UNIVERSIDAD NACIONAL DE CHIMBORAZO

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

"Trabajo de grado previo a la obtención del Título de Ingeniero en Electrónica Y Telecomunicaciones"

TRABAJO DE GRADUACION

Título del proyecto:

DISEÑO E IMPLEMENTACIÓN DE UN ROBOT PERSONAL TRANSPORTADOR DE OBJETOS

Autor: (es)
Edgar Vinicio Carrión Sampedro
Cristian Rubén Hinojosa Luzuriaga

Director:
Ing. Fabián Gunsha

Riobamba – Ecuador

2016

Los miembros del Tribunal de Graduación del proyecto de investigación de título: **DISEÑO E IMPLEMENTACIÓN DE UN ROBOT PERSONAL TRANSPORTADOR DE OBJETOS** presentado por: **Edgar Vinicio Carrión Sampedro y Cristian Rubén Hinojosa Luzuriaga** y dirigida por: **Ingeniero Fabián Gunsha.**

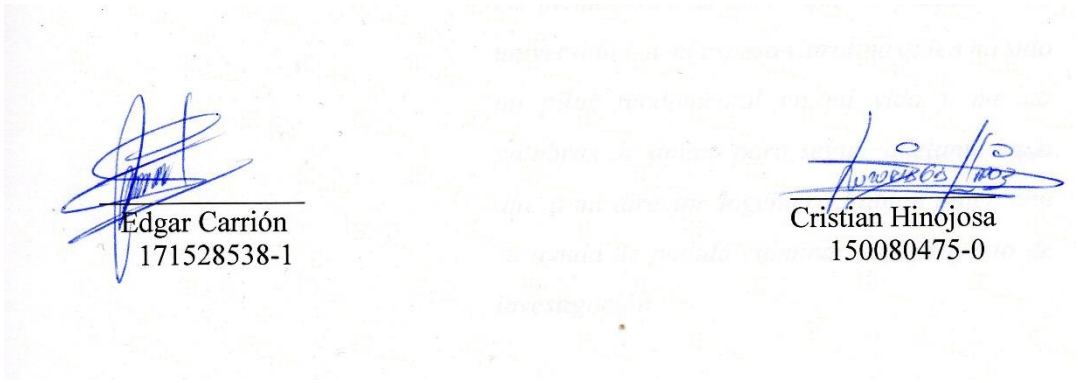
Una vez escuchada la defensa oral y revisado el informe final del proyecto de investigación con fines de graduación escrito en el cual se ha constatado el cumplimiento de las observaciones realizadas, remite la presente para uso y custodia en la biblioteca de la Facultad de Ingeniería de la UNACH

Para constancia de lo expuesto firman:

Ing. Paulina Vélez	
Presidente de Tribunal	**Firma**
Ing. Fabián Gunsha	
Director de Tesis	**Firma**
Ing. Geovanny Cuzco	
Miembro del Tribunal	**Firma**

AUTORÍA DE LA INVESTIGACIÓN

La responsabilidad del contenido de este Proyecto de Graduación, nos corresponde exclusivamente a: Edgar Carrión y Cristian Hinojosa como autores, e Ing. Fabián Gunsha en calidad de Director del Proyecto; y el patrimonio intelectual de la misma a la Universidad Nacional De Chimborazo.



Edgar Carrión
171528538-1

Cristian Hinojosa
150080475-0

AGRADECIMIENTO

Agradezco a Dios por darme la vida, a mis padres Alicia y Patricio quienes me han apoyado en todo el proceso académico, a Francisco y Andrea mis hermanos que han estado conmigo, a Cecilia quien me ha visto en los momentos más duros que he pasado en la universidad, a mi esposa Carolina quien ha sido un pilar fundamental en mi vida y me da palabras de ánimo para seguir adelante cada día, a mi director Ingeniero Fabián quien con su ayuda he podido culminar este proyecto de investigación

Edgar Carrión

AGRADECIMIENTO

Agradezco a dios por darme la vida y siempre cuidarme en mi camino ante cualquier adversidad.

A mi familia, en especial a mi madre Mercy, que siempre ha estado presente en cada momento de necesidad por sus consejos y apoyo y su gran esfuerzo por darme una buena educación.

A mi hermana Karina que sabido ser de gran ayuda e momentos de suma importancia para mi carrera universitaria.

Agradezco a mi tutor de tesis por las sugerencias efectuadas a lo largo del desarrollo exitoso del tema y a mi compañero Edgar Carrión por la paciencia y su amistad sincera en toda mi carrera universitaria.

Cristian Hinojosa

DEDICATORIA

Dedico este trabajo a mis padres Alicia y Patricio, que por su esfuerzo que han realizado para que yo pueda seguir esta carrera.

A mis hermanos Andrea y Francisco por el apoyo que me han brindado.

A mi abuelita Carmen y prima Cecilia que siempre me brindaron el apoyo comprensión y cariño.

A mi esposa Carolina que siempre ha estado con migo en los buenos y malos momento, dándome su apoyo y cariño incondicionalmente.

Edgar Carrión

DEDICATORIA

Dedico este trabajo a mi madre como muestra de agradecimiento por nunca rendirse en mi desarrollo como profesional por inculcarme buenos valores y siempre esforzarse por sus hijos, a mi hermana por el apoyo incondicional y aliento a no rendirme con mis metas propuestas.

Cristian Hinojosa

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE TABLAS

RESUMEN

El presente trabajo de investigación, es el diseño e implementación de un robot personal transportador de objetos con la finalidad de seguir a una persona específica, consta de un diseño mecánico con medidas estándar para gradas, 15x30cm (alto x largo), con un sistema de locomoción de pistas de deslizamiento (oruga), para el desplazamiento entre los vértices de cada grada.

El sistema es controlado por arduino Mega, que se comunica con la cámara Pixy para la detección del objeto, además posee tres sensores ultrasónicos, un frontal para la distancia con la persona y los otros para la evasión de obstáculos a los costados del robot.

A medida que se utilizó la cámara Pixy se observa que no pierde de vista a la persona que porta el objeto a seguir, pero con algunos inconvenientes tales como: la confusión de un mismo color, los cambios de luz demasiado bruscos, problemas de reconocimiento a grandes distancias y objetos pequeños. Además de crearse conflictos cuando existe demasiadas personas que poseen la gama de color del objeto a seguir. Provocando que el sistema tenga inconveniente, en consecuencia, el diseño e implementación del robot personal de objetos demuestra ser un sistema electrónico aceptable para el cumplimiento del trabajo de investigación.

UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
CENTRO DE IDIOMAS INSTITUCIONAL

Ms. Janeth Caisaguano **10 de Agosto del 2016**

ABSTRACT

The present research work designs and implement a personal object transporter robot with the purpose to follow a specific person. It consists of a mechanical design with standard measures for bleachers, 15x30cm (height x length), with a system of locomotion of sliding tracks (Track), to travel between the vertices of each harrow.

The system is controlled by the Arduino Mega, which communicates with the camera Pixy for the detection of the object, also it has three ultrasonic sensors, a front for the distance with the person and the other for the evasion of obstacles to the sides of the robot.

When we used the Pixy camera noted that it did not lose the sight of the person who carried the object to follow, but with some drawbacks such as: the confusion of a same color, changes in abrupt light, problems of recognition in great distances and small objects.

In addition there were conflicts when there is too many people that have the color range of the object to follow causing the system has inconvenience, according to the design and implementation of a personal object transporter robot testing to be an acceptable compliance electronic system research work.

CENTRO DE IDIOMAS INSTITUCIONAL
UNIVERSIDAD NACIONAL DE CHIMBORAZO
RIOBAMBA - ECUADOR
DIRECCION

INTRODUCCIÓN

La tecnología es una característica que el ser humano posee siendo la capacidad para construir a partir de materias primas, una gran diversidad de objetos, máquinas y herramientas, logrando conseguir una vida más segura y cómoda.

La tecnología abarca tanto el proceso de creación como los resultados, dependiendo los campos tenemos múltiples ramas o tecnologías: mecánica, materiales, calor y frío, eléctrica, electrónica, química, bioquímica, nuclear, telecomunicaciones, etc.

Los robots autónomos (RA) son sistemas independientes que operan en entornos abierto o cerrados sin la necesidad de la manipulación humana. La propiedad fundamental de los RA es la configuración dinámica de poder resolver distintas tareas de acuerdo a las características de su entorno. “Hacemos énfasis en que son sistemas completos que perciben y actúan en entornos dinámicos y parcialmente impredecibles, coordinando interoperaciones entre capacidades complementarias de sus componentes. La funcionalidad de los RA es muy amplia y variada desde algunos RA que trabajan en entornos inhabitables, a otros que asisten a gente discapacitada.” (Gonzales de Rivera, 2016).

El presente proyecto expone el desarrollo de un robot transportador de objetos personal para movilización de objetos no mayores a 50 libras, la información obtenida mediante la recopilación de datos y resultados del robot transportador, será realiza y será obtenida por la cámara Pixy y los sensores distribuidos en el robot, su desplazamiento será realizado en superficies planas y con inclinación no mayor a 30°.

CAPÍTULO I

1. FUNDAMENTACIÓN TEÓRICA

1.1. Robótica

La robótica es la rama de la tecnología que se dedica al diseño, construcción, operación, disposición estructural, manufactura y aplicación de los robots. La robótica combina diversas disciplinas como son: la mecánica, la electrónica, la informática, la inteligencia artificial, la ingeniería de control y la física. Otras áreas importantes en robótica son el álgebra, los autómatas programables y las máquinas de estados.

En la actualidad los robots son una obra de la ingeniería concebidas para producir bienes y servicios. El desarrollo de las máquinas se encuentra influido por el progreso tecnológico que existe día a día. Según (Baratune Ollero, 2001) "de esta forma se pasa de máquinas que tienen como objetivo exclusivo la amplificación de la potencia muscular del hombre, sustituyéndolo en su trabajo físico, a máquinas o instrumentos que también son capaces de procesar información, complementando, o incluso sustituyendo al hombre en algunas actividades intelectuales"(p.1).

De acuerdo a (Baratune Ollero, 2001) "el término robot aparece por primera vez en 1921, en la obra teatral R.U.R (Rossum's Universal Robots) del novelista y autor dramático checo Karel Capek en cuyo idioma la palabra "robota" significa fuerza del trabajo o servidumbre" (p.2). Pero ¿qué es un robot?, al hacer está pregunta siempre imaginamos las películas, robots complejos capaces de realizar tareas sofisticadas, el poder hablar y actuar como un ser humano normal. Pero en realidad, en la actualidad debido a nuestra tecnología se está muy lejos de poder conseguir robots de este tipo.

En conclusión, un robot es una máquina diseñada para realizar una acción o un trabajo de forma controlada o automática con el fin de hacer nuestra vida más cómoda y sencilla.

1.2. Clases De Robots

1.2.1. Robot móvil

El robot móvil es una máquina con la capacidad de trasladarse en cualquier ambiente dado o previamente programado, esta clase de robots son muy utilizados para realizar diferentes tipos de trabajo, requieren tener con anterioridad trayectorias establecidas en su programación para desenvolverse en un ambiente determinado como lo indica la Figura 1.

1.2.2. Robot industrial

El robot industrial tiene diferentes movimientos programados los cuales pueden ser modificados por personas desde un ordenador influyendo en su programación, estos poseen la destreza de manipular piezas y realizar diferentes tareas con movimientos secuenciales, es comúnmente usado en la industria para la realización de procesos de fabricación. Los modelos más usados en la actualidad son los llamados brazos robóticos debido a su forma como lo indica la Figura 2.

1.2.3. Robot humanoide

El robot humanoide propiamente dicho adquiere su nombre de la similitud que posee con el ser humano, son fabricados con rasgos físicos, que se asemejan a los nuestros. Según (Baratune Ollero, 2001) manifiesta que esta clase de robot tiene la capacidad de poder interactuar con las personas y almacenar la información recopilada del entorno o la persona con la cual interactúa, realiza estas acciones mediante los diferentes tipos de sensores que están presentes a lo largo de su estructura, como lo indica la Figura 3.

1.2.4. Robot teleoperador

El robot teleoperador es diseñado para recibir órdenes directas, controladas por operadores o controladores en tiempo real, de manera directa o mediante un ordenador. Capturan el entorno en el que se desenvuelven a través de sensores que le permiten su

comunicación con su operador, pueden ser controlados a grandes distancias y con gran precisión haciéndolo muy útil para situaciones de peligro como el desarmar bombas o adentrarse en lugares contaminados o ambientes extremos para el hombre. La Figura 4 muestra un robot teleoperado capaz de desactivar bombas.

Figura 1. Robot explorador del tipo móvil

Fuente: Carpio, M. (2016). Robot explorador del tipo móvil. [Figura]. Recuperado de http://www.monografias.com/trabajos93/inteligencia-artificial-aplicada-robotica/inteligencia-artificial-aplicada-robotica.shtml

Figura 2. Brazo robótico de tipo Industrial

Fuente: Carpio, M. (2016). Brazo robótico del tipo Industrial. [Figura]. Recuperado de http://www.monografias.com/trabajos93/inteligencia-artificial-aplicada-robotica/inteligencia-artificial-aplicada-robotica.shtml

Figura 3. ASIMO Robot del tipo humanoide

Fuente: Carpio, M. (2016). ASIMO robot del tipo humanoide. [Figura]. Recuperado de http://www.monografias.com/trabajos93/inteligencia-artificial-aplicada-robotica/inteligencia-artificial-aplicada-robotica.shtml

Figura 4. Robot desactiva bombas del tipo Teleoperador

Fuente: Carpio, M. (2016). Robot desactiva bombas del tipo teleoperador. [Figura]. Recuperado de http://www.monografias.com/trabajos93/inteligencia-artificial-aplicada-robotica/inteligencia-artificial-aplicada-robotica.shtml

1.2.5. Robot autónomo

Uno de los principales retos de la robótica móvil consiste en la generación del comportamiento autónomo. De acuerdo a (Constain Arroyave, 2012) sugiere que cuando el robot conoce su objetivo o lugar de destino, debe tomar la mejor decisión y alcanzarlo (p.30).

Este concepto es destacado como navegación, siendo uno de los principales temas de investigación en robótica móvil. Pero existe un problema la navegación no puede ser posible sin cuatro bases principales como son: percepción, localización, planificación y mapeo.

- **Percepción:** Consiste en la información que adquiere el robot sobre el entorno que le rodea, para la percepción emplea y manipula señales de diferentes sensores, tomando valores reales de cada espacio que está en el área que se desempeña.

- **Localización:** Hace mención a la posición del robot en el plano. Quiere decir, que debe estar al corriente de la posición de todos los puntos que estructuran el plano donde se desempeña. El problema para resolver la localización será resuelto si la posición del robot lo proporciona un sistema GPS. No obstante, la precisión de estos dispositivos tiene un error de unos cuantos metros, al mismo tiempo, es importante que el robot este ubicado en campo abierto donde no se pierda la señal administrada por del dispositivo GPS. Conjuntamente si el robot interactúa con personas, el robot debe poder identificarlos y definir su posición relativa respecto a ellos.

- **Mapeo:** Hace memoria de los datos del entorno sobre el cual se transporta el robot, recolectando previamente en su memoria. Esto facilita que el robot pueda localizarse y realizar adecuadamente la planificación de su trayectoria, evadiendo con mayor pericia los obstáculos y determinando la disposición del espacio.

- **Planificación:** Para alcanzar el punto planteado, partiendo de la ubicación real del robot. Este debe optar por la ruta o trayecto más eficiente para llegar al lugar establecido, evitando obstáculos, considerando las superficies de su lugar de trabajo. Así, conociendo el plano y el juicio del punto de llegada, el planteamiento de la trayectoria comprende en identificar la ruta que hará que el robot alcance el punto de llegada cuando la orden sea ejecutada.

