin inconvenientes al momento de la competencia, ya que todo va radicar del tipo de batería lipo, es decir, que cumpla con los parámetros del motor dc. La relación del rendimiento es:

$$\eta = \frac{P_{salida}}{P_{entrada}}$$

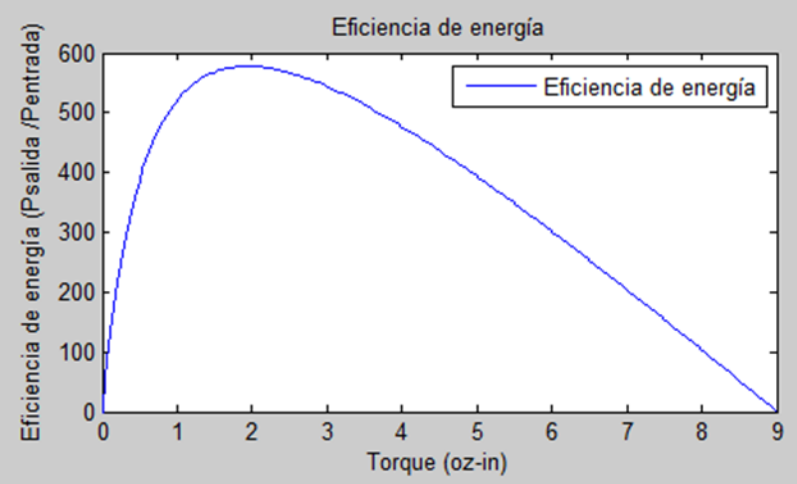


Figura 3. 10: Gráfica de la eficiencia de energía versus torque para el motor dc 30:1.
Elaborado por: Autores.

3.1.3. Diseño de la placa del robot laberinto en Altium.

El diseño de la placa del robot laberinto fue a doble capa. En la capa superior (véase la figura 3.11) van los motores dc, batería lipo, resistencias, diodos Leds, pulsador y puerto USB (para programación del uC Atmega 328A). En la figura 3.12 se muestra el diseño de la placa para la capa inferior, en el cual van los drivers o puente H, el microcontrolador Atmega 328A, capacitancias y resistencias.

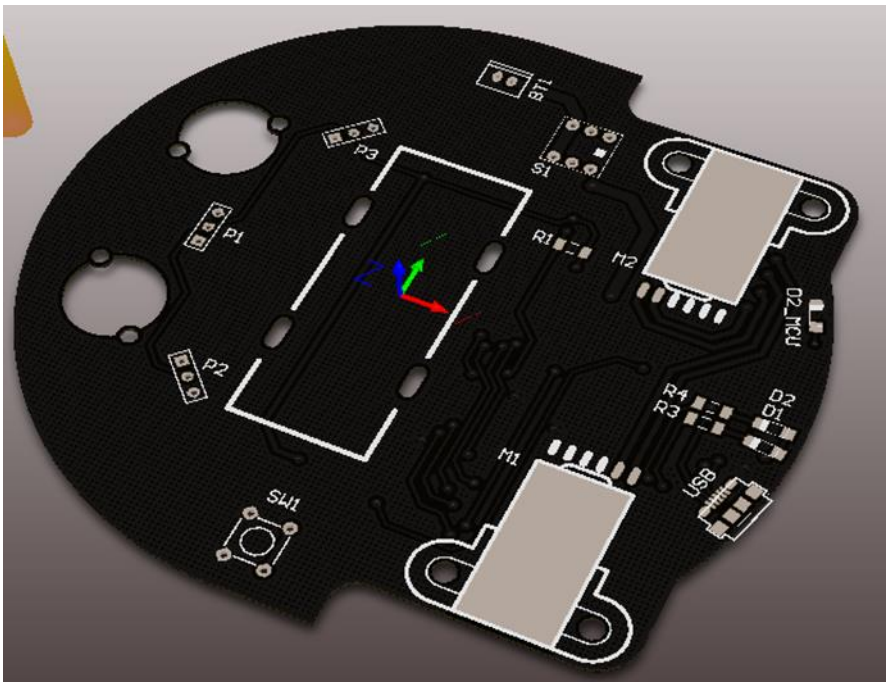


Figura 3. 11: Vista parte superior de la placa del robot laberinto.
Elaborado por: Autores.

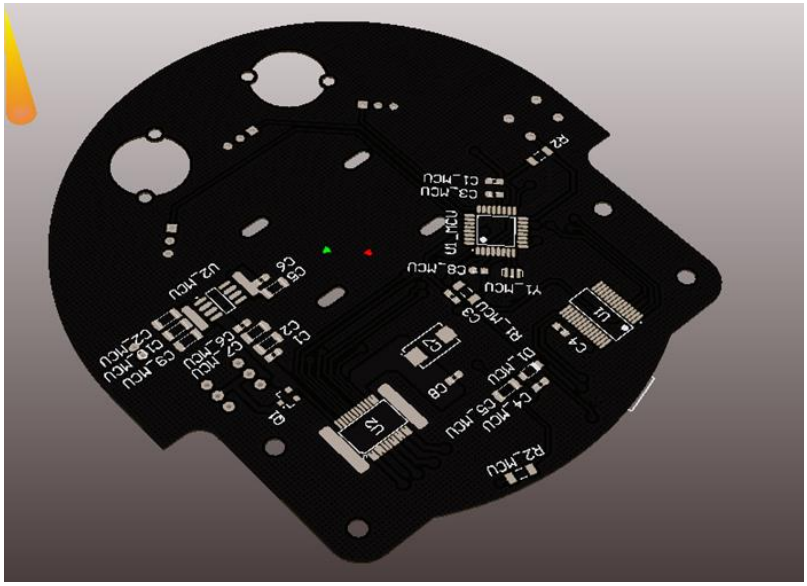


Figura 3. 12: Vista parte inferior de la placa del robot laberinto.
Elaborado por: Autores.

El diseño final del robot laberinto en tres dimensiones se muestran en las figuras 3.13 y 3.14, en la que se muestran implementados todos los dispositivos a utilizar en la parte superior e inferior de la placa.

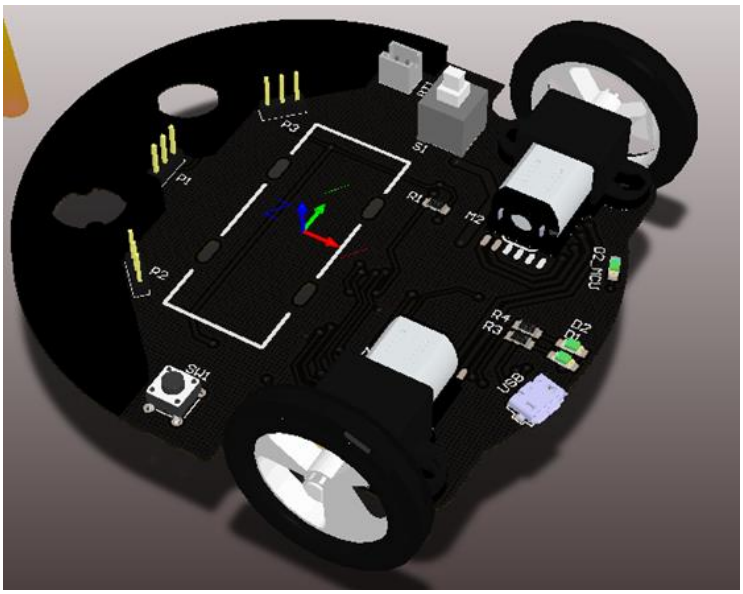


Figura 3. 13: Diseño final 3D de la parte superior del robot laberinto.
Elaborado por: Autores.