1.3. Estructura general de un robot móvil

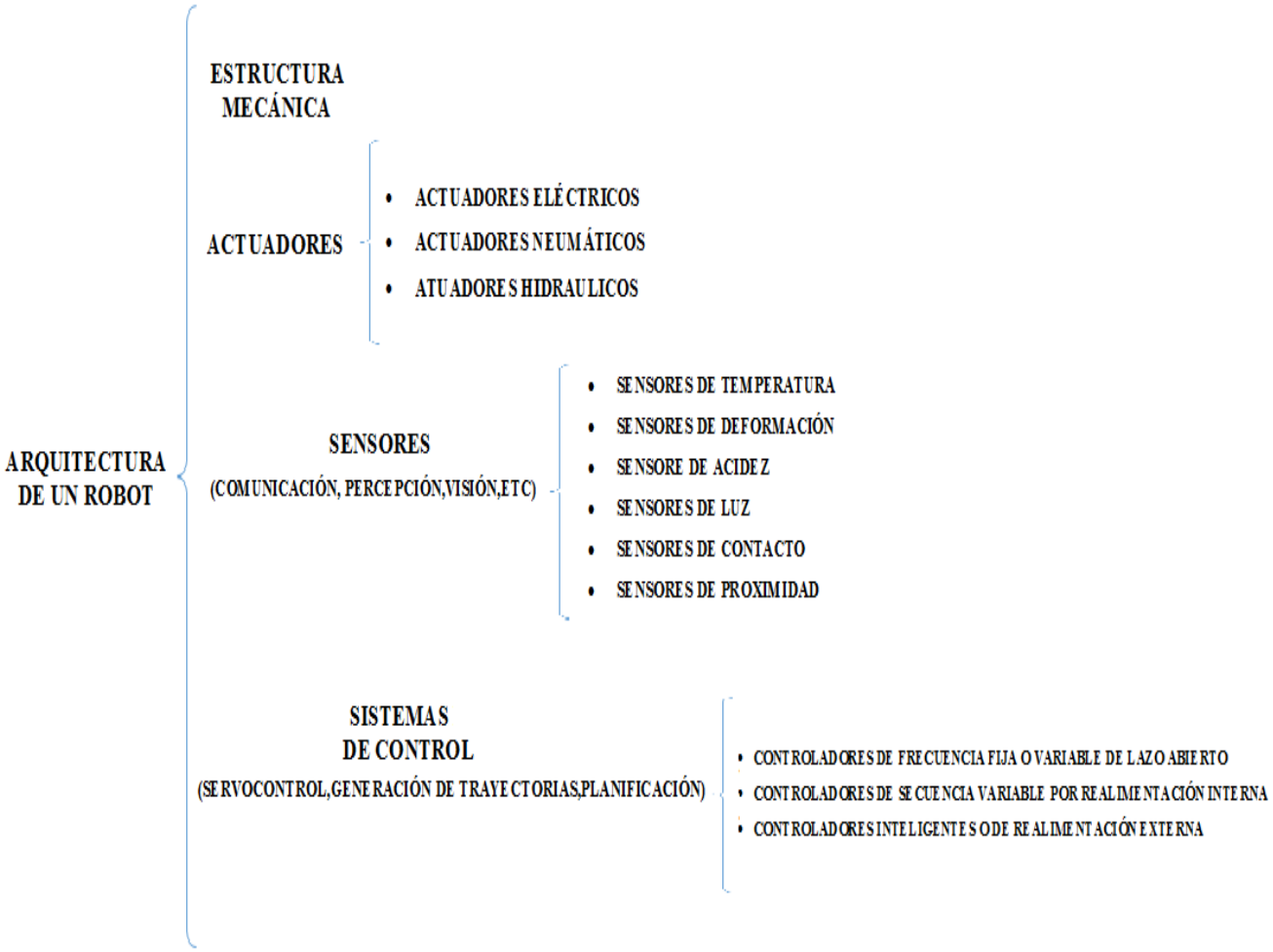


Figura 5. Arquitectura de un Robot

Fuente: Autores

En un robot la estructura puede fácilmente discrepar 4 elementos bien diferenciados:

- **La estructura:** Es el esqueleto del robot y está establecido por los mecanismos y articulaciones que consienten su movimiento.
- **Los actuadores:** Aquellos elementos que tiene la función al igual que un músculo estos se encargan de abastecer la fuerza necesaria para mover la estructura del robot.
- **Los sensores:** Son los sentidos del robot, es decir constituyen un conjunto de elementos, que le permiten estar al tanto el cambio del mundo que le rodea y la posición exacta de sus componentes.
- **Los sistemas de control:** Es la parte principal del robot siendo el cerebro del autómata en función de las tareas que desempeña el robot.

1.4. Sensor

Un sensor es un dispositivo eléctrico y/o mecánico que convierte las magnitudes físicas como la luz, magnetismo, presión, etc. En valores que puedan ser medibles usualmente en señales eléctricas.

La señal eléctrica es modificada por un sistema de acondicionamiento de señal, cuya salida es un voltaje. El sensor dispone de una circuitería que transforma y/o amplifica la tensión de salida, la cual pasa a un conversor A/D, conectado a una PC. El convertidor A/D transforma la señal de tensión continua en una señal discreta.

Las principales características que poseen los sensores:

- **Rango de medida:** El sensor puede captar la magnitud o el valor de entrada o de salida siendo el rango de medida los valores mínimo y máximo captados.
- **Precisión:** Es el error de medida máximo esperado, entre la salida real comparada con la salida teórica.
- **Linealidad:** Es lineal, si existe una constante de proporcionalidad única que relaciona los incrementos de señal de salida con los correspondientes incrementos de señal de entrada, en todo el rango de medida.
- **Sensibilidad:** Es la razón de cambio de salida frente a los cambios de entrada.

- **Resolución:** Es la cantidad mínima de variación de la variable de entrada que puede detectar a la salida.
- **Deriva:** Es la variación de la señal de salida, en un periodo de tiempo mientras conserva la variable y las condiciones ambientales, como la humedad, la temperatura y otras como el envejecimiento (oxidación, desgaste, etc.) del sensor.
- **Repetitividad:** Es la capacidad de reproducir varias veces la misma medida de una variable.

1.4.1. Cámara Pixy (CMUcam5)

Pixy es una asociación entre el Instituto de Robótica de Carnegie Mellon y laboratorios Charmed. Pixy viene de una larga línea de CMUcams, pero Pixy consiguió su comienzo real como una campaña de Kickstarter.

Pixy CMUCam5 es un sensor de imagen que posee un procesador de gran alcance que posible de programar para enviar sólo la información que se está buscando para que su microcontrolador no se siente agobiado por la cantidad de datos.

Está cámara o sensor de imagen utiliza tono y la saturación como su principal medio de detección de imágenes en lugar de la normal RGB. Lo que significa que la iluminación o exposición no afectarán a la detección de un elemento de la Pixy CMUCam, problema frustrante para muchos sensores de imagen. Siendo una gran mejora respecto a versiones anteriores de la Pixy CMUCam, proporcionando así una mayor flexibilidad cuando se trata de cambios en la iluminación y la exposición.

Además puede recordar siete firmas de color diferentes, encontrar cientos de objetos a la vez. También tiene la facilidad de configurarla para que sólo le envíe las imágenes que específicamente se va a capturar o buscar. Es fácil y rápido y tiene una aplicación de código abierto llamado PixyMon.

Actualmente la Cámara Pixy (CMUcam5) que se utiliza en este proyecto cuenta con soporte para los siguientes microcontroladores.

- Arduino
- Raspberry Pi
- BeagleBone Black

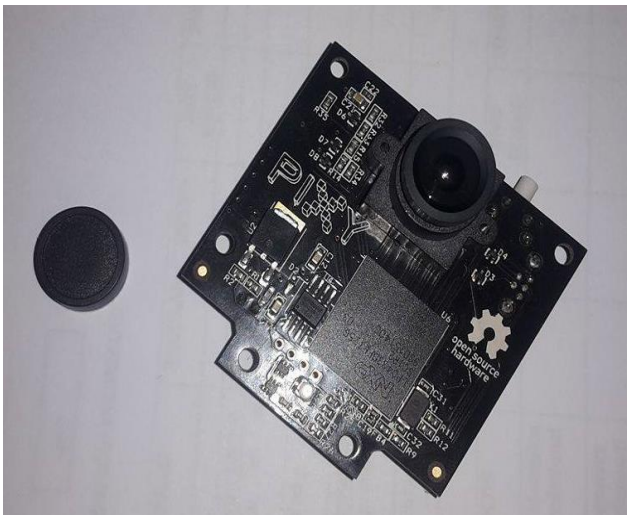


Figura 6. Cámara Pixy (CMUcam5)

Fuente: Autores

Pixy posee tres métodos de comunicación los cuales son:

- **Serial:** Esto incluye SPI, I2C y las interfaces UART. Las interfaces en serie todos utilizan un protocolo simplificado con un código y la memoria de pequeño tamaño. El código es sencillo de portar a diferentes microcontroladores. Este es el método que se utiliza para la comunicación con Arduino. Utilizando el protocolo serial, Pixy envía información completa acerca de lo que se detecta.

- **USB:** La interfaz USB está diseñado para microcontroladores que tienen más recursos de memoria (RAM y flash). El código para el protocolo USB tiene una huella de memoria más grande. Este es el método que se utiliza para hablar con Raspberry Pi y BeagleBone Black, y es utilizado por PixyMon para transmitir vídeo en directo y leer la información de configuración / escritura.

- **Analógico / digital:** esta es la interfaz más simple y ni siquiera tiene un protocolo.

1.4.2. Sensor Ultrasónico HC-SR04

HC-SR04 es un dispositivo emisor de ultrasonidos, el receptor y los circuitos de control. Cuando Trig que envía una serie de impulsos ultrasónicos que trabaja 40KHz y recibe eco de un objeto. La distancia entre la unidad y el objeto se calcula midiendo el tiempo de viaje del sonido y la salida como la anchura de un pulso TTL.

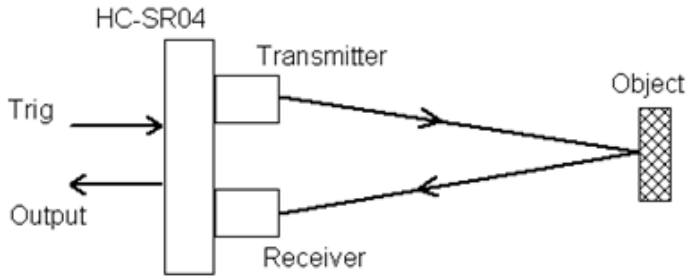


Figura 7. Sensor Ultrasónico HC-SR04

Fuente: HC-SR04 Ultrasónic Range Finder Manual. (2016). [Figura]. Recuperado de http://www.accudiy.com/download/HC-SR04_Manual.pdf

Para medir la distancia que necesita para generar una señal de disparo y conducirlo a la clavija de entrada Trig. La señal leve trig debe cumplir con los requisitos de nivel TTL (es decir, alto nivel de> 2,4 V, bajo nivel <0.8V) y su anchura debe ser mayor que 10 us. Al mismo tiempo que necesita para controlar el pin de salida mediante la

medición de la anchura de impulso de la señal de salida. La distancia detectada se puede calcular mediante la siguiente fórmula.

$$Distancia(m) = \frac{Ancho\ de\ Pulso(s) * Velocidad\ del\ sonido(m/s)}{2}$$

Donde el ancho de pulso está en la unidad de segundo y sonido es la velocidad en la unidad de metro / segundo. Normalmente, la velocidad del sonido es de 340m / s bajo temperatura ambiente.

En la Figura 8, podemos observar el flanco de disparo del ultrasónico HC-SR04

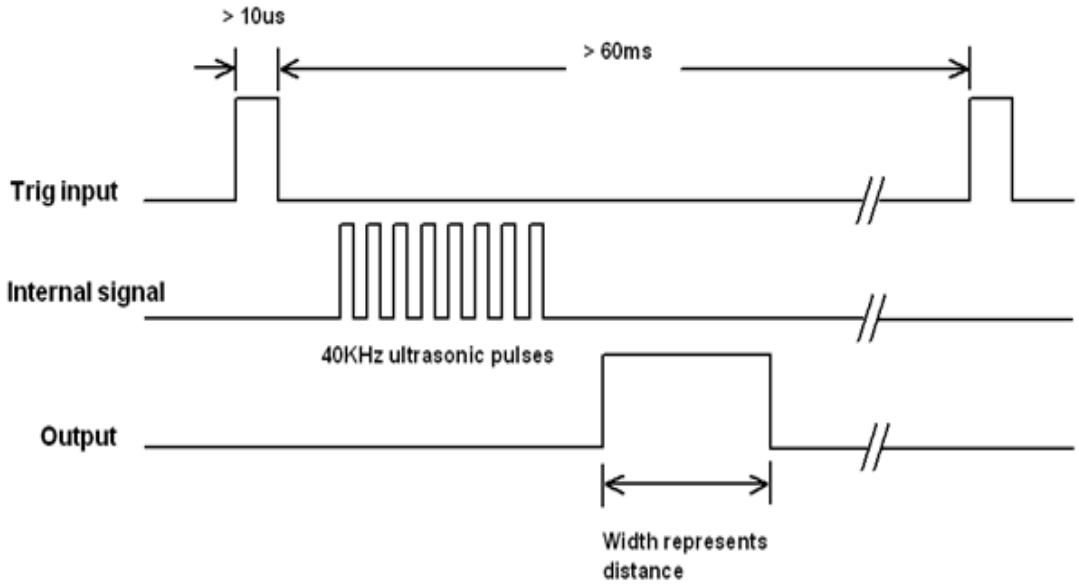


Figura 8. Flanco de disparo del ultrasónico

Fuente: HC-SR04 Ultrasónic Range Finder Manual. (2016). [Figura]. Recuperado de http://www.accudiy.com/download/HC-SR04_Manual.pdf

Parámetros	Especificación
Voltaje de Operación	+5VDC
Corriente de operación	15mA
Frecuencia de operación	40KHz
Máxima distancia	400cm
Mínima distancia	2cm
Angulo de detección	15°
Resolución	0.3cm
Señal de entrada Trig	> 10us pulso TTL
Señal de salida	Pulso TTL con la distancia que representa ancho

Tabla 1. Especificaciones del sensor HC-SR04

Fuente: Autores

1.5. Motores con reductor

Figura 9. MOTOR SPUR GEAR

Fuente: Autores

Los motores con reductor son motores cuyo objetivo principal es la reducción de velocidad por medio de engranajes o caja reductora a cambio de conseguir aumentar su fuerza, cualquier máquina que su movimiento sea generado por un motor necesita que la velocidad de dicho motor se adapte a la velocidad necesaria para el buen funcionamiento que este necesite o fuera diseñado.

Logrando así la reducción en sus llantas con la finalidad de conseguir más fuerza obteniendo que su adherencia en superficies lisas sea la más adecuada.

1.6. Sabertooth 2x12

Es un controlador de motor de doble optimizado, puede suministrar dos motores DC con un máximo de 12 Amperios cada uno corrientes de pico de 25 Amperios son alcanzables por un par de segundos.

Es el primer controlador del motor síncrono regenerativa de su clase, significa que sus baterías se recargan cada vez que el robot frena o revierte su giro.

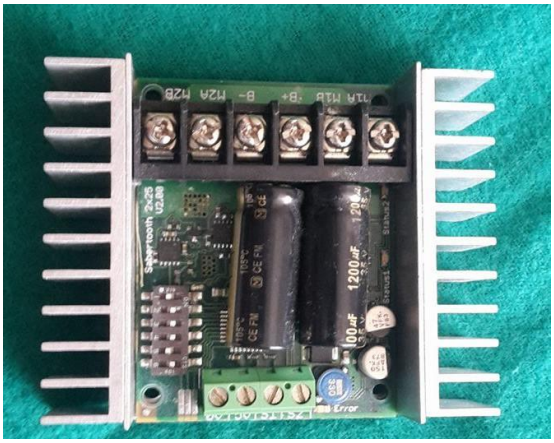


Figura 10. Sabertooth 2x12

Fuente: Autores

1.7. Tracción del robot utilizando ruedas.

1.7.1 Ackerman

Está configuración normalmente se encuentra y vemos en los vehículos convencionales, siendo este sistema de locomoción una configuración muy probada y estable. Su diseño se basa en una estructura de cuatro ruedas colocadas en dos ejes, donde sólo las dos ruedas delanteras permiten un giro sobre el eje.

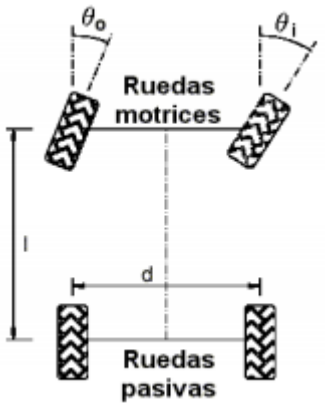


Figura 11. Configuración tipo Ackerman

Fuente: Suarez J. (2016). Microbot con configuración tipo Ackerman. [Figura]. Obtenido de http://eii.unex.es/profesores/jisuarez/descargas/otras/Robotmov.pdf

1.7.2 Triciclo

Este sistema de locomoción utiliza solo 3 ruedas aportando con poca estabilidad en el sistema en terrenos difíciles. Su rueda delantera sirve para la tracción tanto como para el direccionamiento. El centro de gravedad tiende a desplazarse cuando el vehículo se desplaza por una pendiente, causando así la perdida de tracción.

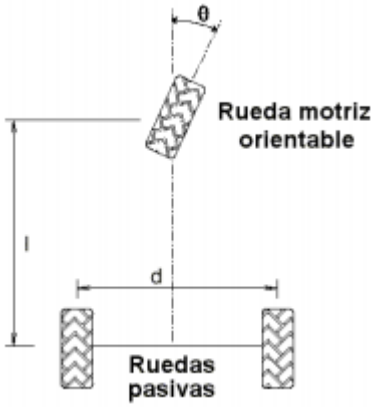


Figura 12. Configuración Tipo Triciclo-Clásico

Fuente: Suarez J. (2016). Microbot con configuración en Triciclo. [Figura]. Obtenido de http://eii.unex.es/profesores/jisuarez/descargas/otras/Robotmov.pdf

1.7.3 Pistas de deslizamiento

Este sistema de locomoción podemos encontrarlo en los tanques. Conocidos como vehículos de tipo oruga, basándose en el direccionamiento por diferencia de velocidades. De esta manera, al realizar un giro aumentamos la velocidad de una de las pistas laterales para producir el giro en el vehículo. Es una configuración diseñada para el trabajo en terrenos irregulares y normalmente se suelen incluir ruedas (skid-steering).