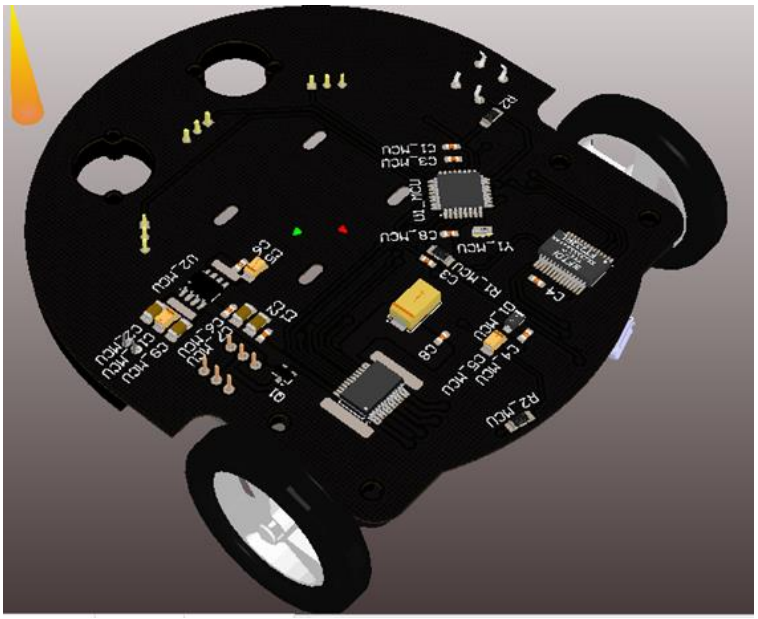


Figura 3. 14: Diseño final 3D de la parte inferior del robot laberinto.
Elaborado por: Autores.

3.1.4. Programación en IDE Arduino del robot laberinto.

A continuación, se describe el funcionamiento lógico del código fuente del robot Laberinto, el cual ha sido escrito en lenguaje de programación de alto nivel IDE Arduino que permite programar el microcontrolador Atmega 328A. IDE Arduino está compuesto por un conjunto de herramientas de programación que facilitan la comprensión de la actividad realizada por el robot laberinto. Para entender la programación debemos ver la posición de los sensores infrarrojos que se muestran en la figura 3. 15.

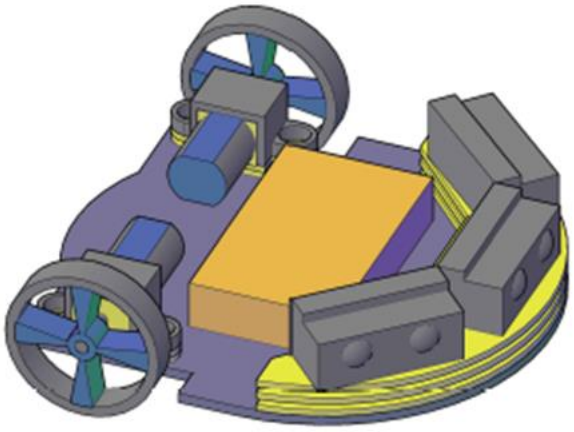


Figura 3. 15: Posición de los sensores infrarrojos.
Elaborado por: Autores.

De acuerdo al esquema funcional del diagrama de flujo del robot laberinto que se muestra en la figura 3.16, se empezará por distinguir las variables iniciales y más importantes del código fuente: DISTANCIA1, DISTANCIA2 y DISTANCIA3, las cuales almacenarán el valor medido por los sensores infrarrojos; INFRAROJO1, INFRAROJO2 e INFRAROJO3. Estos sensores son los encargados de hacer el cálculo de la distancia entre las paredes del laberinto.

Adicional a esto, las palabras MOTOR1 y MOTOR2 permitirán explicar el funcionamiento de los dos motores del robot mediante funciones predefinidas que ya están incluidas en las librerías del programa IDE Arduino. Inicialmente, el sensor INFRAROJO1 que está ubicado en la parte frontal del robot Laberinto

mide la distancia entre la pared del laberinto y la ubicación del mismo. Este valor calculado se almacena en la variable DISTANCIA1 para permitirle al microcontrolador Atmega 328A preguntar si dicho valor es mayor a veinte. Se toma como referencia el número veinte ya que las paredes del laberinto distan entre sí con ese valor.

Si el valor que reside en la variable DISTANCIA1 es mayor a veinte, entonces el sensor INFRAROJO2 que está ubicado en el costado derecho del robot calcula la distancia con su pared respectiva. La nueva distancia se almacena en la variable DISTANCIA2 para que, posteriormente el microcontrolador pregunte si dicha distancia es menor a cinco. Como el robot Laberinto mide aproximadamente 10cm y el ancho del laberinto es de 20cm, el número cinco equivale al valor que centra al robot dentro del mismo con el objetivo de no chocar con la pared opuesta a la distancia que está calculando.

En caso de que DISTANCIA2 contenga un valor menor a cinco, el robot se repelará hacia su derecha mientras avanza por el laberinto. Por consiguiente, el sensor INFRAROJO3 calculará la distancia respectiva con su pared y almacenará este valor en la variable DISTANCIA3. El objetivo de este proceso es que el microcontrolador pregunte si el valor acumulado en dicha variable también es menor a cinco. De ser así, el robot se repelará hacia la izquierda y se adelantará ejecutando el proceso cuantas veces sea necesario hasta encontrar la salida. Pero si la DISTANCIA2 contiene un valor mayor a cinco inmediatamente el microcontrolador comparará el valor guardado en DISTANCIA3 y si este valor también fuera mayor a cinco, se iniciará el proceso nuevamente.

Regresando a la primera condición, si la variable DISTANCIA1 acumula un valor menor a veinte entonces el microcontrolador comparará los valores que yacen en las variables DISTANCIA 2 y DISTANCIA3 de tal manera que, si el valor

de la variable DISTANCIA2 es mayor que el de la variable DISTANCIA3, el robot hará un giro a la izquierda.

En el proceso de giro, el MOTOR1 disminuirá la velocidad en tanto el MOTOR2 la aumenta. Para ello el código fuente se vale de símbolos establecidos por el programa, los cuales pueden tomar el valor de cero (0) o uno (1). El microcontrolador continúa en la toma de decisiones, pregunta de manera inversa si el valor guardado en la variable DISTANCIA3 es mayor que el contenido en la variable DISTANCIA2; si es así, el robot opta por un giro a la derecha donde el MOTOR 2 reduce su velocidad mientras que el MOTOR 1 la aumenta.

Sin embargo, si la variable DISTANCIA2 contiene un valor menor al de la variable DISTANCIA3 o, por el contrario, el valor de la variable DISTANCIA3 es menor que el de la variable DISTANCIA2; el microcontrolador entenderá que el robot está atrapado en un lugar del laberinto y ordenará a los sensores para que tomen nuevas lecturas del medio y por ende se realizará el proceso otra vez.

Cabe recalcar que el proceso descrito anteriormente se hace en microsegundos de tal forma que el robot siempre está en movimiento. Para una mayor comprensión del funcionamiento del robot Laberinto véase la figura 3.16.

A continuación, se muestra las líneas de programación en IDE Arduino del robot laberinto. En las siguientes líneas de programación se declaran las librerías para los motores y sensores infrarrojos, así como declarar la botonera de inicio (Start_bt 12) y el pin para habilitar (Ena_Motors 4) los dos motores dc 30:1.

```
#include <OrangutanMotors.h>
#include <DistanceGP2Y0A41SK.h>
#include <DistanceGP2Y0A21YK.h>

#define Start_bt      12              // Pin Botonera
#define Ena_Motors     4              // Pin Habilitar Motores
```