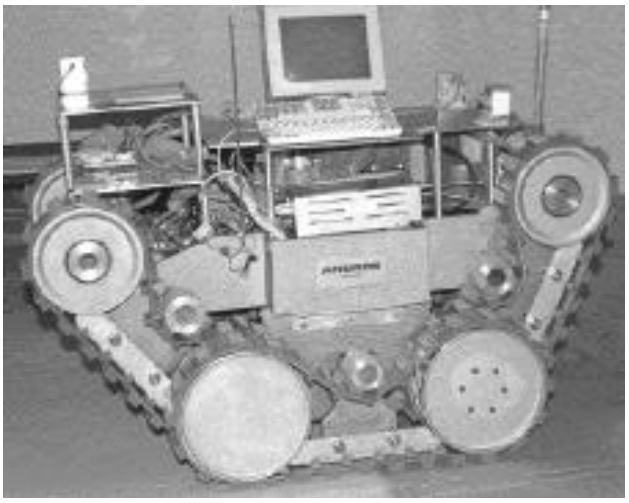


Figura 13. Locomoción con Pistas de Deslizamiento

Fuente: Suarez J. (2016).Microbot oruga Andros V creado por la Universidad de Michigan. [Figura]. Obtenido de http://eii.unex.es/profesores/jisuarez/descargas/otras/Robotmov.pdf

1.7.4 Direccionamiento diferencial

De acuerdo a (Baratune Ollero, 2001), el direccionamiento viene dado por la diferencia de velocidades de las ruedas laterales y la tracción se consigue con estas mismas ruedas. Adicionalmente, existen una o más ruedas para su soporte. Siendo así la configuración más frecuente de robots para trabajar interiores.

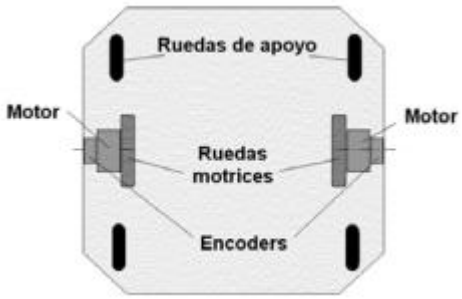


Figura 14. Configuración tipo Dirección diferencial

Fuente: Suarez J. (2016). Microbot con configuración diferencial. [Figura]. Obtenido de http://eii.unex.es/profesores/jisuarez/descargas/otras/Robotmov.pdf

1.8. Microcontroladores

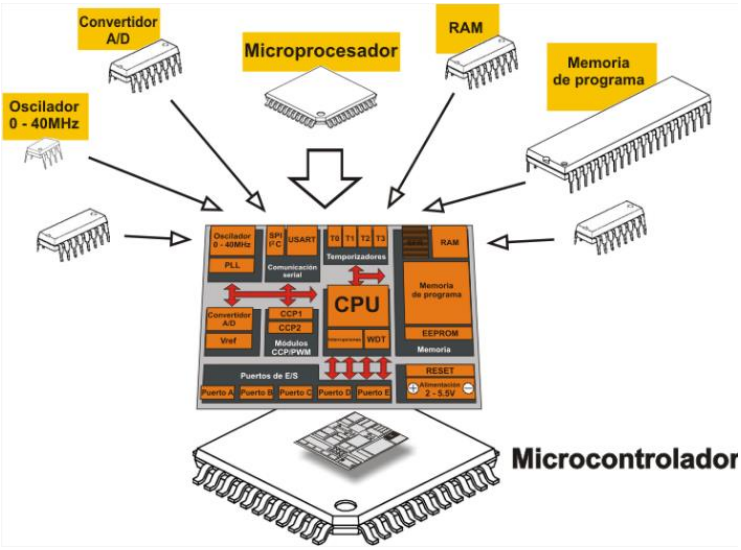


Figura 15. Microcontrolador

Fuente: Mundo de los microcontroladores. (2016). [Figura]. Obtenido de http://learn.mikroe.com/ebooks/microcontroladorespicc/chapter/introduccion-al-mundo-de-los-microcontroladores/

Los microcontroladores son circuitos integrados que generalmente deben ser programados, capaces de ejecutar dicha programación que fueron grabadas en su

memoria. Su composición viene dada por varios bloques funcionales, que cumplen una tarea específica, son capaces de operar uno o más procesos, por lo general los microcontroladores están basados en la arquitectura de Harvard, que consiste en dispositivos de almacenamiento separados (memoria de programa y memoria de datos).

Un microcontrolador posee en su interior la arquitectura de una computadora que son tres principales unidades funcionales, las cuales son: unidad central de procesamiento, memoria y periféricos de entrada y salida.

1.8.1. Arquitectura interna de los microcontroladores

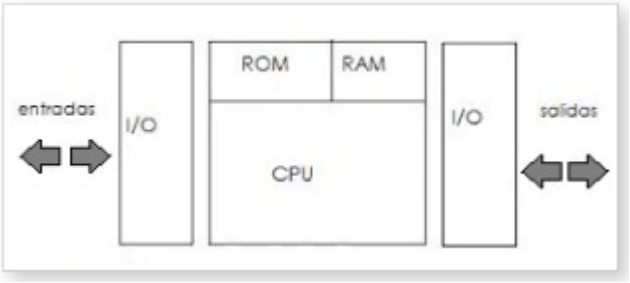


Figura 16. Arquitectura interna de un microcontrolador

Fuente: Introducción y Arquitectura de microcontroladores. (2016). [Figura]. Obtenido de https://microcontroladoressesv.wordpress.com/arquitectura-de-los-microcontroladores/

- **CPU (unidad central de proceso):**

  El CPU es el núcleo del microcontrolador, este se encarga de ejecutar las instrucciones que se encuentran almacenadas en la memoria. Es lo que habitualmente conocemos como procesador o microprocesador, término que puede ser confundido con el de microcontrolador. Tomando en cuenta que ambos términos no son lo mismo, el microprocesador es considerado una parte de un microcontrolador y sin este sería no sería de utilidad, por el contrario un microcontrolador es un sistema completo que puede desarrollar de manera autónoma cualquier labor.

- **Memoria:**

  Son los diferentes componentes que se utiliza para almacenar la información durante un periodo determinado de tiempo. Para la ejecución del programa se necesita el código y los diferentes datos para la ejecución del mismo. Por eso la memoria comprende en memoria de programa y memoria de datos. La memoria de programa comprende en la memoria ROM (Read-Only Memory), EPROM (Erasable Programmable Read Only Memory), EEPROM (Electrical Erasable Programmable Read Only Memory).La memoria de datos que es la memoria RAM (Random Access Memory) destinada al almacenamiento de información temporal.

- **Unidades de entrada/salida:**

  Como ya lo mencionan las unidades de entrada/salida son los sistemas que emplea el microcontrolador para lograr comunicarse con el exterior. Dicho así los dispositivos de entrada permitirán introducir información en el microcontrolador y los dispositivos de salida permitirán enviar información desde el microcontrolador hacia el exterior.

1.9. PixyMon

PixyMon es una aplicación que permite configurar Pixy y poder ver lo que estamos observando con la cámara Pixy. Se puede ejecutar en varias plataformas diferentes, incluyendo Windows, MacOS y Linux, así como otros sistemas integrados más pequeños como Raspberry Pi y BeagleBone Black.

- **Botones:** Se encuentra ubicada en la parte superior de la ventana principal permitiendo realizar las configuraciones para el video.
- **Ventana de vídeo:** En la ventana de video es donde PixyMon hace varios tipos de vídeo en bruto o procesado.
- **Comando / estado:** Aquí es donde se muestran los mensajes de estado y donde los comandos de Pixy se pueden introducir.

Aquí está un detalle de los botones:

Figura 17. Botones de PixyMon

Fuente: Botones de PixyMon (2016). [Figura]. Obtenido de http://www.cmucam.org/projects/cmucam5/wiki/PixyMon_Overview

- **Detener / reanudar:** Al pulsar este botón se detiene el vídeo que está siendo generado en la ventana de vídeo. Esto es útil cuando se agarra un marco o escribir comandos en la ventana de comandos / status. Presione este botón de nuevo para reanudar el vídeo.
- **Programa por defecto:** al pulsar este botón se ejecuta el programa por defecto, que es el programa que se ejecuta cuando los poderes de Pixy arriba. Es normalmente el programa que realiza todo el procesamiento de Pixy y emite los resultados (objetos detectados por ejemplo) a través de uno de los puertos serie del Pixy.
- **Vídeo en bruto:** Al pulsar este botón de muestra de vídeo en bruto, sin procesar. Esto es útil para el enfoque de ajuste, brillo de la cámara, etc.
- **Cocinado de vídeo:** Al pulsar este botón se muestra procesado de vídeo "cocinado". Presionar este botón si desea obtener una buena idea de cómo Pixy está procesando las imágenes dentro del programa predeterminado. El modo de cocinado es vídeo en bruto con superposición de vídeo procesado.
- **Configurar:** Al pulsar este abre el diálogo Configurar, que contiene varios parámetros configurables para Pixy y PixyMon.

CAPÍTULO II

2. METODOLOGÍA

2.1 TIPO DE ESTUDIO

El presente estudio es una investigación grupal, aplicada, de campo, documental, experimental, deductiva y científica debido a la forma de obtención de los datos para realizar el desarrollo del dispositivo electrónico que transportara objetos de forma autónoma, mediante el uso de sensores que permita localizar a la persona a seguir.

2.1.1 Investigación Bibliográfica- Documental

La información adquirida será un proceso de investigación basada en electrónica y robótica de prototipos de robots móviles y autónomos, además de sensores que permitan la localización del individuo a la que se va a seguir.

2.1.2 Nivel o tipo de la exploración

El proceso investigativo será de Nivel Exploratorio debido a que permitirá conocer los factores, características y procesos actuales del diseño e implementación del robot.

2.2 POBLACIÓN Y MUESTRA

2.2.1 Población

Es la información de las diversas pruebas realizadas, en función del reconocimiento del objeto y su capacidad de reacción para seguir a la persona.

2.2.2 Muestra

La muestra está establecida por el cálculo de población infinita o desconocida.

$$n = \frac{Z_{\propto^2 pq}}{i^2}$$

$Z_{\propto}$ Valor correspondiente de la distribución de gauss, $Z_{\propto}$ = 0.05= 1,96

p prevalencia esperada de los datos de evaluación a evaluar

p=80%

q=1-p

q= 1-0.8

q=0.2

i el error que se pretende cometer, de un 20%

$$n = \frac{1.96^2(0.8)(0.2)}{(0.2)^2}$$

$$n = \frac{0.6146}{0.04}$$

$$n = 15.36$$

$$n \approx 15$$

2.2.3 Hipótesis

"Con el diseño e implementación del robot transportador permitirá seguir a una persona de forma autónoma en superficies planas y con una inclinación no mayor a 30°."

2.3 OPERACIONALIZACIÓN DE VARIABLES

VARIABLE	DIMENSIÓN	IDENTIFICADORES	ÍTEMS
INDEPENDIENTE Objeto	Precisión en la identificación de objetos	Objeto a seguir por medio de la cámara	Cámara Pixy
Trayectoria		Pulsos de medición de distancia	Sensores ultrasónicos

DEPENDIENTE Diseño e implementación del robot transportador	Módulo de recepción de imagen Movilidad	Captura de imagen Trayectoria a realizar	Localización del objeto Correcta movilidad y funcionamiento

Tabla 2. Variable Independiente y Dependiente

Fuente: Autores

2.4 PROCEDIMIENTOS

Para el diseño e implementación de un robot transportador de objetos se implementó la siguiente serie de procesos a seguir, como se observa a continuación.

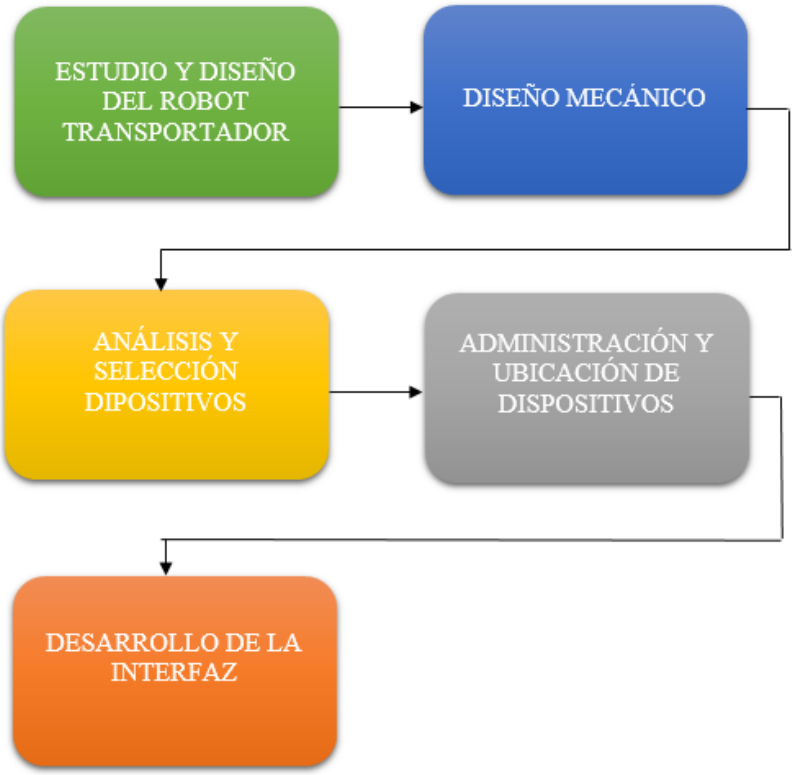


Figura 18. Diagrama de Procedimiento

Fuente: Autores

Diagrama de conexiones

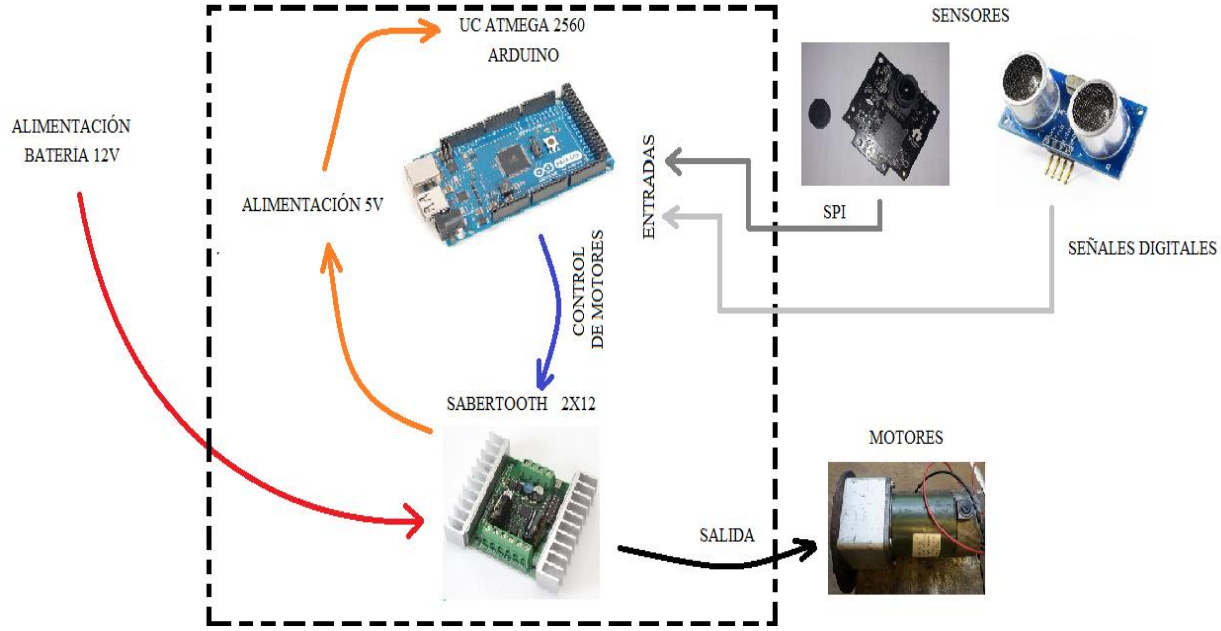


Figura 19. Diagrama de Conexiones

Fuente: Autores

2.5 PROCESAMIENTO Y ANÁLISIS

2.5.1. Estudio y Diseño del robot transportador

El estudio del diseño del robot transportador personal de objetos se sustenta en las medidas de gradas estándar 15x30cm (alto x largo), basado en prototipos y recolectando información sobre robótica y robots autónomos los autores decidieron la construcción del robot en tracción de pistas de deslizamiento (oruga), teniendo en cuenta esto su diseño mecánico es el siguiente.

2.5.2. Diseño Mecánico

El diseño mecánico fue implementado en AutoCAD 2016 para tener una mayor exactitud en el desarrollo de cada una de las piezas.

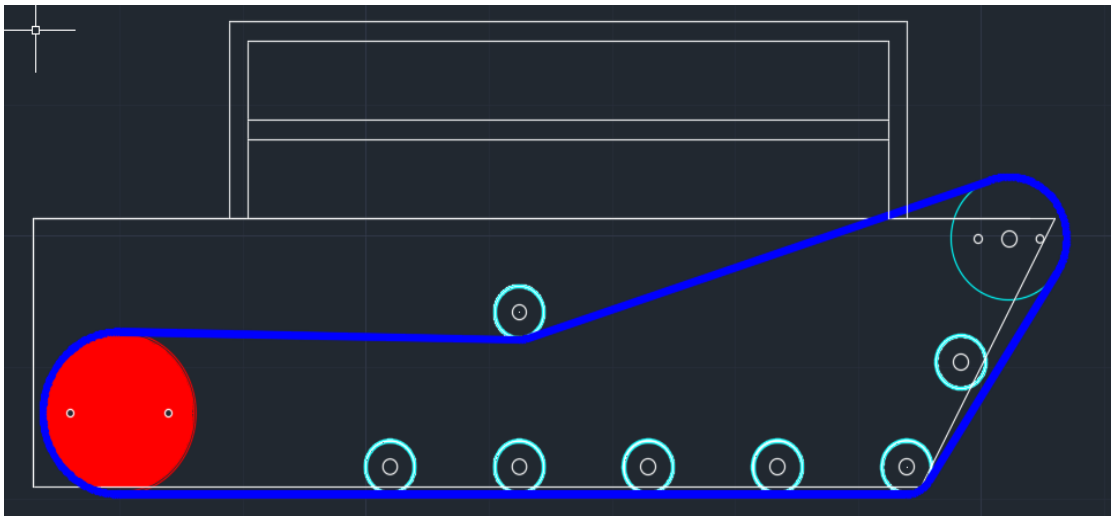


Figura 20. Diseño Mecánico en AutoCAD 2016

Fuente: Autores

El material seleccionado es el acero y nylon, por su alta resistencia, soldabilidad, ductilidad y trabajabilidad, las medidas de la plancha de acero son de 83cm de largo por 20.4cm de alto, se usó una plancha de 3mm de espesor. Esto se puede observar en la Figura 21.

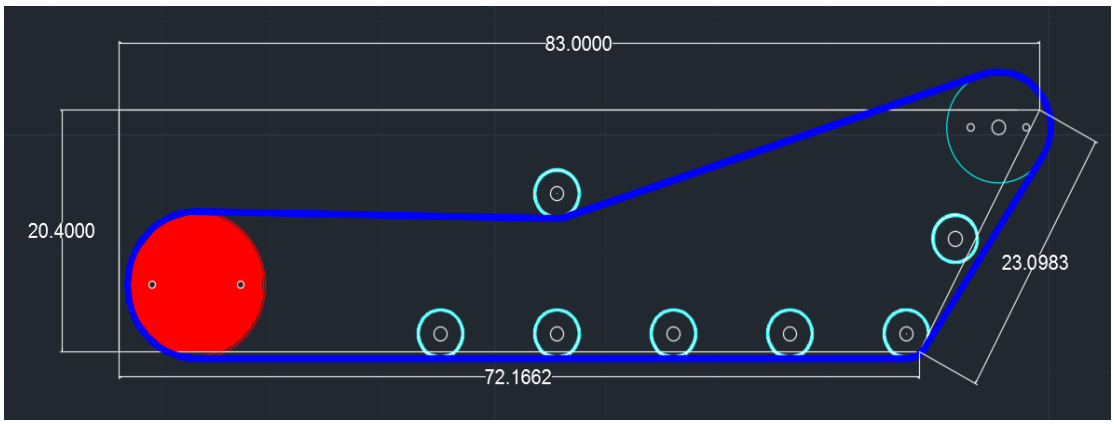


Figura 21. Medidas Robot transportador

Fuente: Autores

El motor está recubierto de un rodillo de 12.4cm de diámetro para obtener una mayor tracción y aumentar su desplazamiento con un menor número de revoluciones, posee

dos tornillos de 6mm de diámetro para poder realizar los ajustes del motor con el rodillo.

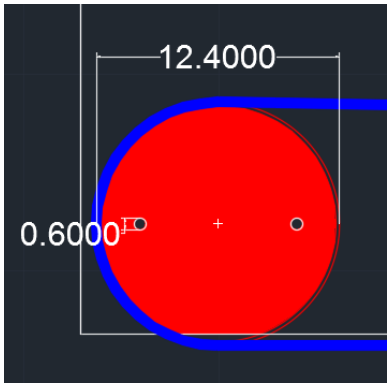


Figura 22. Medida de xxx

Fuente: Autores

Para un mejor funcionamiento está conformado por ocho rodillos, siete de 4.2cm de diámetro los cuales cinco están al piso, para poder mantener la oruga templada y con la menor cantidad de vibración en la superficie donde se desliza, junto con otro rodillo ubicado en la parte frontal de la estructura, el cual permite reducir la cantidad de impacto cuando la oruga se encuentra con superficies irregulares, en la parte superior posee un rodillo de 9.4cm de diámetro usado como guía para que la oruga no se desplace hacia los costados, por ultimo posee un rodillo que cumple la función de templador, el cual permite regular la tensión de la oruga.

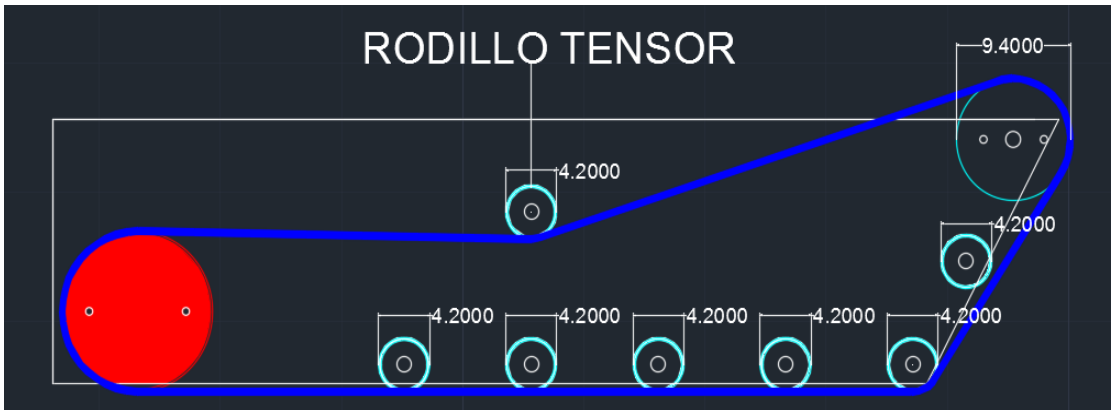


Figura 23. Medida de los rodillos

Fuente: Autores

El resultado obtenido es:

Figura 24. Forma Final del Diseño Mecánico

Fuente: Autores

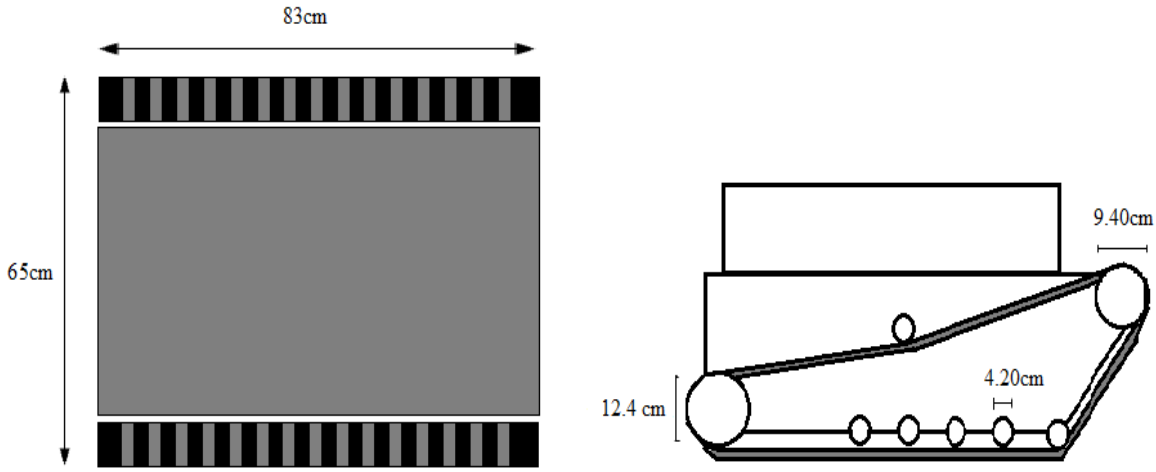


Figura 25. Diseño del robot Transportador

Fuente: Autores

2.5.3. Análisis y selección de dispositivos

2.4.3.1 Cámara Pixy (CMUcam5)

La Pixy CMUCam5 es el sensor principal del robot transportador porque cumple con la función de seguimiento del objeto que la persona portara. Se encuentra ubicada en la parte frontal y mitad del robot para mayor facilidad de detección por parte del sistema, está se comunica con el arduino a través de protocolo SPI enviando los datos correspondientes que se necesita para cumplir con la programación (Anexo 1).

2.4.3.2 Ultrasónicos

El sensor ultrasónico HC-SR04 fue seleccionado para el presente proyecto, por su precisión y su capacidad de detectar objetos que se encuentra en un rango máximo de 4m y rango mínimo de 2cm. Teniendo en cuenta que un sensor ultrasónico puede trabajar en áreas oscuras sin que las luces detengan o afecten su funcionamiento. La adquisición de datos de este dispositivo se encuentra en un rango de 30° siendo para el proyecto el sensor indicado para detectar, evadir obstáculos y cualquier impedimento. Se colocó los sensores como se muestra en la Figura 26**.**

2.4.3.3 Sabertooth 2x12

La selección del Sabertooth 2x12 (ver Anexo 4) es para el cambio de giro que se le debe ofrecer a los motores para realizar movimientos hacia delante, atrás, derecha e izquierda respectivamente. El sabertooth da la posibilidad de trabajo con un voltaje de 6v a 24v nominal. También posee un regulador de 5v sirviendo como alimentación para el resto de dispositivos la cámara Pixy y el arduino que es la alimentación necesaria para su funcionamiento, recibe 2 señales para el movimiento de los motores, una de las ventajas de dichas señales es que permite trabajar con pwm para el control de velocidad de los motores.

2.4.3.4 Microcontrolador ATMEGA 2560

El microcontrolador posee una gran capacidad de procesamiento y memoria, con este dispositivo podemos obtener datos, procesarlos y ejecutar las funciones

necesarias que requiera el robot transportador tal siendo el indicado por el manejo de varios dispositivos y organización de datos para el cumplimiento de sus funciones.

2.4.3.5 Motores DC con Reductor

El motor SF7152 (ver Anexo 8) genera una potencia de 150W, permitiendo un desplazamiento de un gran peso, debido a su reductora 1/180 la cual ayuda a que el motor no sufra ningún desgaste y genere la fuerza necesaria para la movilidad del robot transportador.

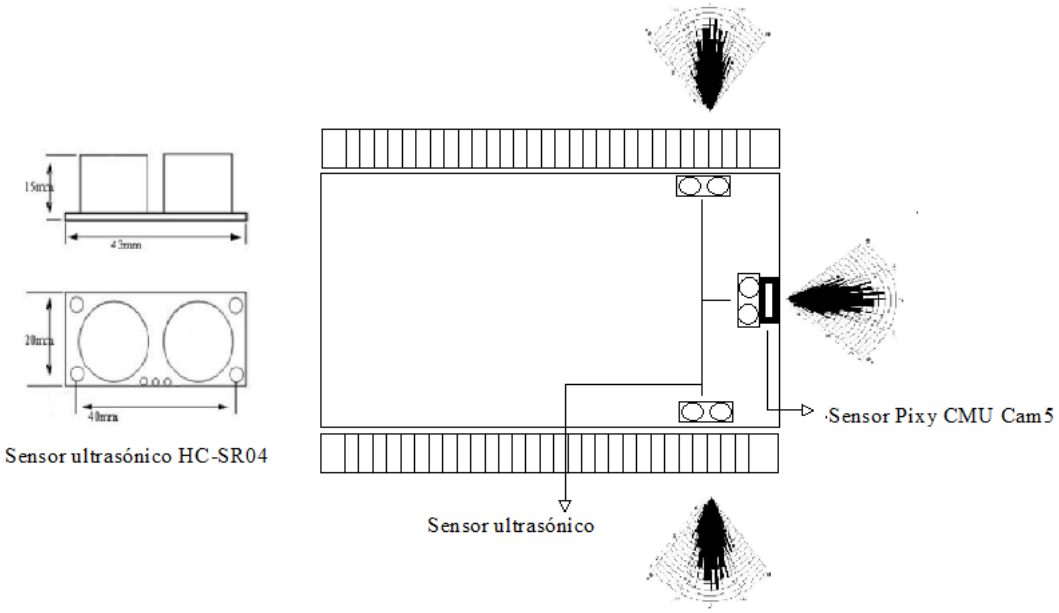


Figura 26. Ubicación sensores ultrasónicos y Cámara Pixy

Fuente: Autores

2.5.4. Administración y Ubicación de dispositivos

El robot transportador después de ser realizado su construcción y estructura, luego de cierto análisis de los proyectistas se estableció la ubicación óptima de los elementos para un mejor funcionamiento del robot transportador (ver Anexo 7).

- **Motores o actuadores:** Los motores fueron ubicados en la parte de atrás del robot para poder tener más tracción al momento del desplazamiento.

- **Sensores Ultrasónicos:** Son tres sensores ultrasónicos ubicados uno al frente, para poder detener los movimientos del motor y uno en cada lado en la parte delantera del robot transportador para evitar los obstáculos.
- **Sabertooth:** Esta ubicado en la parte central del robot cerca de los motores y cerca del Arduino para la administración segura y fácil de los dispositivos.
- **Cámara Pixy:** Considerada el sensor principal por la detección del objeto a seguir está ubicada al frente en la parte central del robot.
- **Alimentación:** Se encuentra ubicado en la parte de adelante para poder compensar el peso de los motores y tratar de encontrar equilibrio en el robot transportador; se trata de dos baterías de 12v a 6.5 A para alimentar a los motores, sabertooth, Arduino y Cámara Pixy.

2.5.5. Diseño del software

El diseño y programación fueron realizados en Arduino para proporcionar las acciones del robot transportador como lo muestra en el Anexo 1.

2.5.6. Circuito control general

El circuito de control general (ver Anexo 2) es diseñado para administrar todo los dispositivos electrónicos distribuidos en el robot transportador para poder controlarlo con el microcontrolador ATmega 2560, según los requerimientos que el robot necesite. El Arduino Mega genera pulsos PWM por medio de los pines 9 y 10 hacia el sabertooth 2x12 que recibe las señales S1 y S2 para el control de giros de los motores. Las entradas que recibe el Arduino Mega por los sensores HC-SR04 ingresan a los pines 22, 24,26, como lecturas de sensor (echo) y los pines 23, 25,27 son los pines que generan el pulso (trig), por medio del protocolo SPI la cámara se comunica con el arduino para la recepción de información y posterior generación de movimiento para los motores.

2.5.7. Desarrollo de la interfaz en el Microcontrolador

Para el desarrollo el medio de programación a utilizar ha sido la plataforma Arduino, en la cual se ha utilizado librerías, variables, funciones, condiciones, para el desarrollo óptimo del programa.

A continuación explicaremos el funcionamiento de la programación para las etapas y funciones óptimas del robot transportador.

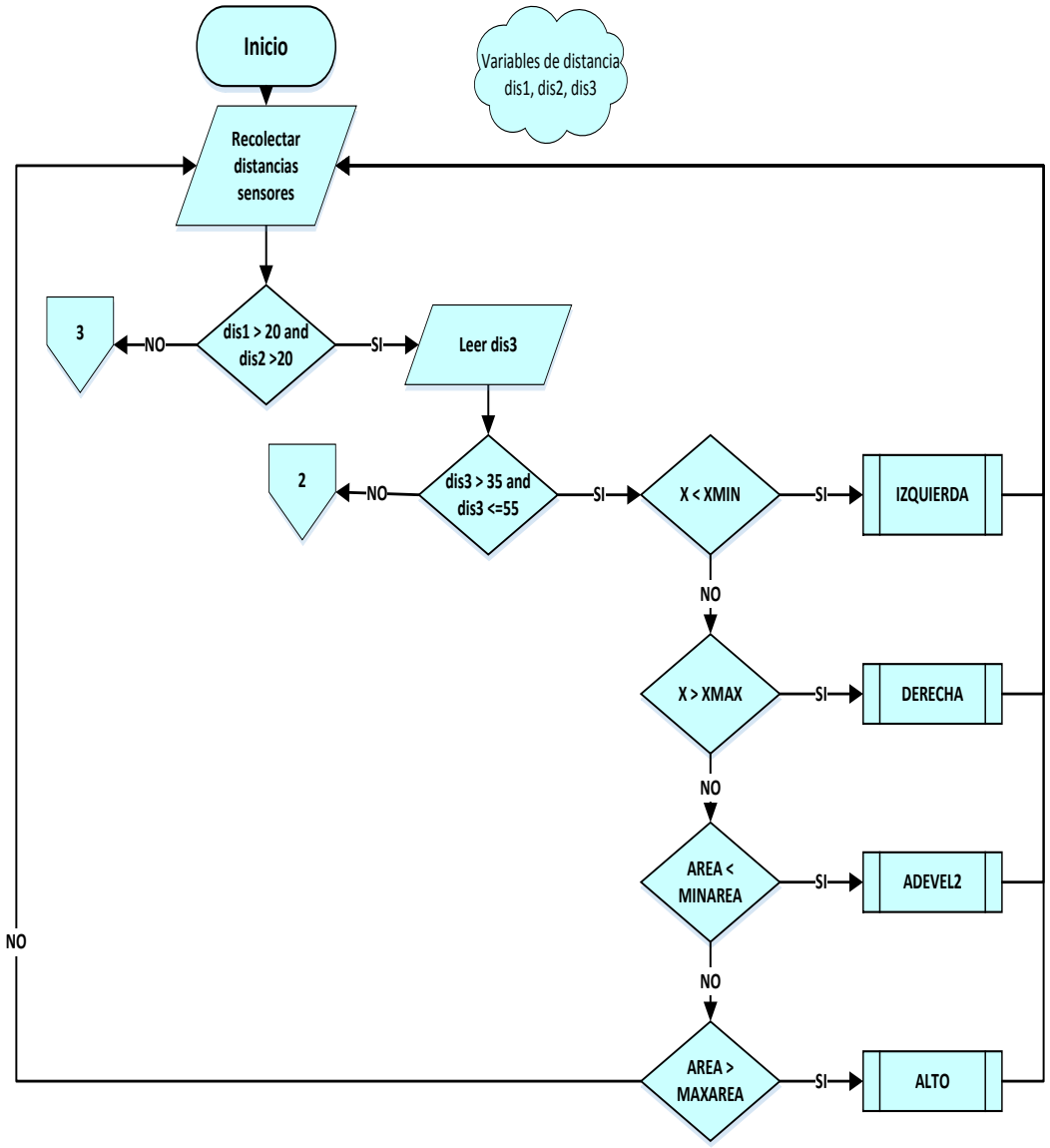


Figura 27. Programación de los movimientos para seguir objeto cuando las condiciones de distancia se cumplen.