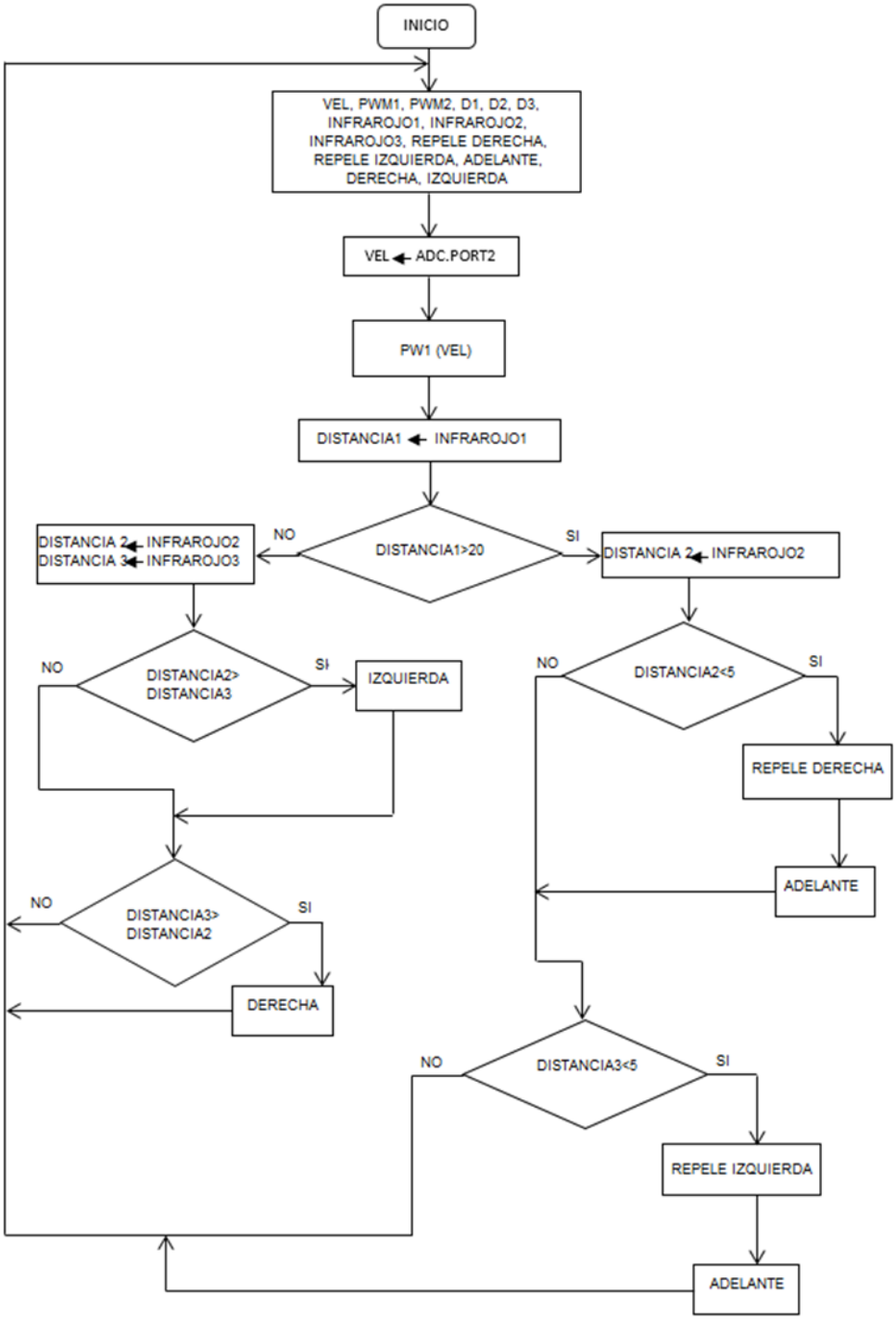


Figura 3. 16: Diagrama ASM del robot laberinto.
Elaborado por: Autores.

De acuerdo a las librerías del sensor infrarrojo, se está definiendo las variables de esa función, así que más adelante se refieren a Dist1 (sensor infrarrojo frontal), Dist2 (sensor infrarrojo derecho) y Dist3 (sensor infrarrojo izquierdo) como la función que hay que ir según como se estableció en include.

```
DistanceGP2Y0A41SK Dist1;
DistanceGP2Y0A41SK Dist3;
DistanceGP2Y0A21YK Dist2;
```

A continuación, se declaran las variables int de tipo entero y para los valores de kp, ki y kd (controlador proporcional, integrativo y derivativo) son de tipo flotante.

```
int val = 0;
int state = 1;
int Inicio;
int mode = 0;
int DistDer = 0;
int DistIzq = 0;
int DistFrtn = 0;

int Vgiro;
int Set_Point;
int proporcional,last_proporcional,integral,derivativo,power_difference,Vmax;
float Kp,Ki,Kd;
```

Posteriormente, se crea el objeto de los motores según la declaración de la librería. A continuación, se muestra las líneas de programación de la función de configuración pinMode (expresión para determinar el comportamiento de los pines), en este caso recuerde q Start_bt es el pin 12 antes descrito y dice que será de salida. Así mismo la línea siguiente, Serial.begin define la velocidad que trabajará, en este caso será a 9600 baudios. Mientras que Dist1.begin, Dist2.begin y Dist3.begin toman a los puertos A1, A2 y A3 como analógicos para recibir los valores analógicos y convertirlos en un valor con que trabajar.

```
OrangutanMotors motors;       // Creo el Objeto Motors

void setup() {

  pinMode(Start_bt,INPUT);
  pinMode(Ena_Motors, OUTPUT);
  Serial.begin(9600);
  Dist1.begin(A1);
  Dist2.begin(A2);
  Dist3.begin(A3);
```

Los coeficientes del control (kp → proporcional, ki → integrativo y kd → derivativo), así como las variables de control, son declarados en las siguientes líneas de programación:

```
//Coeficientes del Control
 Set_Point = 8;
 Kp = 21;
 Ki = 0.0001;
 Kd = 18;
//Variables del Control
 proporcional= 0;
 last_proporcional= 0;
 integral= 0;
 derivativo= 0;
 power_difference= 0;
 Vmax= 100;
 Vgiro = 110;
```

A continuación, se muestra las líneas de programación del código fuente del robot laberinto.

```
  digitalWrite(Ena_Motors,HIGH);
  motors.setSpeeds(0,0);              // Motores detenidos

  while (val < 50) {
    val=0;
    if (mode>2)
    mode = 0;

//    mode -----> 0   Derecha
//    mode -----> 1   Izquieda
//    mode -----> 2   Correccion

    switch (mode) {
     case 0: state = !state;                 // Por Derecha
             delay(800);
             Inicio = 0;
             digitalWrite(13, state);
             break;
   case 1: state = !state;                 // Por Izquierda
           delay(300);
           Inicio = 1;
           digitalWrite(13, state);
           break;
   case 2: state = !state;                 // Camino COrregido
           delay(80);
           Inicio = 2;
           digitalWrite(13, state);
           break;
  }

if(digitalRead(Start_bt) != 1) {
 while(digitalRead(Start_bt) != 1){
  if (val > 50) {
   digitalWrite(13, LOW);
   state = 1;
   }else{
   delay(125);
   digitalWrite(13, HIGH);
   val++;
   }
 }
```

Con las siguientes líneas de programación, se configura el inicio del robot. Se debe colocar en el centro de la pista de laberinto y así se apagará el led, lo

que indica que ha sido calibrado; y en esta posición se apaga el Led indicando que está equidistante a 8 cm. Después con solo pasar la mano al frente del sensor, el robot laberinto inicia su participación hasta buscar la salida del laberinto en el menor tiempo posible.

```
DistFrtn = Dist2.getDistanceCentimeter();
while (DistFrtn > 11) {
  DistDer = Dist3.getDistanceCentimeter();
  DistIzq = Dist1.getDistanceCentimeter();
 switch (Inicio) {
   case 0: if (DistDer == Set_Point)
           digitalWrite(13, LOW);
           else
           digitalWrite(13, HIGH);
           break;
   case 1: if (DistIzq == Set_Point)
           digitalWrite(13, LOW);
           else
           digitalWrite(13, HIGH);
           break;
  }
 DistFrtn = Dist2.getDistanceCentimeter();
  delay(50);
}
```