Fuente: Autores

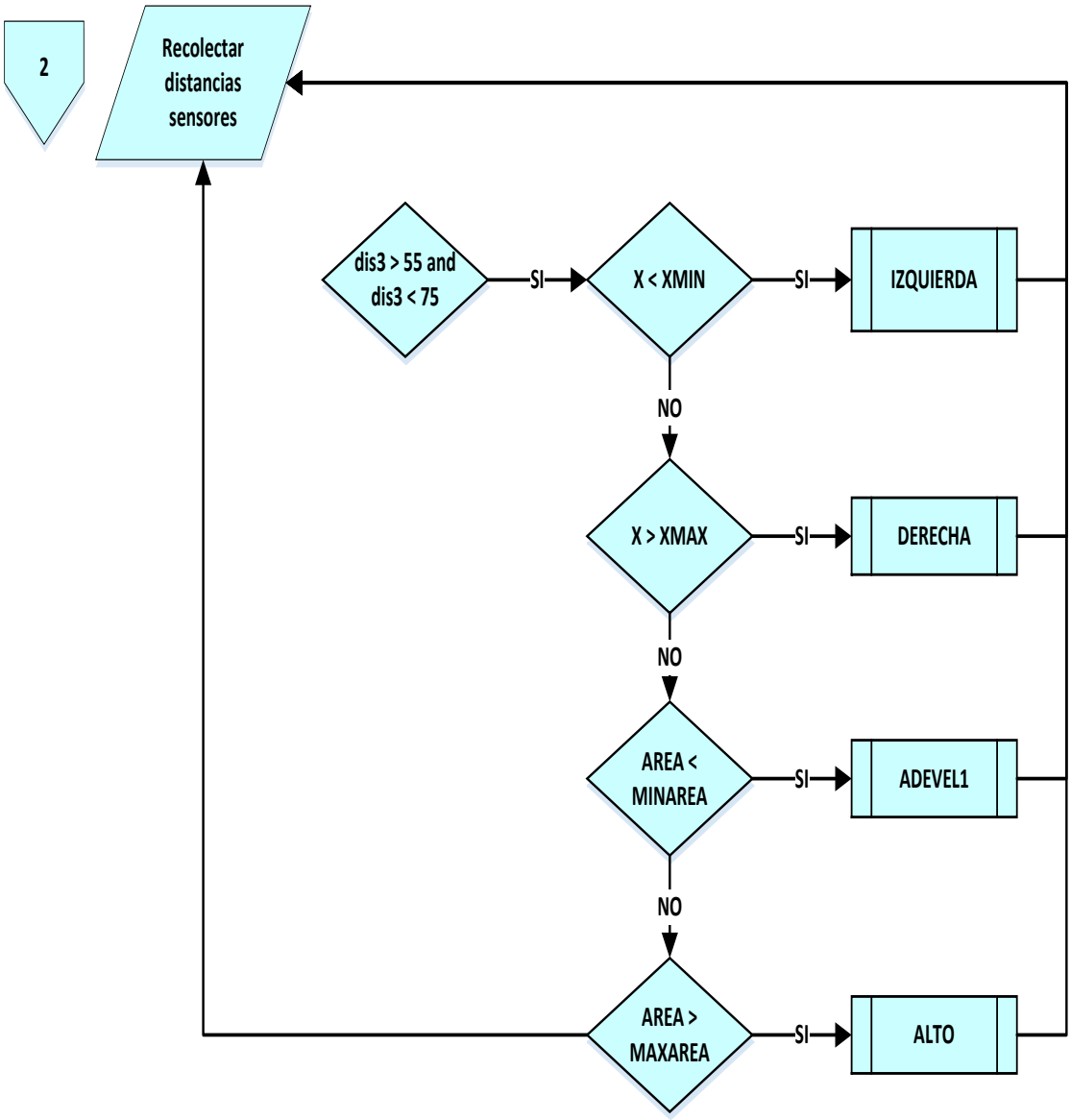


Figura 28. Programación de movimiento para distancia mayor a 55cm y menor a 75cm.

Fuente: Autores

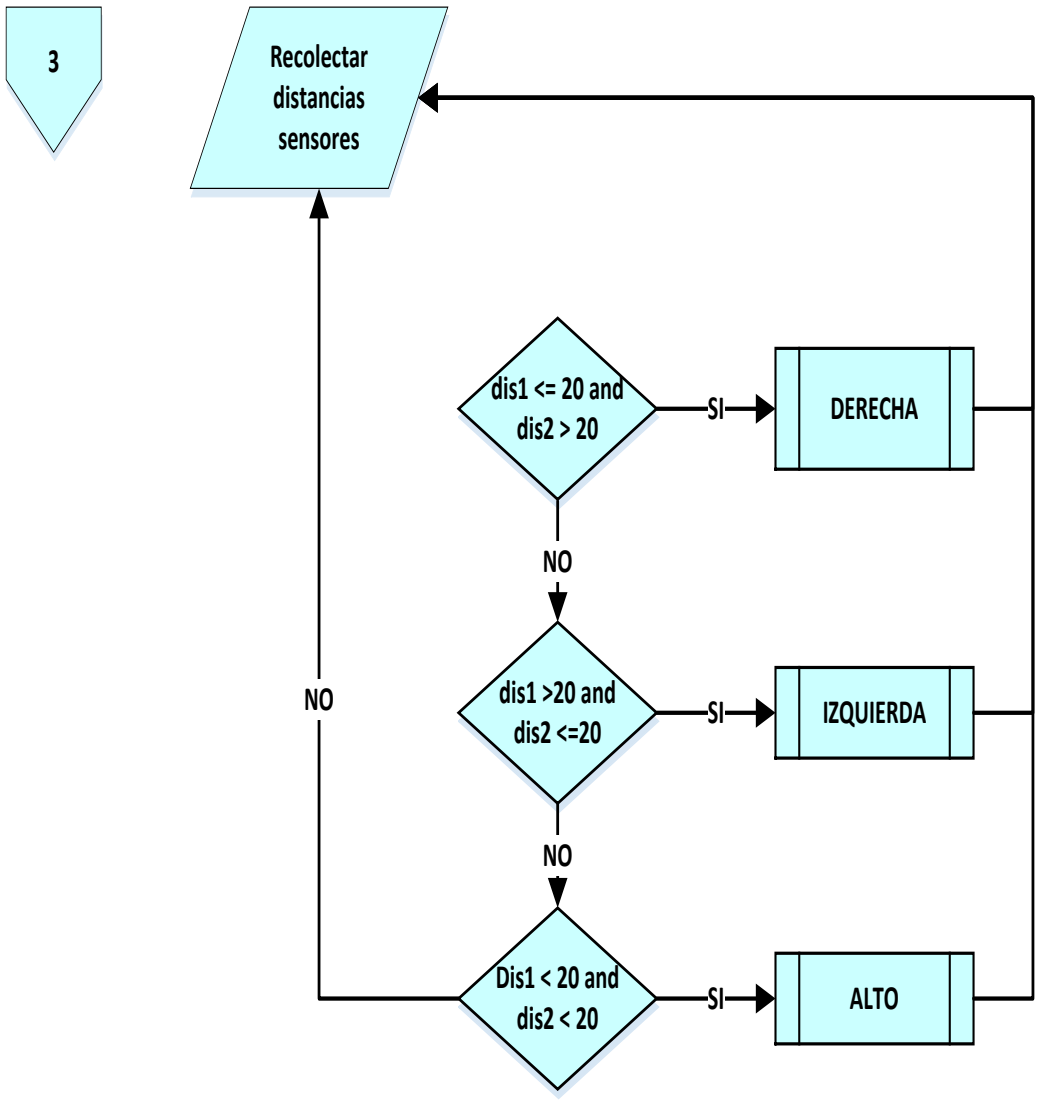


Figura 29. Programación de movimiento cuando los sensores laterales detectan obstáculo.

Fuente: Autores

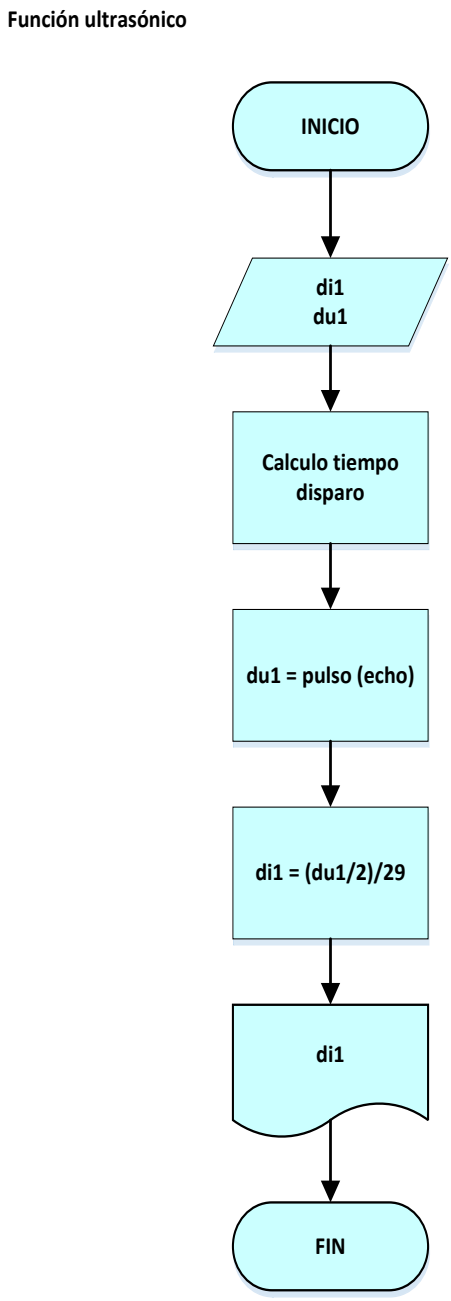


Figura 30. Función para el cálculo de las distancias de los sensores ultrasónicos.

Fuente: Autores

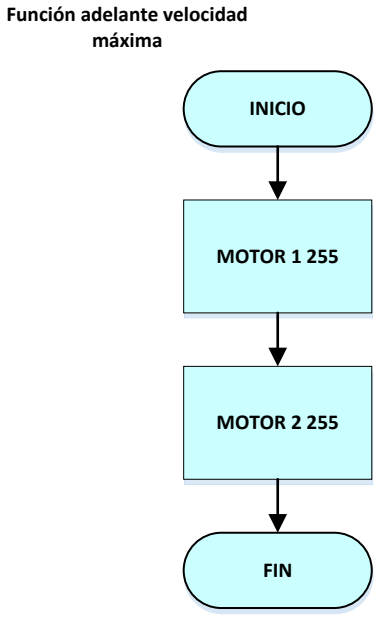
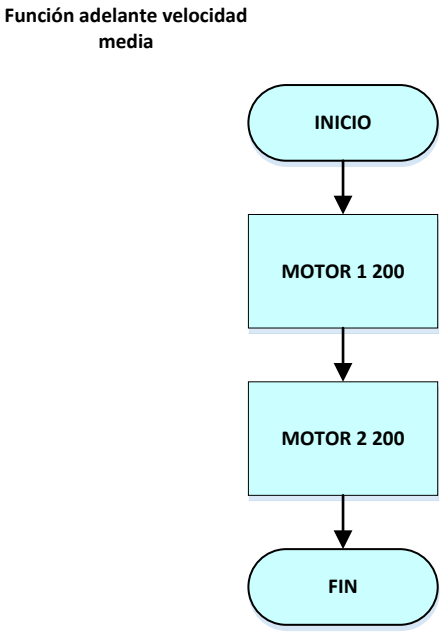


Figura 31. Función para movimiento de los motores a su máxima velocidad, (255 con pwm).

Fuente: Autores



*F*igura 32. Función para movimiento a velocidad media (200 con pwm).

Fuente: Autores

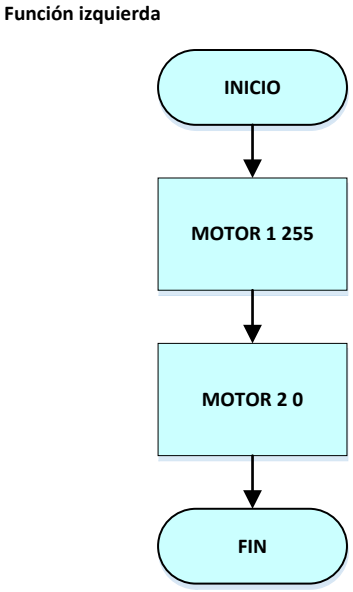


Figura 33. Función para movimiento hacia la izquierda.

Fuente: Autores

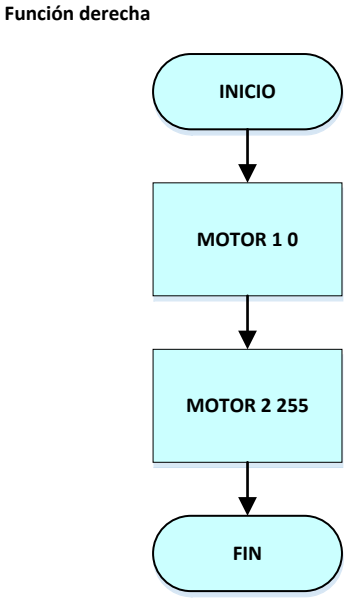


Figura 34. Función para movimiento derecha.

Fuente: Autores

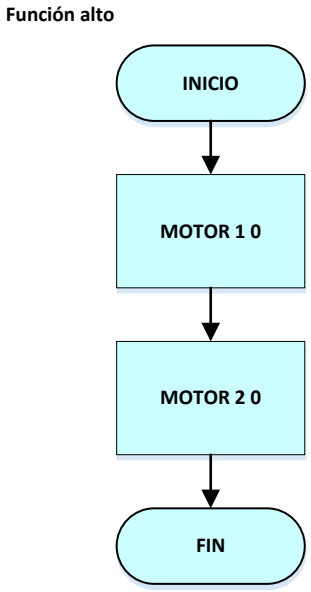


Figura 35. Función para detección del robot

Fuente: Autores

2.5.8. Guía de aplicación

Este robot tiene la característica de poderse mover libremente siguiendo a un usuario a través de reconocimiento de objetos.

El usuario procede a realizar la captura del objeto de forma manual para que este se grabe en el procesador de la Cámara Pixy, se realiza el reconocimiento y posicionamiento del objeto para ser reconocido, usando como medio de programación a Arduino, el cual ayudara a realizar el reconocimiento de cada uno de los movimientos del objeto y procede a dar movimiento a los motores para respectivamente el movimiento del robot transportador.

Para una captura óptima de las imágenes con la Cámara Pixy es necesario el uso del software propio de la cámara llamado PixyMon (Figura 36), mediante esta aplicación podemos observar lo que la cámara está viendo, y el objetivo que queremos seguir.

Para la captura y funcionamiento de objetos con la cámara Pixy ver (Anexo 9).

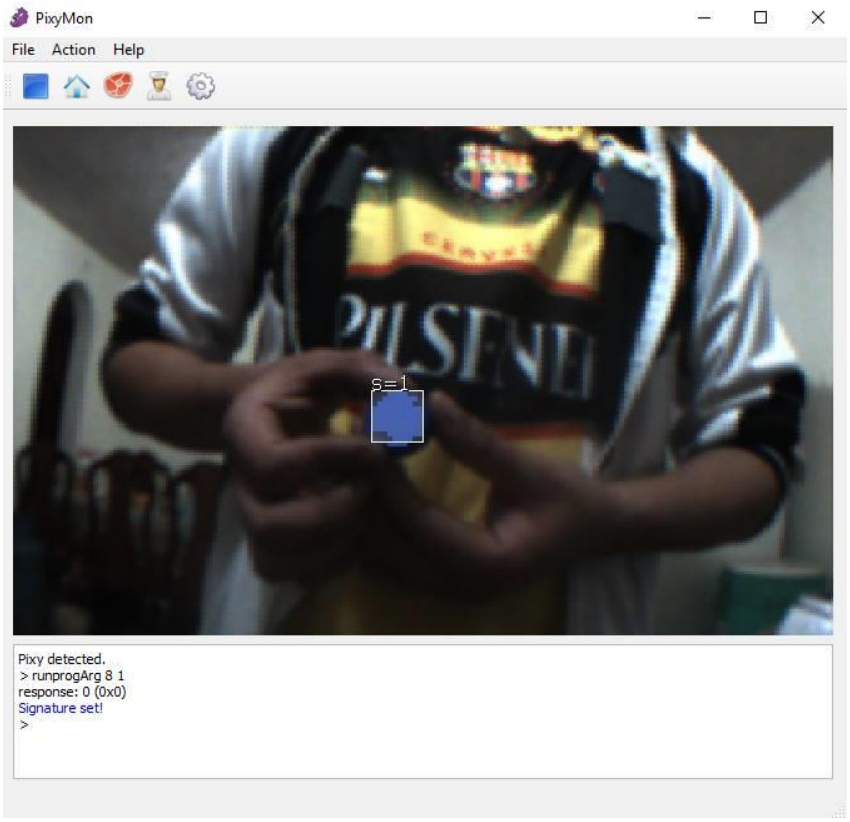


Figura 36. Software PixyMon

Fuente: Autores

2.6 COMPROBACIÓN DE LA HIPÓTESIS

Para la comprobación de la hipótesis se utiliza el método estadístico chi-cuadrado, puesto que los valores obtenidos no son números si no categorías y accede a dos grados de posibilidad, alternativa la que se quiere comprobar y nula la que se rechaza la hipótesis alternativa.

2.6.1. Planteamiento de la hipótesis estadística

Hipótesis nula (Ho): Con el diseño e implementación del robot transportador no permitirá seguir a una persona de forma autónoma en superficies planas y con una inclinación no mayor a 30°.

Hipótesis alternativa (Hi): Con el diseño e implementación del robot transportador permitirá seguir a una persona de forma autónoma en superficies planas y con una inclinación no mayor a 30°.

2.6.2. Establecimiento de nivel de significancia.

Las pruebas se realizaron con un 80% de confiabilidad, es decir, se trabajó con un nivel de significancia de $\alpha = 0.05$.

2.6.3. Determinación del valor estadístico de prueba.

Si el valor de CHI-CUADRADO es menor o igual al Chi-Cuadrado crítico entonces se acepta la hipótesis nula, caso contrario se la rechaza.

$$X^2 \leq Valor\ crítico$$

Para aceptar o rechazar esta hipótesis se tomó en cuenta el sistema de censado, que determina si utilizando la cámara Pixy podemos seguir o no al objeto en un área plana o con inclinación. Por lo que se requirió tomar un número de muestras realizada con el sistema.

	RECONOCIMIENTO DEL OBJETO	
Numero de muestras	SUPERFICIE PLANA	SUPERFICIE CON INCLINACION
1	SI	SI
2	SI	NO
3	SI	NO
4	SI	NO
5	SI	SI
6	SI	SI
7	SI	SI
8	SI	NO
9	SI	SI
10	NO	SI

11	SI	SI
12	SI	SI
13	SI	NO
14	SI	SI
15	SI	NO
16	SI	SI
17	NO	SI
18	SI	SI
19	SI	NO
20	SI	NO

Tabla 3. Número de muestras

Fuente: Autores

Para realizar el cálculo de X^2 es necesario trabajar con la frecuencia obtenida, y frecuencia esperada los valores se han obtenido en la Tabla 5 y 6.

FRECUENCIA OBTENIDA

FUNCIONAMIENTO	RECONOCIMIENTO DEL OBJETO		
	1	2	TOTAL
SI	18	12	30
NO	2	8	10
TOTAL	20	20	40

Tabla 4. Frecuencia Obtenida

Fuente: Autores

FRECUENCIA ESPERADA

FUNCIONAMIENTO	RECONOCIMIENTO DEL OBJETO		
	1	2	TOTAL
SI	15	15	30
NO	5	5	10
TOTAL	20	20	40

Tabla 5. Frecuencia esperada

Fuente: Autores

NUMERO DE FILAS	2
NUMERO DE COLUMNAS	2
VALOR CALCULADO X	4,80
GRADOS DE LIBERTAS	1
NIVEL DE SIGNIFICANCIA	0,05
PROBABILIDAD	0,8
VALOR CRITICO	3,8415

Tabla 6. Resultados

Fuente: Autores

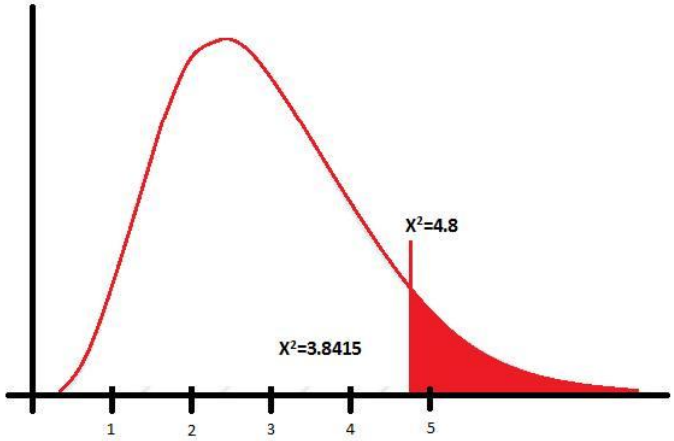


Figura 37. Resultado de comparación X^2tabla con el X^2calculado

Fuente: Autor

Como $X^2\ calculado = 4{,}80 > x^2\ tabla = 3{,}8415$ (Anexo 6) se rechaza Ho y se concluye, al tener un nivel de significancia del 0.05 y aceptar la hipótesis, es decir:

El diseño e implementación del robot transportador permitirá seguir a una persona de forma autónoma en superficies planas y con una inclinación no mayor a 30°.

CAPÍTULO III

3. RESULTADOS

Con el desarrollo del robot transportador se pudo obtener una relación de consumo de voltaje respecto al tiempo de uso en superficie plana y superficie con inclinación.

PRUEBA	VOLTAJE	SUPERFICIE
1	12.53	Plana
2	12.51	Plana y peso
3	12.48	Con inclinación
4	12.43	Con inclinación y peso

Tabla 7: Pruebas en superficies

Fuente: Autores

Figura 38. Prueba en superficie plana

Fuente: Autores

Figura 39. Prueba en superficie inclinada.

Fuente: Autores

Figura 40. Prueba superficie plana con peso

Fuente: Autores

CAPÍTULO IV

4. DISCUSIÓN

El desarrollo del Robot transportador, será capaz de ayudar a movilizar objetos de pesos no mayores a 50 libras en superficies planas y con inclinaciones no mayores a los 30°, siendo está de manera autónoma.

La realización de la adquisición de datos es mediante 3 sensores los cuales se encuentran distribuidos en el robot transportador ya examinados anteriormente, con el fin de una recopilación adecuada de datos; con el propósito de evaluar su entorno de objetos uniformes, fijos o móviles, mediante emisión y recepción de sonido desde el sensor hacia dichos obstáculos permitiendo la facilidad de evitar los mismos, para un posicionamiento óptimo del robot transportador al seguir su trayectoria.

Además de poseer un sensor de imagen con la capacidad de reconocimiento de figuras o formas, con la finalidad de seguir a su propietario a donde este se dirija, mediante el uso de esta cámara se llegó a realizar diversas pruebas siendo estas en su mayoría positivas.

CAPÍTULO V

5. CONCLUSIONES Y RECOMENDACIONES.

5.1 CONCLUSIONES

- El consumo de corriente pico al producirse el arranque en el sistema es de 9 Amperios y su corriente nominal es de 6 Amperios, variando según las condiciones de superficie para su desplazamiento.

- El error evaluado al momento de realizar las pruebas es que existe un retraso producido por la mecánica del robot debido a que por el uso de la oruga su desplazamiento en los giros es más lento, haciendo que la cámara tienda a perder su objetivo.

- Al momento de la evaluación del robot en las diferentes superficies en los que se desplaza sería un factor a tener muy encuenta debido a que el coeficiente de rozamiento de la oruga de caucho puede tender a patinar en superficies demasiado lisas, provocando que el voltaje consumido sea mayor que en una superficie normal.

- El dispositivo de seguimiento tiende a fallar en lugares con demasiada exposición luminosa y con poca luminosidad interfiriendo en su reconocimiento del objeto.

- Al momento de capturar al objeto tener muy en cuenta el tipo de color y tamaño del objeto a seguir debido a que si es muy pequeño este no será reconocido por la cámara Pixy y si el color es común tiende a producir errores de reconocimiento en el dispositivo.

5.2 RECOMENDACIONES

- La funcionalidad del 100% de la capacidad de los motores es a 24v, por motivos de seguridad es recomendable trabajar a la mitad de su capacidad para reducir la velocidad de los motores.

- Para el reconocimiento de la cámara Pixy se recomienda el uso de los siguientes colores: Azul, Rojo, Amarillo siendo los colores más óptimos que identifica la cámara también utilizar objetos de tamaño considerable.

- Para mejor seguimiento del robot es recomendable usar la captura de solo un objeto debido a que la captura de varios indicadores pueden ocasionar conflictos en el seguimiento hacia la persona creando errores en su funcionamiento.

- Realizar una relación del peso a transportar debido a que exige un mayor consumo de voltaje en las baterías no excediéndose del peso establecido.

- El recomendable utilizar baterías según el uso que se le va a dar al robot debido que por la corriente que exige los motores las baterías tienden a inflarse, explotar. Para evitar esto se recomienda el uso de baterías de Hidrógeno.

CAPÍTULO VI

6. PROPUESTA

6.1. Título de la propuesta

Implementación de un sistema de seguimiento mediante el uso de frecuencia y giroscopio.

6.2. Introducción

Para el óptimo funcionamiento del robot transportador se requiere un sistema del uso de frecuencia y giroscopio que facilitará la ubicación correcta entre el usuario y el robot permitiendo que su seguimiento sea funcional siguiendo la misma trayectoria de la persona.

6.3. Objetivos

6.3.1. Objetivo General

- Diseño e implementación de un robot transportador de objetos mediante el uso de frecuencia y giroscopio.

6.3.2. Objetivo Específico

- Desarrollar el sistema de localización mediante giroscopio
- Desarrollar el sistema de seguimiento mediante uso de frecuencia.

6.4. Fundamentación científica-técnica

Para el desarrollo e implementación del sistema de seguimiento con uso de frecuencia y giroscopio se debe tener encuenta que en la actualidad contamos con tecnología que día a día ha evolucionado permitiendo mayor comodidad y facilidad en el desarrollo de nuestras vidas.

El sistema de localización consta de un giroscopio, los giroscopios, son dispositivos que miden o mantienen el movimiento de rotación. MEMS (sistemas

microelectromecánicos) giroscopios son pequeños sensores para medir la velocidad angular.

Los Giroscopios se pueden utilizar para determinar la orientación y normalmente podemos encontrar en la mayoría de los sistemas de navegación autónomos. Obteniendo la facilidad para equilibrar un robot, un giroscopio también es usado para medir la rotación de la posición de equilibrio y enviar correcciones al motor con el fin de cumplir interacción con el usuario o con un mejor seguimiento.

6.5. Descripción de la propuesta

La implementación del sistema de usos de frecuencia en el robot, permitirá un seguimiento del robot transportador personal con mayor exactitud debido a que su funcionalidad será específico solo para el operador o usuario tendiendo siempre a mantener cerca al usuario haciendo un mapeo de la frecuencia en la que se encuentren ajustadas dependiendo el uso del tipo de dispositivo tendremos mayor o menor distancia de reconocimiento, además de desarrollar el sistema de seguimiento con giroscopio para determinar la orientación para poder equilibrar los motores y enviar una corrección de los mismos para su trayectoria.

6.6. Diseño organizacional

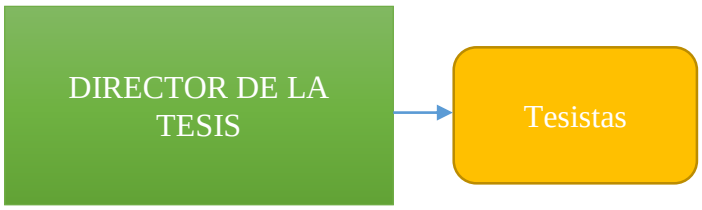


Figura 41. Esquema organizacional

Fuente: Autores

En la Figura 41, se observa el esquema organizacional de las personas que participan en el desarrollo e implementación para la propuesta.

6.7. Monitoreo y evaluación de la propuesta.

Para el monitoreo y evaluación de la propuesta del sistema de seguimiento por uso de frecuencia y giroscopio se realizará las pruebas necesarias en su trayectoria evaluando la posición del robot transportador verificando si cumple o no con el posicionamiento correcto en los motores y confiabilidad de seguimiento hacia la persona, obteniendo resultados para poder establecer las limitaciones que presenta el sistema al cumplir con su funcionamiento.

CAPÍTULO VII

7. BIBLIOGRAFÍA

Accudiy. (2011). Obtenido de http://www.accudiy.com/download/HC-SR04_Manual.pdf

Adaufrit. (s.f.). Obtenido de https://www.adafruit.com/product/1906

Baratune Ollero, A. (2001). Robótica, manipuladores y robots móviles. España: Alfaomega Marcocombo.

Bravo, A. (03 de agosto de 2011). *Robótica al descubierto*. Obtenido de http://solorobotica.blogspot.com/2011/08/motores-de-corriente-continua.html

Carletti, E. J. (2007). *Robots*. Obtenido de http://robots-argentina.com.ar/Sensores_rangers.htm

Carpio Moreta, M. A. (s.f.). *Inteligencia Artificial Aplicada a la Robótica*. Obtenido de Monografias.com: http://www.monografias.com/trabajos93/inteligencia-artificial-aplicada-robotica/inteligencia-artificial-aplicada-robotica.shtml

cmucam.org. (2006-2011). Obtenido de http://www.cmucam.org/projects/cmucam5/wiki/Installing_PixyMon_on_Windows_Vista_7_or_8

cmucam.org. (2006-2011). Obtenido de http://www.cmucam.org/projects/cmucam5/wiki/PixyMon_Overview

CMUcam5 Pixy. (s.f.). Obtenido de http://cmucam.org/projects/cmucam5/wiki

Constain Arroyave, J. A. (12 de Abril de 2012). *Diseño e implementación de la teleoperación de un robot móvil*. Obtenido de http://docplayer.es/7542027-Diseno-e-implementacion-de-la-teleoperacion-de-un-robot-movil-javier-antonio-constain-arroyave.html

Espericueta Gamez, S. (13 de diciembre de 2013). *SlideShare*. Obtenido de http://es.slideshare.net/SarahiEspericuetaGamez/motorreductores-y-motores-de-dc

Gonzales de Rivera, G. (2016). *Robótica introducción*. Obtenido de http://arantxa.ii.uam.es/~gdrivera/robotica/introd.htm

Jesús. (06 de Marzo de 2009). *Historia del Arte de la Robótica*. Obtenido de http://robotik-jjlg.blogspot.com/2009/03/tipos-de-robots-2.html

Microcontroladores Pic - Programación en Basic. (s.f.). Obtenido de MikroElektronika: http://learn.mikroe.com/ebooks/microcontroladorespicc/chapter/introduccion-al-mundo-de-los-microcontroladores/

Navarro, K. (05 de junio de 2014). *Panama Hitek*. Obtenido de http://panamahitek.com/sensor-ultrasonico-hc-sr04-arduino/

Robotica. (s.f.). Obtenido de http://roble.pntic.mec.es/jlop0164/archivos/ROBOTICA.pdf

Sánchez, S. (s.f.). *Microcontroladores*. Obtenido de https://microcontroladoressesv.wordpress.com/arquitectura-de-los-microcontroladores/

Solano Huarocc, E. W. (17 de octubre de 2013). *Scribd*. Obtenido de https://es.scribd.com/doc/176792041/El-Puente-h

TESLABEM. (s.f.). Obtenido de http://teslabem.com/pixy-cmucam5-camara-de-robotica.html

Veloso. (09 de marzo de 2016). *ETOOLS*. Obtenido de http://www.electrontools.com/Home/WP/2016/03/09/regulador-de-voltaje-7805/

CAPÍTULO VIII

8. ANEXOS

8.1. Anexo 1. Programación en Arduino

```
#include <SPI.h>
#include <Pixy.h>

#define echo1 22
#define trig1 23
#define echo2 24
#define trig2 25
#define echo3 26
#define trig3 27

Pixy pixy;

int signature = 1;
int x = 0;
int y = 0;
int width = 0;
int height = 0;
int area = 0;
int maxArea = 1000;
int minArea = 800;
int Xmin = 140;
int Xmax = 180;

long duracion, distancia;

int motor1=8;
int motor2=9;

long dis1;
long dis2;
long dis3;

void adevel1(){
  analogWrite(motor1,255);
  analogWrite(motor2,255);
}

void adevel2(){
  analogWrite(motor1,128);
  analogWrite(motor2,128);
}

void izquierda(){
```