A continuación, se muestra el programa principal de contorneo derecha e izquierda:

```
//  Programa Principal
void loop() {
    DistDer  = Dist3.getDistanceCentimeter();
    DistIzq  = Dist1.getDistanceCentimeter();
    DistFrtn = Dist2.getDistanceCentimeter();
    switch (Inicio) {
    // Contorneo por Derecha
     case 0:
     if((DistFrtn > 13) && (DistDer < 25)){
        proporcional = DistDer - Set_Point;
        derivativo = proporcional - last_proporcional;
        integral = integral + proporcional;
        last_proporcional=proporcional;
        power_difference=(5*proporcional)+(integral*0.00)+((derivativo*6)/2);
```

```
   if(power_difference > Vmax)
      power_difference = Vmax;
   else
   if(power_difference < -Vmax)
      power_difference = -Vmax;
   if(power_difference < 0)
      motors.setSpeeds(Vmax, Vmax+power_difference);
      else
       motors.setSpeeds(Vmax-power_difference, Vmax);
} else   if(DistDer > 25){

       motors.setSpeeds(Vgiro,Vgiro);
       delay(130);
       while(DistDer > 9){
       DistDer  = Dist3.getDistanceCentimeter();
       motors.setSpeeds(Vgiro/3, Vgiro);
       }

} else   if(DistFrtn < 13){

     if (DistIzq < 15){
       motors.setSpeeds(0,0);
       delay(5);
       motors.setSpeeds(Vgiro,-Vgiro*1.3);
       delay(5);
        }else{
         motors.setSpeeds(Vgiro,0);
         delay(5);
       }
  }

          break;

// Contorneo por Izquierda
  case 1:
  if(DistFrtn > 13 && DistIzq < 25){

     proporcional = DistIzq - Set_Point;
     derivativo = proporcional - last_proporcional;
     integral = integral + proporcional;
     last_proporcional = proporcional;

     power_difference=Kp*proporcional+Ki*integral+Kd*derivativo;

     if(power_difference > Vmax)
     power_difference = Vmax;
```

```
    if(power_difference < -Vmax)
    power_difference = -Vmax;

          if(power_difference < 0)
           motors.setSpeeds(Vmax+power_difference, Vmax);
          else
           motors.setSpeeds(Vmax, Vmax-power_difference);
          break;

} else   if(DistIzq > 25){

        motors.setSpeeds(Vgiro,Vgiro);
        delay(120);
        while(DistIzq > 9){
        DistIzq  = Dist1.getDistanceCentimeter();
        motors.setSpeeds(Vgiro, Vgiro/3);
        }

} else   if(DistFrtn < 13){

      if (DistDer < 15){
        motors.setSpeeds(0,0);
        delay(5);
        motors.setSpeeds(-Vgiro*1.3,Vgiro);
            delay(5);
          }


     }

     // Camino Corregido
     case 2:

            break;
    }

}
```

3.2. Implementación del kit robot balancín.

La implementación del robot balance de dos ruedas utiliza, la tarjeta Arduino UNO basado en Shield balance, motores dc a 12 V con relación 43.8:1, driver L298P, un giroscopio MPU6050, batería de lipo, módulo de comunicación inalámbrica Wifi, Bluetooth y comunicación por radiofrecuencia. Adicional el

dispositivo se puede manipular vía celular inteligente, PC, control RC. En la figura 3.17 se muestra el diseño de la estructura del robot balance.

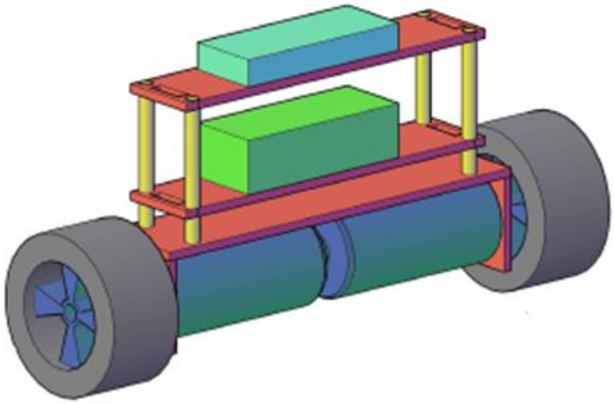


Figura 3. 17: Estructura del robot balance.
Elaborado por: Autores.

A continuación, se describiremos el procedimiento paso a paso el ensamblaje de nuestro robot balancín. Consta de varias partes en acrílico, por su costo y fácil accesibilidad fue el material elegido, además facilitó realizar los cortes en una maquina CNC según las especificaciones que requeríamos. Se diseñó tres plataformas, cada una de ellas sirve de soporte para una parte fundamental del robot. El primer piso o base nos ayuda a soportar los motores y las otras plataformas superiores, en el segundo nivel sirve de apoyo a la batería de lipo de 3000 mAh, y el ultimo nivel nos ayuda de cimiento para la parte electrónica de control.

En la figura 3.18 se demuestra la base de los motores, que estarán sujetos a la primera plataforma inferior. Estos motores fueron elegidos entre los diferentes modelos que encontramos en el mercado por que incluyen los encoders en sus ejes. Este encoder nos permite realizar el algoritmo de programación que sirve a corregir y mantener el robot en equilibrio.

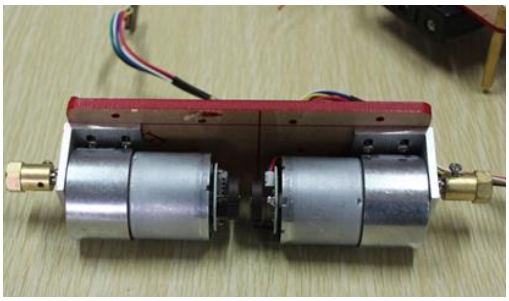


Figura 3. 18: Soporte de motores en base de acrílico.
Elaborado por: Autores.

El siguiente nivel superior va la batería de lipo y se ajustará como se muestra en la figura 3.19, con soportes metálicos y con tornillos. Es importante que todo quede bien sujeto para que nuestro prototipo no sufra deformación en la estructura cuando realice movimientos bruscos.

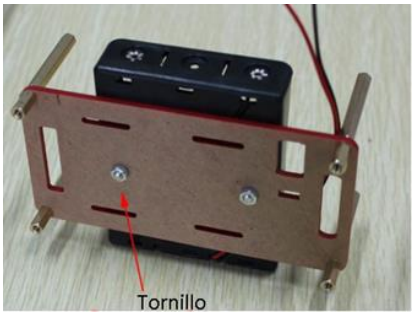


Figura 3. 19: Soporte de motores en base de acrílico.
Elaborado por: Autores.

3.2.1. Tarjeta controladora Arduino UNO.

La tarjeta Arduino UNO (Open Hardware) que se muestra en la figura 3.20 se utiliza como el controlador del robot. Arduino UNO es en sí una CPU, está pequeña placa de desarrollo fue creado para la experimentación y con propósitos de aficionados. Se tienen 54 pines E/S, entre los cuales de clasifican así: 14 pines de salidas PWM, 16 entradas analógicas, 4 UARTs con 1 puerto I2C y 3 puertos SPI disponibles. La tarjeta tiene incorporado el microcontrolador Atmega 2560 con una velocidad de reloj de 16 MHz. En la figura 3.21

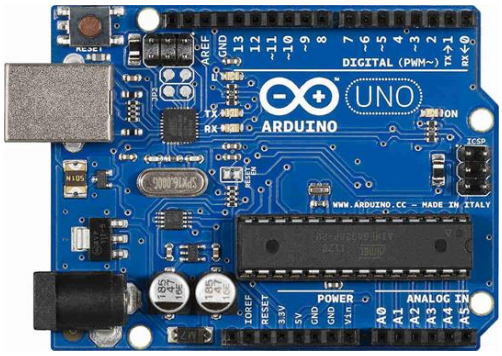


Figura 3. 20: Microprocesador Arduino UNO.
Fuente: (Arduino, 2016)

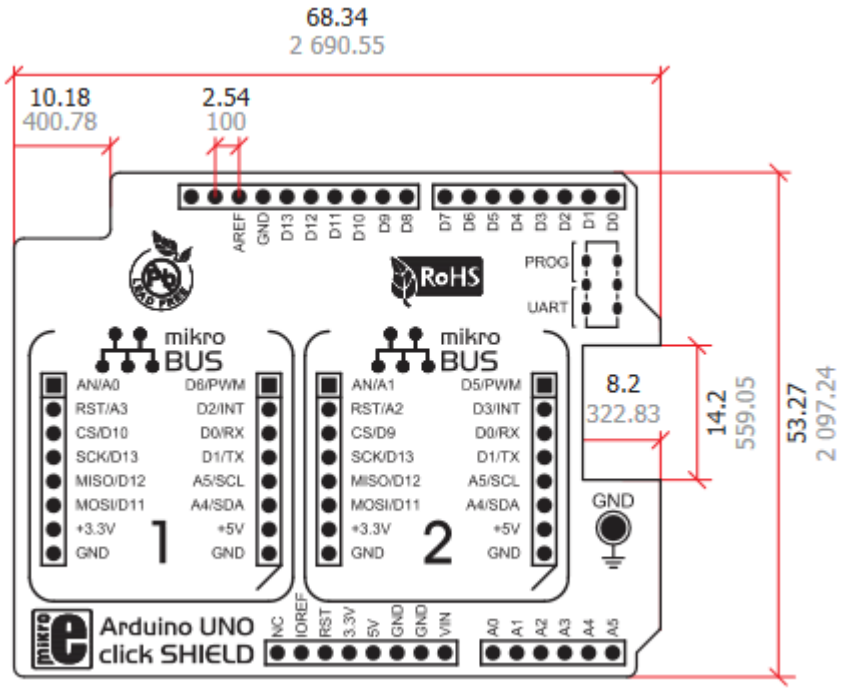


Figura 3. 21: Esquemático de la placa Arduino UNO.
Fuente: (Mikroe, 2016)

Arduino Uno, requiere un voltaje de entrada entre 7 y 12 V para el funcionamiento óptimo y puede ser alimentado y ser programado a través de un cable USB. También viene con una programación IDE de Arduino y un cargador de arranque pre-programado. El IDE utiliza tanto para la adaptación de lenguaje Arduino y/o para el procesamiento de la escritura de código C ++.