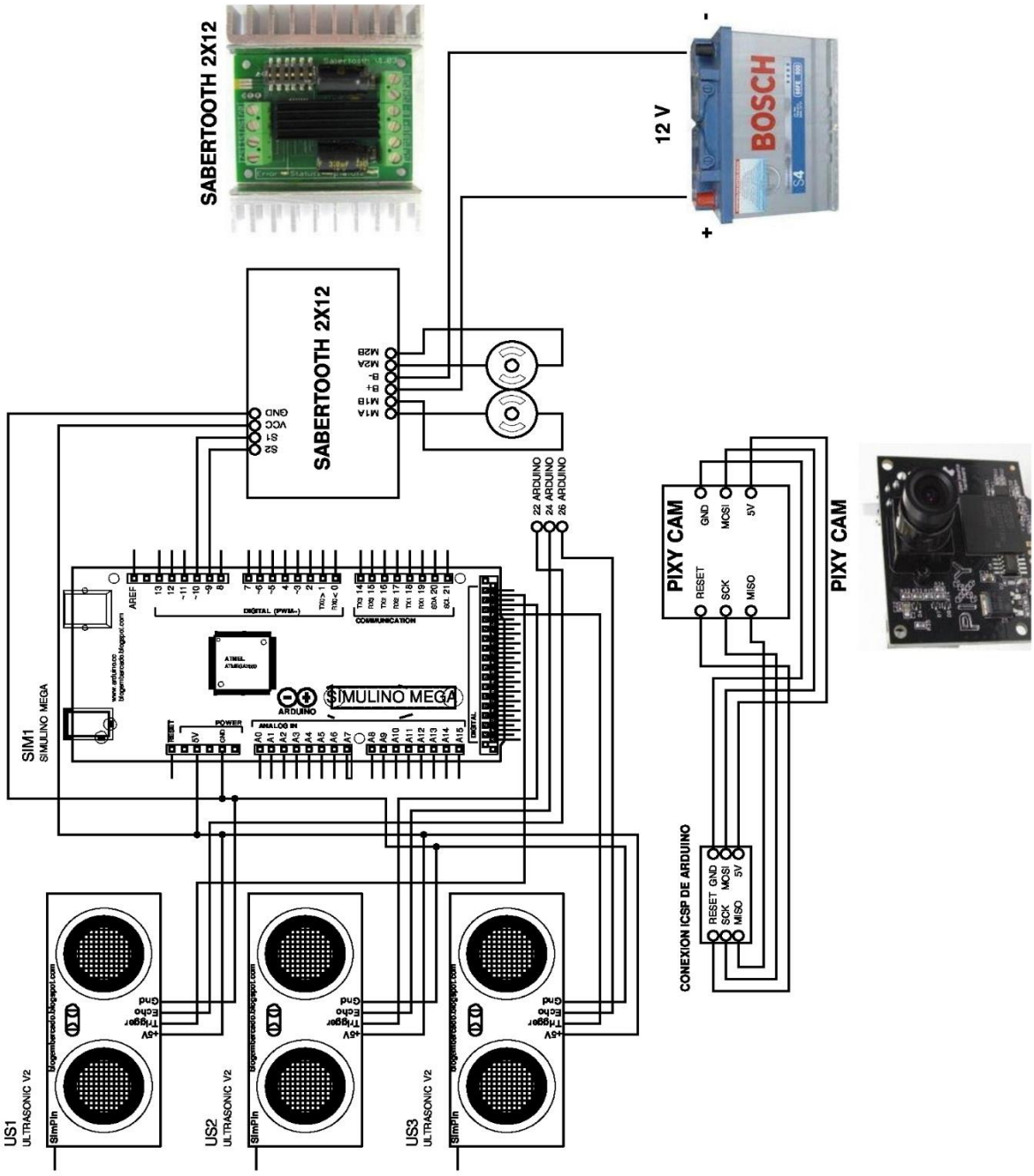
```
  analogWrite(motor1,255);
  analogWrite(motor2,5);
}

void derecha(){
  analogWrite(motor1,5);
  analogWrite(motor2,255);
}

void alto(){
  analogWrite(motor1,0);
  analogWrite(motor2,0);
}

long sensorfrente(){
  long di1,du1;
  digitalWrite(trig1,LOW);
  delayMicroseconds(2);
  digitalWrite(trig1,HIGH);
  delayMicroseconds(10);
  digitalWrite(trig1,LOW);
  du1=pulseIn(echo1,HIGH);
  di1=(du1/2)/29;
  return(di1);
}

void setup() {
  Serial.begin (9600);
  pinMode(echo1, INPUT);
  pinMode(trig1, OUTPUT);
  pinMode(echo2, INPUT);
  pinMode(trig2, OUTPUT);
  pinMode(echo3, INPUT);
  pinMode(trig3, OUTPUT);

  pinMode(10, 1);
  pinMode(11, 1);
  pinMode(12, 1);
  pinMode(13, 1);
}

void loop() {
  dis1=sensorizquierda();
  dis2=sensorderecha();
  if((dis1>20)&&(dis2>20)){
    dis3=sensorfrente();
    //distancia media
    if((dis3>35)&&(dis3<=55)){
      digitalWrite(10, 1);
      digitalWrite(11, 0);
      digitalWrite(12, 0);
```

```
    digitalWrite(13, 0);
    if (x < Xmin){
      izquierda();
      delay(200);
    }
    else if (x > Xmax){
      derecha();
      delay(200);
    }
    else if(area < minArea){
      adevel2();
      delay(200);
    }
    else if(area > maxArea){
      alto();
      delay(200);
    }
  }
  if((dis3>55)&&(dis3<=75)){
    digitalWrite(10, 1);
    digitalWrite(11, 0);
    digitalWrite(12, 0);
    digitalWrite(13, 0);
    if (x < Xmin){
      izquierda();
      delay(200);
    }
    else if (x > Xmax){
      derecha();
      delay(200);
    }
    else if(area < minArea){
      adevel1();
      delay(200);
    }
    else if(area > maxArea){
      alto();
      delay(200);
    }
  }
  if(dis3>75){
    digitalWrite(10, 0);
    digitalWrite(11, 1);
    digitalWrite(12, 0);
    digitalWrite(13, 0);
    alto();
    delay(200);
  }
}
if((dis1<=20)&&(dis2>20)){
  digitalWrite(10, 1);
```

```
  digitalWrite(11, 0);
  digitalWrite(12, 1);
  digitalWrite(13, 0);
  izquierda();
  delay(200);
  if (x < Xmin){
    izquierda();
    delay(200);
  }
  else if (x > Xmax){
    derecha();
    delay(200);
  }
  else if(area < minArea){
    adevel1();
    delay(200);
  }
  else if(area > maxArea){
    alto();
    delay(200);
  }
 }
 if((dis1>20)&&(dis2<=20)){
  digitalWrite(10, 1);
  digitalWrite(11, 0);
  digitalWrite(12, 0);
  digitalWrite(13, 1);
  izquierda();
  delay(200);
  if (x < Xmin){
    izquierda();
    delay(200);
  }
  else if (x > Xmax){
    derecha();
    delay(200);
  }
  else if(area < minArea){
    adevel1();
    delay(200);
  }
  else if(area > maxArea){
    alto();
    delay(200);
  }
 }
 if((dis1<=20)&&(dis2<=20)){
  digitalWrite(10, 1);
  digitalWrite(11, 0);
  digitalWrite(12, 1);
  digitalWrite(13, 1);
```

```
    adevel1();
    delay(200);
    if (x < Xmin){
      izquierda();
      delay(200);
    }
    else if (x > Xmax){
      derecha();
      delay(200);
    }
    else if(area < minArea){
      adevel1();
      delay(200);
    }
    else if(area > maxArea){
      alto();
      delay(200);
    }
  }
}
```

8.2. Anexo 2. Circuito Control General



8.3. Anexo 3. Sensores ultrasónicos HC-SR04

Cytron Technologies

User's Manual

V1.0

May 2013

Index

1.0 INTRODUCTION

The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats or dolphins do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. From 2cm to 400 cm or 1" to 13 feet. It operation is not affected by sunlight or black material like Sharp rangefinders are (although acoustically soft materials like cloth can be difficult to detect). It comes complete with ultrasonic transmitter and receiver module.

Features:

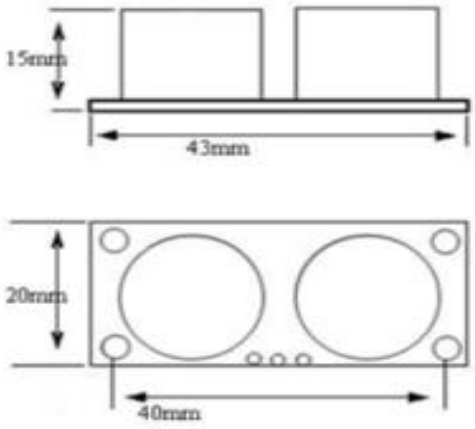
- Power Supply :+5V DC
- Quiescent Current : <2mA
- Working Currnt: 15mA
- Effectual Angle: <15°
- Ranging Distance : 2cm – 400 cm/1" - 13ft
- Resolution : 0.3 cm
- Measuring Angle: 30 degree
- Trigger Input Pulse width: 10uS
- Dimension: 45mm x 20mm x 15mm

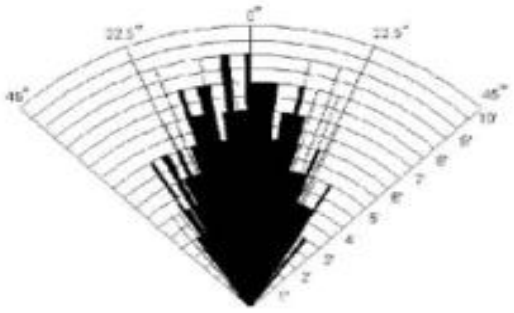
2.0 PACKING LIST

1. 1 x HC-SR04 module

3.0 PRODUCT LAYOUT

VCC = +5VDC
Trig = Trigger input of Sensor
Echo = Echo output of Sensor
GND = GND





Practical test of performance,
Best In 30 degree angle

4.0 PRODUCT SPECIFICATION AND LIMITATIONS

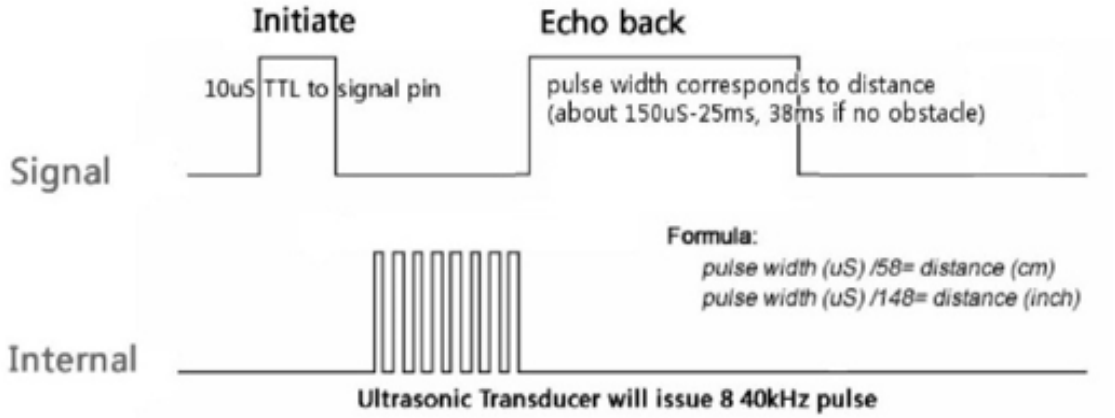
Parameter	Min	Typ.	Max	Unit
Operating Voltage	4.50	5.0	5.5	V
Quiescent Current	1.5	2	2.5	mA
Working Current	10	15	20	mA
Ultrasonic Frequency	-	40	-	kHz

5.0 OPERATION

The timing diagram of HC-SR04 is shown. To start measurement, Trig of SR04 must receive a pulse of high (5V) for at least 10us, this will initiate the sensor will transmit out 8 cycle of ultrasonic burst at 40kHz and wait for the reflected ultrasonic burst. When the sensor detected ultrasonic from receiver, it will set the Echo pin to high (5V) and delay for a period (width) which proportion to distance. To obtain the distance, measure the width (Ton) of Echo pin.

Time = Width of Echo pulse, in uS (micro second)

- Distance in centimeters = Time / 58
- Distance in inches = Time / 148
- Or you can utilize the speed of sound, which is 340m/s

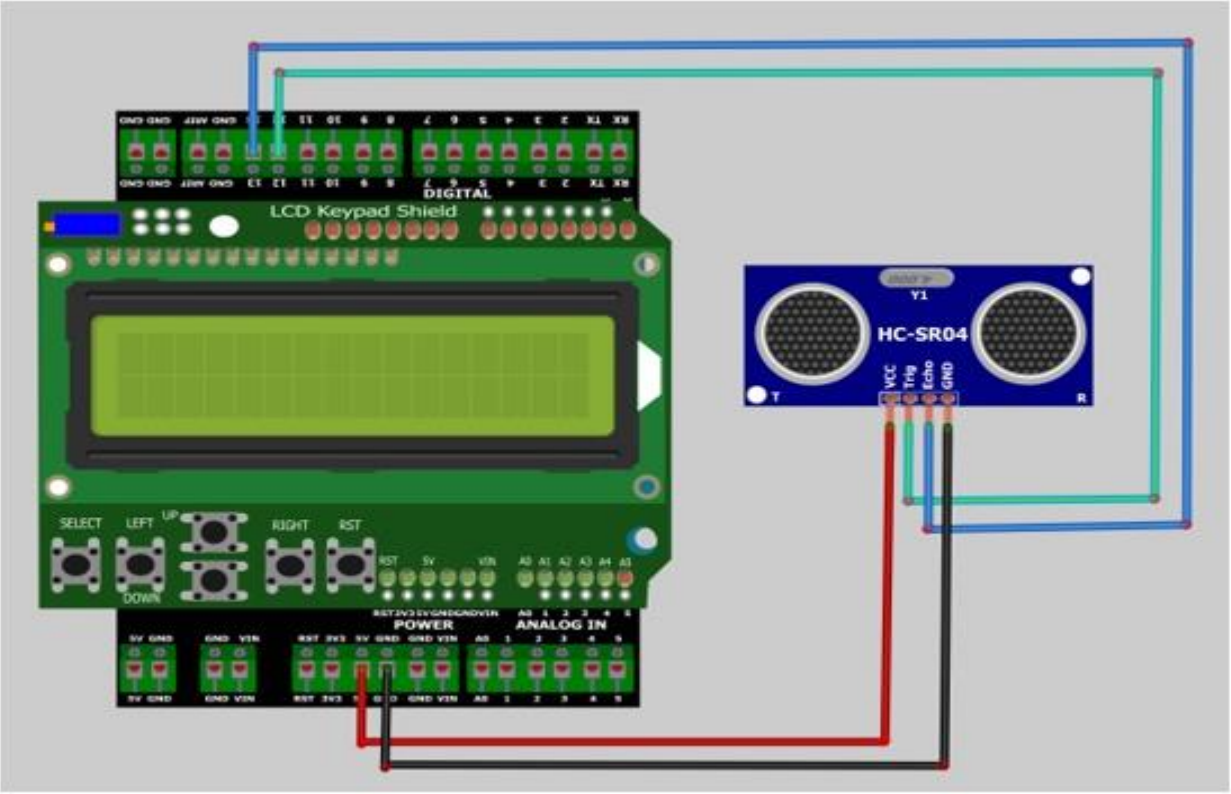


Note:

- Please connect the GND pin first before supplying power to VCC.
- Please make sure the surface of object to be detect should have at least 0.5 meter^2 for better performance.

6.0 HARDWARE INTERFACE

Here is example connection for Ultrasonic Ranging module to Arduino UNO board. It can be interface with any microcontroller with digital input such as PIC, SK40C, SK28A, SKds40A, Arduino series.



7.0 EXAMPLE CODE

This is example code Ultrasonic Ranging module. Please download the complete code at the product page.

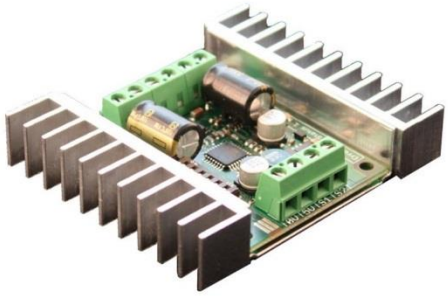
```
#include "Ultrasonic.h"
#include <LiquidCrystal.h>
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
Ultrasonic ultrasonic(12,13);

void setup() {
lcd.begin(16, 2);
lcd.setCursor(0, 0);
lcd.print("HC-SR4 testing..");
delay(1000);

}

void loop()
{
  //lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print(ultrasonic.Ranging(CM));
  lcd.print("cm  ");

  delay(100);
}
```

8.4. Anexo 4. Sabertooth 2x12

DimensionEngineering

Sabertooth 2x12 User's Guide

April 2012



Input voltage: 6-24V nominal, 30V absolute max.
Output current: Up to 12A continuous per channel. Peak loads may be up to 25A per channel for a few seconds. These ratings are for input voltages up to 18v in still air without additional heatsinking.
5V switching BEC: Up to 1A continuous and 1.5A peaks across the entire range of input voltages.

Recommended power sources are:

- 5 to 18 cells NiMH or NiCd
- 2s to 6s lithium ion or lithium polymer. Sabertooth motor drivers have a lithium battery mode to prevent cell damage due to over-discharge of lithium battery packs.
- 6v to 24v lead acid
- 6v to 24v power supply (when in parallel with a suitable battery).