3.2.2. Giroscopio MPU6050.

Para el control de nuestro prototipo es necesario que la Shield proporcione el resultado de la lectura de un sensor importante como, es el giroscopio. Para este caso ya contiene el modelo MPU6050, tal como se muestra en la figura 3.22. Este mismo es utilizado sobre muchas placas de control de drones por ejemplo que por su velocidad de reacción y costos son los mejores en esta área.

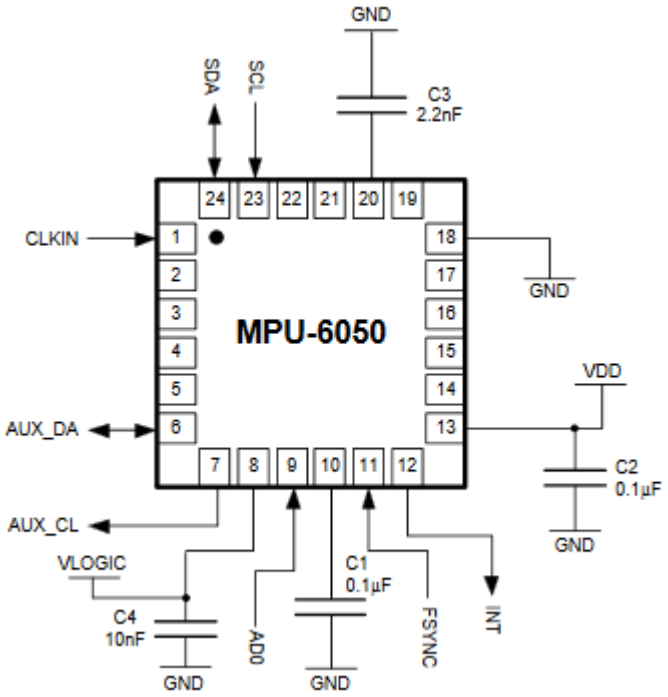


Figura 3. 22: Esquemático del sensor giroscópico MPU6050.
Fuente: (InvenSense, 2013)

Para este trabajo de titulación, específicamente para el robot balance, el sensor MPU6050 permite controlar el equilibrio del robot. Sólo tiene que incluir un módulo Bluetooth y utilizar la aplicación Arduino Bluetooth RC car compatible con los teléfonos inteligentes cuyo sistema operativo es Android, para que el mando sea a distancia.

3.2.3. Driver L298P para el motor DC.

Para alimentar los motores dc se necesitan de drivers (puente H) y se escogió el L298P para el control de los dos motores dc a 12 V. Fue elegida porque se ajusta fácilmente en la Arduino UNO y las únicas conexiones necesarias a excepción de los pasadores ya conectados a través del escudo eran la principal potencia a los motores de 12V. L298 es un puente H y puede suministrar hasta 2A por canal.

Trabaja con voltajes lógicos tanto con 3.3 V y 5 V, donde 5 V es el nivel en el sistema. Los motores se controlaron mediante el establecimiento de un indicador digital de 0 o 1 en el pin de dirección para decidir qué dirección iría el motor y mediante el suministro de una señal PWM para determinar la velocidad de los motores donde 255 estaba lleno de energía del voltaje de alimentación Vin y 0, es completamente apagado. Esto permite 256 niveles diferentes de un lado a otro de la cual el motor da un lapso total de 511 velocidades.

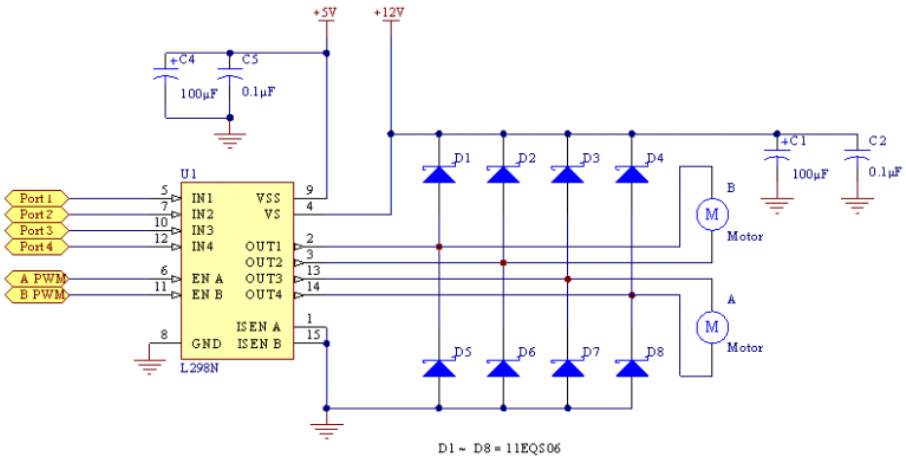


Figura 3. 23: Esquemático del controlador o driver L298P.
Fuente: Autor.

3.2.4. Programación del robot balance.

Declaración de librerías para comunicación del receptor de radiocontrol (RC) y de las interrupciones para lectura de encoder:

```
#include <Wire.h>
#include "Kalman.h"
#include "ComPacket.h"
#include "I2C.h"
#include <EEPROM.h>
#include "MyEEprom.h"
#include <PinChangeInt.h> // para receptor RC
```

Declaración de variables que son necesarias para la programación del robot balance:

```
volatile uint8_t bUpdateFlagsRC = 0;
#define UP_DOWN_FLAG   0b10
#define LEFT_RIGHT_FLAG 0b100
uint32_t UpDownStart;
uint32_t LeftRightStart;
volatile uint16_t UpDownEnd = 0;
volatile uint16_t LeftRightEnd = 0;
int UpDownIn;
int LeftRightIn;
#define SPK_OFF  0x00
#define SPK_ON  0x01
#define SPK_ALARM 0x02
Kalman kalmanX;
Kalman kalmanY;
ComPacket SerialPacket;

double accX, accY, accZ;
double gyroX, gyroY, gyroZ;
int16_t tempRaw;
// Ángulos usando giroscopio
double gyroXangle, gyroYangle;
// Ángulos usando filtros complementarios
double compAngleX, compAngleY;
// Ángulos ussando filtro Kalman
double kalAngleX, kalAngleY;
uint32_t timer;
// Buffer para datos I2C
uint8_t i2cData[14];
```

A continuación, se muestran las líneas de código para la declaración de pines de la tarjeta controladora:

```
#define BUZZER 13
#define LED 13
//motor  define
#define PWM_L 6  //Motor 1 (izquierda)
#define PWM_R 5   //Motor 2 (derecha)
#define DIR_L1 8
#define DIR_L2 7
#define DIR_R1 3
#define DIR_R2 4
//encoder define
#define SPD_INT_R 11   //interrupción
#define SPD_PUL_R 12
#define SPD_INT_L 10   //interrupción
#define SPD_PUL_L 9
```

En las siguientes líneas de código, procedemos a declarar las constantes para el controlador PID (proporcional, integrativo y derivativo), así como, la velocidad de los motores DC, el tiempo de reacción, el ángulo de inclinación, y el sentido de giro de los motores.

```
int Speed_L,Speed_R;
int Position_AVG;
int Position_AVG_Filter;
int Position_Add;
int pwm,pwm_l,pwm_r;
double Angle_Car;
double Gyro_Car;
double KA_P,KA_D;
double KP_P,KP_I;
double K_Base;
struct EEpromData SavingData;
struct EEpromData ReadingData;
int Speed_Need,Turn_Need;
int Speed_Diff,Speed_Diff_ALL;
```

```
uint32_t LEDTimer;
bool blinkState = false;
uint32_t calTimer;
uint32_t delayTimer;
uint32_t  lampTimer;
byte speakMode;
int tonelength;
int tonePin=13;
int tonecnt = 0;
uint32_t  BuzzerTimer;
```

Después, se realiza el inicio de configuración tanto de entradas y salidas, así como, el inicio de los protocolos de comunicación serial I2C:

```
void setup() {
  Serial.begin(9600);
  Init();
  Wire.begin();
  // Configuración del I2C con frecuencia a 400kHz
  TWBR = ((F_CPU / 400000L) - 16) / 2;

  // Establecer la frecuencia de muestreo para:
  // 1000Hz - 8kHz/(7+1) = 1000Hz
  i2cData[0] = 7;
  // Desactivar FSYNC y establecer 260 Hz para filtrado,
  // 256 Hz para filtrado del giroscopio,
  // y muestreo de 8 KHz
  i2cData[1] = 0x00;
  // Ajustar el rango de ecala del giroscopio para ±250deg/s
  i2cData[2] = 0x00;
  // Ajustar el rango de escala del acelerómetro para ±2g
  i2cData[3] = 0x00;
```

Se va a establecer la frecuencia del protocolo I2C, frecuencia de muestreo, filtrado normal y del giroscopio, así como, el rango del giroscopio y acelerómetro. escala Pregunta de datos mediante protocolos para inicializar el mismo:

```
// Escribir en los cuatro registros al mismo tiempo
while (i2cWrite(0x19, i2cData, 4, false));
// PLL con referencia al eje X del giroscopio y
// desactivar el modo de suspensión
while (i2cWrite(0x6B, 0x01, true));

while (i2cRead(0x75, i2cData, 1));
if (i2cData[0] != 0x68) {
  Serial.print(F("Error de lectura del sensor"));
  while (1);
}
delay(200); // Esperar a que el sensor se estabilice

/* Configuración de kalman y ángulo de inicio del giroscopio */
while (i2cRead(0x3B, i2cData, 6));
accX = (i2cData[0] << 8) | i2cData[1];
accY = (i2cData[2] << 8) | i2cData[3];
accZ = (i2cData[4] << 8) | i2cData[5];
```

Inicio de muestreo de interrupciones para encontrar el cambio de estado entre pulsos de los encoder y ubicar el grado de inclinación del robot en radianes.

```
#ifdef RESTRICT_PITCH
  double roll=atan2(accY, accZ) * RAD_TO_DEG;
  double pitch=atan(-accX/sqrt(accY*accY + accZ*accZ))* RAD_TO_DEG;
#else // Eq. 28 and 29
  double roll=atan(accY/sqrt(accX*accX + accZ*accZ))* RAD_TO_DEG;
  double pitch=atan2(-accX, accZ)* RAD_TO_DEG;
#endif

  kalmanX.setAngle(roll); // Ajustamos el ángulo de partida
  kalmanY.setAngle(pitch);
  gyroXangle = roll;
  gyroYangle = pitch;
  compAngleX = roll;
  compAngleY = pitch;
```

Comparación con el grado de inclinación que se establece para evitar colisión con el suelo:

```
void loop() {
  double DataAvg[3];
  double AngleAvg = 0;
  DataAvg[0]=0; DataAvg[1]=0; DataAvg[2]=0;
    while(1)
    {
      if(UpdateAttitude())
      {
        DataAvg[2] = DataAvg[1];
        DataAvg[1] = DataAvg[0];
        DataAvg[0] = Angle_Car;
        AngleAvg = (DataAvg[0]+DataAvg[1]+DataAvg[2])/3;
        if(AngleAvg < 40 || AngleAvg > -40){
          PWM_Calculate();
          Car_Control();
        }
      }
      UserComunication();
       ProcessRC();
      MusicPlay();
    }
 }
```

Explicación de las subclases y del cálculo del controlador PID:

```
void PWM_Calculate()
{
  float ftmp = 0;
  ftmp = (Speed_L + Speed_R) * 0.5;
  if( ftmp > 0)
    Position_AVG = ftmp +0.5;
   else
    Position_AVG = ftmp -0.5;

    Speed_Diff = Speed_L - Speed_R;
    Speed_Diff_ALL += Speed_Diff;
```

```
Position_AVG_Filter = Position_AVG;
Position_Add += Position_AVG_Filter;  //position
Position_Add += Speed_Need;  //
Position_Add = constrain(Position_Add, -800, 800);
pwm =  (Angle_Car-5 + K_Base)* KA_P   //P
    + Gyro_Car * KA_D //D
    +  Position_Add * KP_I    //I
    +  Position_AVG_Filter * KP_P; //P
```

Ahora, preguntamos el control de PWM de los motores DC:

```
if((Speed_Need != 0) && (Turn_Need == 0))
{
  if(StopFlag == true)
  {
    Speed_Diff_ALL = 0;
    StopFlag = false;
  }
  pwm_r = int(pwm + Speed_Diff_ALL);
  pwm_l = int(pwm - Speed_Diff_ALL);
}
else
{
  StopFlag = true;
   pwm_r = pwm + Turn_Need; //
   pwm_l = pwm - Turn_Need;
}
Speed_L = 0;
Speed_R = 0;
```

Las siguientes líneas, se establece el sentido de giro de los dos motores, tanto de la derecha como izquierda, para que puedan avanzar hacia adelante, atrás y girar a la derecha e izquierda. También para el ángulo de inclinación del robot balance:

```
void Car_Control()
{
 if (pwm_l<0)
  { digitalWrite(DIR_L1, HIGH); digitalWrite(DIR_L2, LOW);
    pwm_l =- pwm_l; }
  else
  { digitalWrite(DIR_L1, LOW); digitalWrite(DIR_L2, HIGH); }

  if (pwm_r<0)
  { digitalWrite(DIR_R1, LOW); digitalWrite(DIR_R2, HIGH);
    pwm_r = -pwm_r; }
  else
  { digitalWrite(DIR_R1, HIGH); digitalWrite(DIR_R2, LOW); }
  if( Angle_Car > 45 || Angle_Car < -45 )
  { pwm_l = 0; pwm_r = 0; }
  analogWrite(PWM_L, pwm_l>255? 255:pwm_l);
  analogWrite(PWM_R, pwm_r>255? 255:pwm_r);
}
```

Ahora, se muestra la subclase para guardar, actualizar y asignar el comando mediante comunicación por radiocontrol (RC) o Bluetooth:

```
void UserComunication()
{
  MySerialEvent();
  if(SerialPacket.m_PackageOK == true)
  {
    SerialPacket.m_PackageOK = false;
    switch(SerialPacket.m_Buffer[4])
    {
      case 0x01:break;
      case 0x02: UpdatePID(); break;
      case 0x03: CarDirection();break;
      case 0x04: SendPID();break;
      case 0x05:
                 SavingData.KA_P = KA_P;
                 SavingData.KA_D = KA_D;
                 SavingData.KP_P = KP_P;
```

```
                SavingData.KP_I = KP_I;
                SavingData.K_Base = K_Base;
                WritePIDintoEEPROM(&SavingData);
                break;
      case 0x06:  break;
      case 0x07:break;
      default:break;
    }
  }
}
```

Actualizamos el cálculo de las variables del controlador PID:

```
void UpdatePID()
{
  unsigned int Upper,Lower;
  double NewPara;
  Upper = SerialPacket.m_Buffer[2];
  Lower = SerialPacket.m_Buffer[1];
  NewPara = (float)(Upper<<8 | Lower)/100.0;
  switch(SerialPacket.m_Buffer[3])
  {
    case 0x01:KA_P = NewPara;
    Serial.print("Get KA_P: \n");
    Serial.println(KA_P);break;
    case 0x02:KA_D = NewPara;
    Serial.print("Get KA_D: \n");
    Serial.println(KA_D);break;
    case 0x03:KP_P = NewPara;
    Serial.print("Get KP_P: \n");
    Serial.println(KP_P);break;
    case 0x04:KP_I = NewPara;
    Serial.print("Get KP_I: \n");
    Serial.println(KP_I);break;

    case 0x05:K_Base = NewPara;
    Serial.print("Get K_Base: \n");
    Serial.println(K_Base);break;
    default:break;
  }
```

A continuación, se estable la lectura de los encoders de los dos motores. Es decir, que tanto el motor de izquierda (L) o derecha (R) permiten desplazar al robot balance hacia arriba o abajo.

```
// móvil arriba es positivo - móvil abajo es negativo
void Encoder_L()
{
  if (digitalRead(SPD_PUL_L))
    Speed_L += 1;
  else
    Speed_L -= 1;
}

// móvil arriba es positivo - móvil abajo es negativo
void Encoder_R()
{
  if (!digitalRead(SPD_PUL_R))
    Speed_R += 1;
  else
    Speed_R -= 1;

  // Serial.print("SPEED_R:    ");
   // Serial.println(Speed_R);
}
```

3.3. Implementación del robot seguidor obstáculo.

En esta sección se realiza la implementación del robot seguidor de obstáculos. Se va utilizar dispositivos, tales como tarjeta controladora de JSumo, dos motores DC 30:1, dos sensores de proximidad a escoger entre los modelos Mz80 y Mr45, batería de lipo y el chasis (placa de acrílico). El objetivo de este robot en una competencia, es que cuando detecta un objeto el robot debe esquivar el obstáculo, caso contrario dependiendo del reglamento es penalizado hasta puede ser descalificado. En la figura 3.24 se muestra el diagrama de conexiones de los dispositivos ya descritos para implementar el robot de obstáculos y posteriormente se presenta la programación en IDE Arduino.

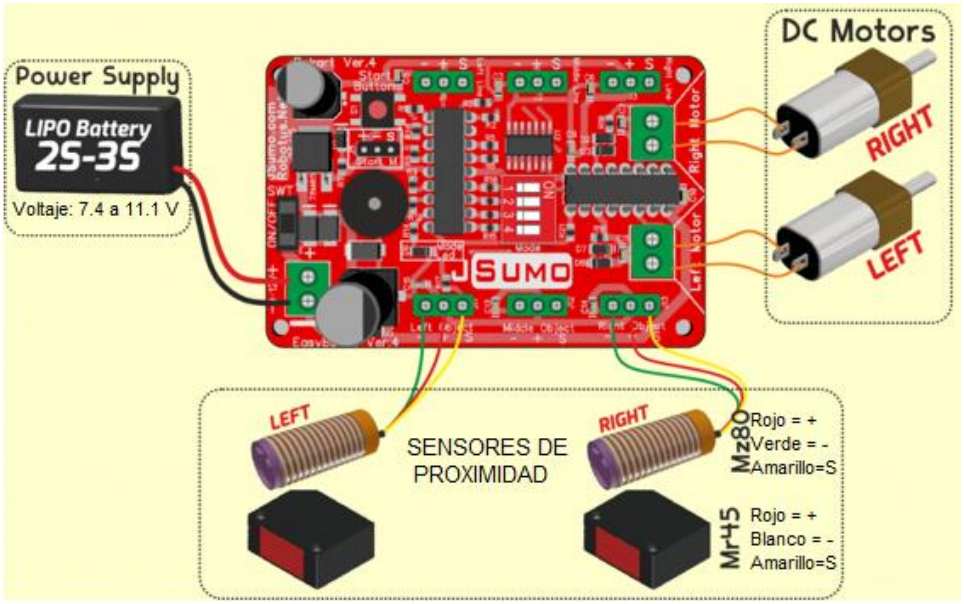


Figura 3. 24: Diagrama de conexiones del robot de obstáculos.
Fuente: (Dede, 2016)

En la figura 3.25 se muestra el diseño del robot similar al propuesto por JSumo.

Figura 3. 25: Implementación del robot de obstáculo.
Fuente: (Dede, 2016)

Los dispositivos infrarrojos (MZ80) detectan los objetos cercanos y luego miden el tiempo que tarda el volver al sensor. Dado que el robot está diseñado

para permanecer lejos de todos los obstáculos directamente en su camino de conducción. La parte más importante de este robot autónomo, es el algoritmo que se utiliza para detectar un posible obstáculo. En la cual Arduino UNO establece que las ruedas giren en direcciones opuestas y continúa la lectura desde el sensor infrarrojo MZ80. Cuando el sensor ya no detecta un obstáculo el robot continúa con su comportamiento normal.

A continuación, se describe el algoritmo de programación desarrollado en IDE Arduino para el robot de obstáculos. Las siguientes líneas de programación permiten activar el módulo de accionamiento del sensor infrarrojo en los pines de la placa Arduino UNO tanto para la izquierda (Left) como la derecha (Right). Esto permite determinar si algo está en el rango permitido por el sensor infrarrojo. También permite seleccionar el pin para el diodo LED.

```
int sensorPinLO = 6;
int sensorPinRO = 5;
int ledPin = 13;
int sensorValue = 0;
```

Después, inicializamos el dispositivo Arduino UNO, el cual será utilizado para la comunicación con los sensores infrarrojos (izquierdo y derecho) y del diodo que indica la comunicación entre los motores y el puente H.

```
void setup () {
Serial.begin(9600);
pinMode (sensorPinLO, INPUT);
pinMode (sensorPinRO, INPUT);
pinMode (ledPin, OUTPUT);
Serial.println("finished setup");
}
```

Finalmente, tenemos la lectura del sensor infrarrojo, para ver si algo está en el lado derecho de nosotros, es decir, que el sensor infrarrojo lee el valor del sensor para ver si algo está en el lado derecho o izquierdo. Deben esperar

cuando el sensor detecta un objeto. Ese tiempo de espera, sería en milisegundos para dejar que el software opere a una velocidad reconocible por el ser humano.

```
void loop () {
int charIn;
if (digitalRead(sensorPinRO) == LOW) {
Serial.println("Sensing Right");
digitalWrite (ledPin, HIGH);
}
if (digitalRead(sensorPinLO) == LOW) {
Serial.println("Sensing Left");
digitalWrite (ledPin, HIGH);
}
delay (1000);
digitalWrite (ledPin, LOW);
}
```

CAPÍTULO 4: CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

- En base al problema planteado al inicio de este proyecto, podemos decir que dudas y aspectos planteados meses atrás hoy son aclarados a cabalidad con la demostración de este proyecto. El mismo que está destinado como material de estudio para promociones venideras de la Universidad Católica de Santiago de Guayaquil, tanto los robots como nuestra fundamentación teórica.
- La fundamentación teórica permitió conocer las herramientas básica con las que se trabajó en el desarrollo de implementación de los tres robots que serán utilizados por el club de robótica “ROBOFET” en próximas competencias de robótica.
- La implementación de los tres robots no fue tan sencilla, aunque se utilizaron dispositivos de hardware abierto (open hardware) de la familia Arduino. La utilidad de estas tarjetas radica en el microcontrolador Atmel (Atmega 328p), así como, la facilidad de integrar periféricos de entrada como, sensores infrarrojos, giroscopio, encoders, entre otros.

4.2. Recomendaciones

- Es importante mencionar que al momento de la manipulación de los materiales con los que se implementan los robots se debe tener mucho cuidado puesto que elementos como microcontroladores son materiales muy sensibles.
- Nuestra Facultad Técnica deberá incentivar a los estudiantes a realizar nuevos proyectos relacionados con la robótica y al mismo tiempo promover la investigación; para así en futuras promociones contar con proyectos innovadores que motiven a los nuevos estudiantes a seguir con el desarrollo de nuevos prototipos.

Referencias Bibliográficas:

Andaluz. (2011). *Modelación, Identificación y Control De Robots Móviles.* Quito.

Ante & Espinel . (2014). *Diseño y Construcción De Un Sistema De Control De Frenado En las Ruedas Posteriores En Un Vehículo Común Sin ABS De Transmisión Manual y En Condiciones Específicas .* Quito.

Arduino. (2016). Arduino - ArduinoBoardUno. Recuperado el 12 de agosto de 2016, a partir de https://www.arduino.cc/en/Main/ArduinoBoardUno

Ávila B., H. (02 de Junio de 2016). *Introducción a la metodología de la investigación.* Obtenido de EUMED - Enciclopedia Virtual: http://www.eumed.net/libros-gratis/2006c/203/

Ayovi & Mero. (2011). *Construcción De Un Sistema De Entrenamiento Para Microcontroladores Atmel Para La EIE-CRI.* Riobamba.

Borobia. (2012). *Desarrollo de Comunicaciones Inalámbricas entre PLC'S.* Logroño.

Condor & Paredes. (2009). *Implementación De Un Sistema De Acceso Electrónico Mediante La Huella Dactilar Y Una Clave De Acceso.* Quito.

Dede, F. (2016). JSumo Sumo Robot Blog - Robot Sumo Projects, Parts. Recuperado a partir de https://jsumo.com/

Delgado. (2011). *Redes Inalámbricas .* 2011: MACRO.

Gonzalez. (2013). *Aplicaciones Prácticas De microcontroladores A Través De La Plataforma De Programación Matlab.* Guayaquil.

Gordillo. (2011). *Construcción De Un Sistema De Seguridad Utilizando Telefonía Móvil GSM.* Quito.

Hernández, R., Fernández, C., & Baptista, P. (2006). *Metodología de la Investigación.* México: Mc-Graw Hill.

InvenSense. (2013). MPU-6050 -DataSheet. Recuperado el 12 de agosto de 2016, a partir de https://store.invensense.com/datasheets/invensense/MPU-6050_DataSheet_V3%204.pdf

Lalama. (2014). *Implementación De Un Robot Móvil Terrestre Omnidireccional Semiautónomo y Telecontrolado a Través De Un Teléfono Inteligente Android.* Guayaquil.

Lindao & Quilambaqui . (2014). *Diseño y Construcción De 2 Robots Sumo Para Las Categorías Pesado y Liviano y Un Robot Seguidor De Línea Modalidad Velocidad.* Guayaquil.

Martínez. (2014). *Sistema Automatizado De Reconocimiento y Manipulación De Objetos Usando Visión Por Computadora y Un Brazo Industrial.* León.

Mikroe. (2016). Arduino UNO Shield. Recuperado el 12 de agosto de 2016, a partir de http://download.mikroe.com/documents/add-on-boards/click-shields/arduino-uno/arduino-uno-click-shield-schematic-v101.pdf

Molina C., C. (05 de Junio de 2016). *Procesadores.* Obtenido de El Microcontrolador vs el Microprocesador: http://www.redtauros.com/Clases/Procesadores/02_Microncotroladores_Microprocesadores.pdf

Nieves C., F. (04 de Junio de 2016). *La Investigación Exploratoria.* Obtenido de Gestiopolis : http://www.gestiopolis.com/la-investigacion-exploratoria/

Pazmiño & Sánchez. (2010). *Diseño y Construcción De Un Prototipo Inalámbrico RF Para el monitoreo De La Seguridad Residencial De Forma Remota Empleando La Internet.* Quito.

Pilaquingua. (2009). *Diseño y Construcción De Una Mano Robótica Controlada Mediante Un Guante Sensorizado.* Quito.

Pino & Vallejo . (2010). *Implementación De Un Prototipo De Robots Cazadores Basados En Un Algoritmo Deliberativo.* Riobamba.

Reyes. (2008). *Microcontroladores Pic Programación En Basic.* Quito: RISPERGRAF.

Rodríguez, D. L. (22 de 07 de 2016). *Arquitectura de computadoras*. Obtenido de Diagrama de una computadora: https://sites.google.com/site/arquitecturadecomputadoras2015/indice/diagrama-de-una-computadora

Rossano. (2009). *Electrónica & Microcontroladores PIC.* Mexico: Users.

Salazar. (2014). *Diseño y Construcción De Prototipo De Un Sistema De Simulación De Presencia Para Un Hogar Mediante Telefonía Móvil .* Quito.

Salguero. (2009). *Desarrollo De Un Piano Digital Con Tecnología Inalámbrica, Para Mejorar Las Destrezas Motoras De Los Niños Del Nivel Preescolar De La Escuela Cristóbal Colón Del Cantón Salcedo.* Ambato.

Samaniego. (2009). *Diseño e Implementación De Un Prototipo De Asistente De Hogar, Caso Practico Aspiradora Autonoma.* Riobamba.

Sánchez & Peña . (2012). *Diseño e Implementación De Un Robot Móvil Publicitario Para La Escuela Superior Politécnica De Chimborazo.* Riobamba .

Trujillo, C.; Candelo, A.; Zipaquira, J. (07 de Junio de 2014). *Arquitectura de computadores*. Obtenido de Arquitectura De La Máquina De Von Newman: http://www.academia.edu/8951985/Arquitectura_de_Computadores_Von_Newman_

Valdéz & Pallas. (2007). *Microcontroladores: Fundamentos Y Aplicaciones Con Pic.* Cataluña: 3Q Editorial.

Vera & Alejandro. (2016). *Diseño e Implementación De Dos Robots Seguidores De Línea Modalidad Velocista y Destreza Para Participaciones En Concursos De Robótica.* Guayaquil.

DECLARACIÓN Y AUTORIZACIÓN

Nosotros, **CAIZA GARCÍA, VÍCTOR DANIEL** con C.C: # 0502609761, y **VERDEZOTO CEDEÑO, CRISTHIAN ENRIQUE** con C.C: # 0926038084 autor del Trabajo de Titulación: **IMPLEMENTACIÓN DE DOS ROBOTS AUTÓNOMOS Y UNO CONTROLADO MEDIANTE TECNOLOGÍA BLUETOOTH PARA LAS CATEGORÍAS SEGUIDOR DE LÍNEA CON OBSTÁCULOS, LABERINTO Y BALANCÍN** previo a la obtención del título de **INGENIERO EN TELECOMUNICACIONES** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaramos tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizamos a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, 13 de Septiembre de 2016

f. _____

Nombre: CAIZA GARCÍA, VÍCTOR DANIEL

C.C: 0502609761

f. _____

Nombre: VERDEZOTO CEDEÑO, CRISTHIAN ENRIQUE

C.C: 0926038084

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA

FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN

TÍTULO Y SUBTÍTULO:	IMPLEMENTACIÓN DE DOS ROBOTS AUTÓNOMOS Y UNO CONTROLADO MEDIANTE TECNOLOGÍA BLUETOOTH PARA LAS CATEGORÍAS SEGUIDOR DE LÍNEA CON OBSTÁCULOS, LABERINTO Y BALANCÍN		
AUTOR(ES)	CAIZA GARCÍA, VÍCTOR DANIEL; VERDEZOTO CEDEÑO, CRISTHIAN ENRIQUE		
REVISOR(ES)/TUTOR(ES)	M. Sc. EDWIN F. PALACIOS MELÉNDEZ		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Facultad de Educación Técnica para el Desarrollo		
CARRERA:	Ingeniería en Telecomunicaciones		
TITULO OBTENIDO:	Ingeniero en Telecomunicaciones		
FECHA DE PUBLICACIÓN:	13 de Septiembre de 2016	**No. DE PÁGINAS:**	94
ÁREAS TEMÁTICAS:	Sistemas Digitales, Microprocesadores, Microcontroladores, Comunicación Inalámbrica.		
PALABRAS CLAVES/ KEYWORDS:	MICROCONTROLADORES, BLUETOOTH, RADIO CONTROL, ROBÓTICA MÓVIL, AUTÓNOMO, ARDUINO.		

RESUMEN/ABSTRACT (150-250 palabras):

Este trabajo de titulación tiene como meta centralizar el estudio de la tecnología robótica móvil, motivando al lector en el área de investigación. Indagar en ámbitos tecnológicos tanto de la parte física como programable (software y hardware) para la implementación de un robot en categoría seguidor de línea con obstáculos, laberinto y balancín. Presentamos a los microcontroladores y su impacto desde años atrás, estándares y procesos que se deben tener en cuenta para la implementación de un robot y como ha sido su impacto en la sociedad. En la fundamentación teórica se observa la información que nos llevara a tener de una manera más clara y concisa que engloba los microcontroladores, memorias, tipos de microcontroladores, tecnologías inalámbricas. Se resalta la tecnología autónoma y de bluetooth necesarias para nuestros robots; revisamos de manera general, robots dentro de estas categorías que han sido importantes para la investigación en el mundo. En este trabajo detallamos a cabalidad todos los elementos sobresalientes requeridos para la implementación de los robots presentados, su programación y diseño como tal de las tres categorías.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:** +593-9-80901602 +593-9-58879722	E-mail: victorcaiza_g@hotmail.com cv_bsc@hotmail.com
CONTACTO CON LA INSTITUCIÓN:	**Nombre:** Palacios Meléndez Edwin Fernando	
	Teléfono: +593-9-68366762	
COORDINADOR DEL PROCESO DE UTE	**E-mail:** edwin.palacios@cu.ucsg.edu.ec	

SECCIÓN PARA USO DE BIBLIOTECA	
No. DE REGISTRO (en base a datos):	
No. DE CLASIFICACIÓN:	
DIRECCIÓN URL (tesis en la web):	