Dimensions:

Size: 2.5" x 2.95" x .6"	64 x 75 x 16mm
Weight: 2.2oz	

Features

Mixed and independent options:
Sabertooth features mixed modes designed especially for differential drive robots, where two motors provide both steering and propulsion. It also has independent options in all operating modes. This is useful for if you have two motors to control, but they aren't necessarily being used to drive a differential drive robot. The motors do not need to be matched or even similar, as long as they both are within Sabertooth's operating limits.

Synchronous regenerative drive:
Going one step farther than just regenerative braking, a Sabertooth motor driver will return power to the battery any time a deceleration or motor reversal is commanded. This can lead to dramatic improvements in run time for systems that stop or reverse often, like a placement robot or a vehicle driving on hilly terrain. This drive scheme also saves power by returning the inductive energy stored in the motor windings to the battery each switching cycle, instead of burning it as heat in the motor windings. This makes part-throttle operation very efficient.

Ultra-sonic switching frequency:
Sabertooth 2x12 features a PWM frequency of 32kHz, which is well above the maximum frequency of human hearing. Unlike some other motor drivers, there is no annoying whine when the motor is on, even at low power levels.

Thermal and overcurrent protection:
Sabertooth features dual temperature sensors and overcurrent sensing. It will protect itself from failure due to overheating, overloading and motor shorts.

Easy mounting and setup:
Sabertooth has screw terminals for all inputs and outputs. There are four mounting holes, which accept 4-40 screws. Mounting hardware is included. All operating modes and options are set with DIP switches – there are no jumpers to struggle with or lose. No soldering is required.

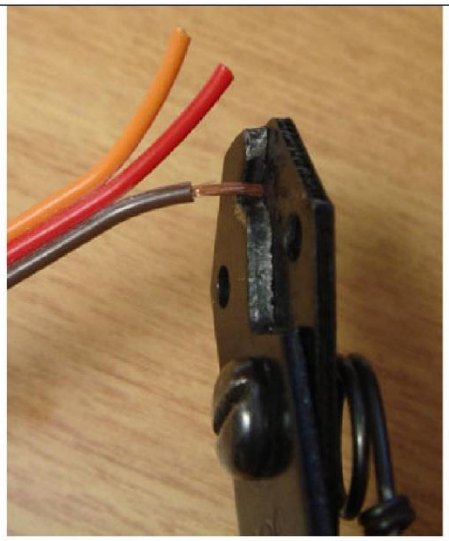
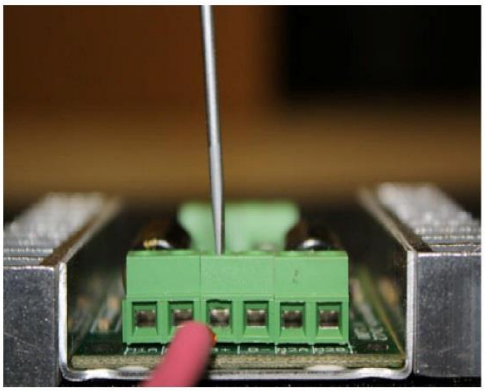
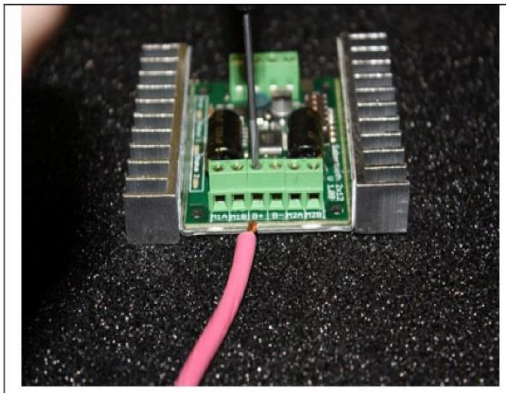
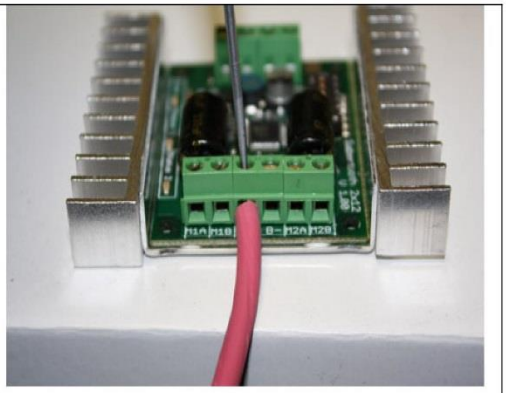
Compact size:
Sabertooth utilizes surface mount construction to provide the most power from a compact package. Its small size and light weight mean you have more space for cargo, batteries, or can make your robot smaller and more nimble than the competition.

Carefree reversing:
Unlike some other motor drivers, there is no need for the Sabertooth to stop before being commanded to reverse. You can go from full forward immediately to full reverse or vice versa. Braking and acceleration are proportional to the amount of reversal commanded, so gentle or rapid reversing is possible.

Many operating modes:
With analog, R/C and serial input modes, as well as dozens of operating options, the Sabertooth has the flexibility to be used over and over, even as your projects grow more sophisticated. Yet it is simple enough to use for your first robot project.

Hooking up the Sabertooth motor driver

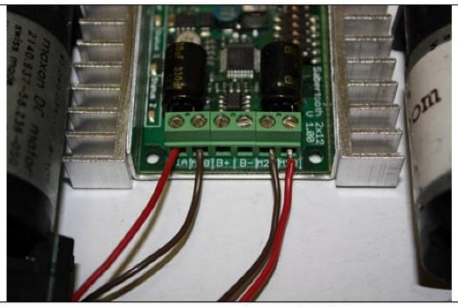
All connections to the Sabertooth are done with screw terminals. This makes it easy to set up and reconfigure your project. If you've never used screw terminal connections before, here is a quick overview.

	
Step 1: Strip the wire which you are using approximately ¼" The wires may be 14 gauge to 30 gauge	**Step 2:** With a small screwdriver, turn the top screw counter-clockwise until it stops gently.
	
Step 3: Insert the stripped portion of the wire into the opening in the screw terminal	**Step 4:** Turn the top screw clockwise until you encounter resistance, then tighten the screw firmly. Pull on the wire gently to ensure that it is secured.

Motor1 Terminals

Motor 1 is connected to terminals M1A and M1B as shown below. If the motor runs in the opposite way that you want, you may reverse the motor wires to reverse rotation.

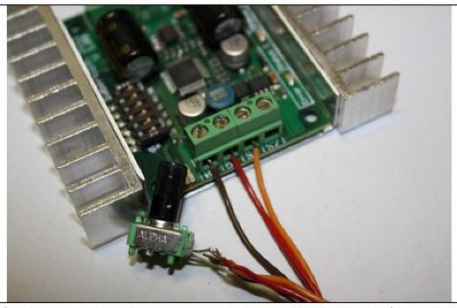
Motor 2 is connected to terminals M2A and M2B.



The motors connect to terminals M1A/B and M2A/B

Signal Input Terminals S1 and S2

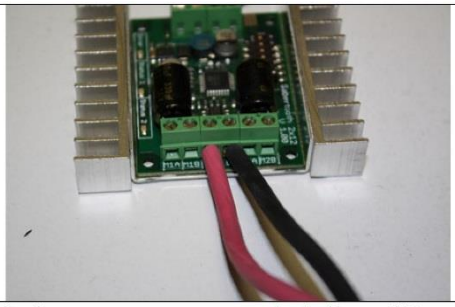
The input signals that control the Sabertooth are connected to terminals S1 and S2. If you are running in analog mode, it is important to have both the signal connected before applying power to the device. Otherwise, the motors may start unexpectedly.



The input signals connect to terminal S1 and/or S2

Battery Terminals B+ and B-

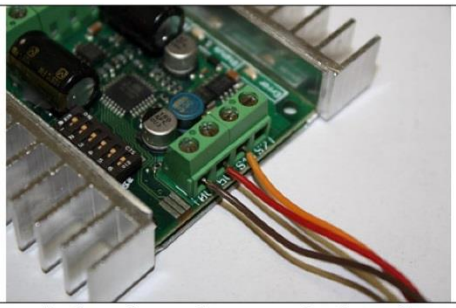
The battery or power supply is connected to terminals B- and B+. B- connects to the negative side of the battery (usually black.) B+ connects to the positive side of the battery (usually red or yellow.) It is usually best to connect the battery through a connector instead of directly to the motor driver. This makes it easy to unplug the battery for charging, and prevents plugging in the battery backwards.



The battery connects to terminals B+ and B-

Warning! Be very careful to wire and plug in the battery and connector correctly. Connecting the battery backwards will destroy the Sabertooth and will void the warranty.

Power terminals 0V and 5V



The 5V terminal can be used to power loads totaling no more than 1A, like a potentiometer or a radio receiver and a servo to two. The 0V signal must be connected to the ground of the device generating the input signal.

The 0V and 5V connections are used to power and interface to low-power control circuits.

The 5V connection is a 5v power **output**. The 2x12 utilizes a **1A switching BEC** to power the onboard electronics as well as to provide power to your receiver and up to 4 standard analog servos. You can power anything that requires 5V straight from the Sabertooth 2x12. There is no need for an external BEC unless you need more than 1 Amp! The BEC will work at full rated output throughout the Sabertooth's operating voltage range. You can use the BEC at full capacity whether you are running 7V or 24V in!

The 0V connection is the signal ground for the Sabertooth. In order to receive input signals correctly, it must be connected to the ground of the device sending the signals.

Using the 0V and 5v connections to power a radio receiver in R/C mode and potentiometer in analog mode is shown in Figures 2.1 and 2.2. ***If you are using multiple Sabertooths running from the same radio receiver, only one should have the 5v line connected.*** You can either take the red lead out of the connection housing or just clip the wire with a pair of cutters.

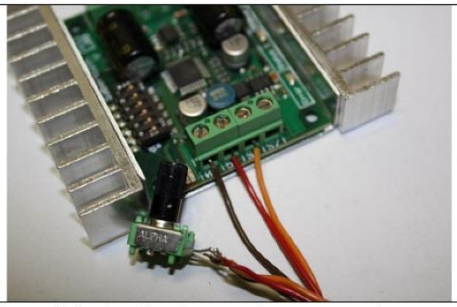


Figure 2.1: Analog input using a potentiometer powered from terminal 5V

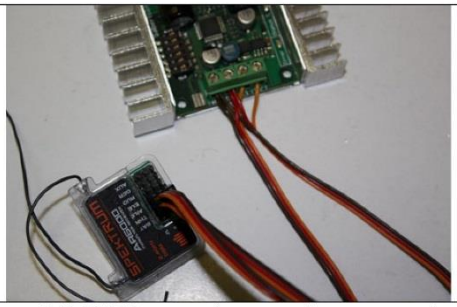
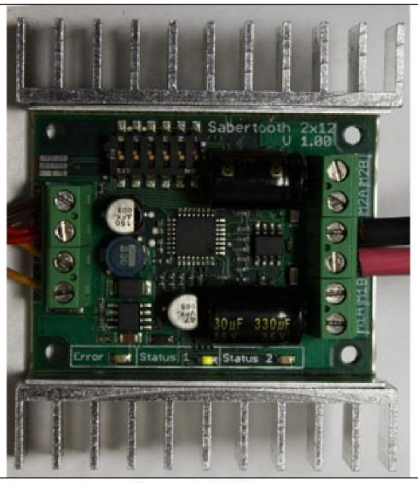


Figure 2.2: R/C input using a receiver powered from terminal 5V

Status and Error LEDs



Status1 LED on

Sabertooth 2x12 has three indicator LEDs.

The green LED marked Status1 is used to communicate various information about the current state. In most cases Status1 acts as a power indicator. In R/C mode, it glows dimly if there is no RC link present and brightly if there is an RC link.

The green LED marked Status2 is used in lithium mode. It blinks to indicate the number of lithium cells detected. Also, when Status2 and Error are flashing at the same time, and you are experiencing no output from the motors, the unit is displaying low voltage mode. Charge your batteries first, and if that does not work, you will need a larger battery. Keep in mind that when a lot of current is pulled from a battery, the voltage will drop. The more depleted the battery, the worse the voltage drop. The Sabertooth will hold the unit in the error state for longer than the battery voltage drops. This is to stop the unit from stuttering and gives the user enough time to diagnose the flashing LEDs.

The red Error LED illuminates if the Sabertooth has detected a problem. It will light if the driver has shut down due to overheating or overcurrent.

Mounting your Sabertooth 2x12

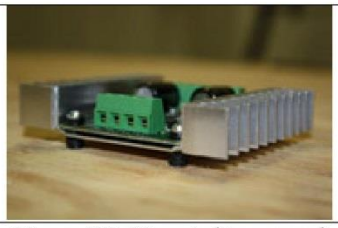


Figure 2.3: Mounted to a wood frame using standoffs

The Sabertooth is supplied with four mounting holes. These can be used to attach it to your robot. The centers of the mounting holes form a 1.5" x 2" rectangle. The holes are .125 inches in diameter. The proper size screw is a 4-40 round head machine or wood screw. Four 5/8" long machine screws and nuts are included.

If your robot or device is constructed from insulating materials such as wood or plastic, it may be necessary to mount the Sabertooth on standoffs to allow air to circulate. This is shown in Figure 2.3

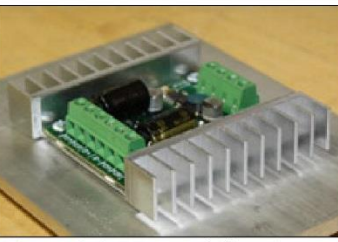


Figure 2.4: Mounted directly to a metal frame

If your robot or device is constructed from metal, it is usually better to attach the bottom heat spreader of the Sabertooth directly to the frame, without standoffs. This will allow your frame to act as a heat sink and will cause the Sabertooth to run cooler. If your chassis is grounded, you must insulate the heatsink from the chassis. This is shown in Figure 2.4

Operating Modes Overview

Mode 1: Analog Input

Analog input mode takes one or two analog inputs and uses those to set the speed and direction of the motor. The valid input range is 0v to 5v. This makes the Sabertooth easy control using a potentiometer, the PWM output of a microcontroller (with an RC filter) or an analog circuit. Major uses include joystick or foot-pedal controlled vehicles, speed and direction control for pumps and machines, and analog feedback loops.

Mode 2: R/C Input

R/C input mode takes two standard R/C channels and uses those to set the speed and direction of the motor. There is an optional timeout setting. When timeout is enabled, the motor driver will shut down on loss of signal. This is for safety and to prevent the robot from running away should it encounter interference and should be used if a radio is being used to control the driver. If timeout is disabled, the motor driver will continue to drive at the commanded speed until another command is given. This makes the Sabertooth easy to interface to a Basic Stamp or other low-speed microcontrollers.

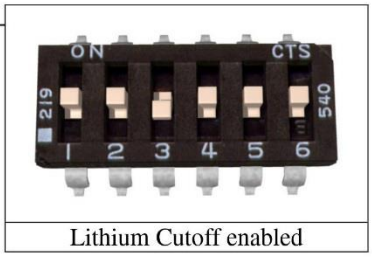
Mode 3: Simplified serial.

Simplified serial mode uses TTL level RS-232 serial data to set the speed and direction of the motor. This is used to interface the Sabertooth to a PC or microcontroller. If using a PC, a level converter such as a MAX232 chip must be used. A USB-to-TTL serial converter is also a viable option for the PC. The baud rate is set via DIP switches. Commands are single-byte. There is also a Slave Select mode which allows the use of multiple Sabertooth 2x12 from a single microcontroller serial port.

Mode 4: Packetized serial

Packetized serial mode uses TTL level RS-232 serial data to set the speed and direction of the motor. There is a short packet format consisting of an address byte, a command byte, a data byte and a 7 bit checksum. The baud rate is set to 9600 by default. See “Baud Rate Selection” later in this guide for information on changing the baud rate. Address bytes are set via DIP switches. Up to 8 Sabertooth motor drivers may be ganged together on a single serial line. This makes packetized serial the preferred method to interface multiple Sabertooths to a PC or laptop. Because Sabertooth uses the same protocol as our SyRen single motor drivers, both can use used together from the same serial master.

Lithium cutoff



Lithium Cutoff enabled

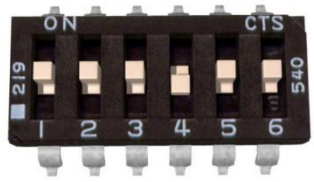
Switch 3 of the DIP switch block selects lithium cutoff. If switch 3 is in the down position as shown the Sabertooth will automatically detect the number of series lithium cells at startup, and set a cutoff voltage of 3.0 volts per cell. The number of detected cells is flashed out on the Status LED. If the number of cells detected is too low, your battery is in a severely discharged state and must be charged before operation. Failure to do so may cause damage to the battery pack. When 3.0V per cell is reached, the Sabertooth will shut down, preventing damage to the battery pack. This is necessary because a lithium battery pack discharged below 3.0v per cell will lose capacity and batteries discharged below 2.0v per cell may not ever recharge. Lithium cutoff mode may also be useful to increase the number of battery cycles you can get when running from a lead acid battery in non-critical applications. Because the system will continue to draw some power, even with the motor shut down, it is important to unplug the battery from the Sabertooth promptly once the cutoff is reached when using lithium batteries. If the Sabertooth is being run from NiCd, NiMH or alkaline batteries, or from a power supply, switch 3 should be in the up position.

Mode 1: Analog Input

Analog input mode is selected by setting switches 1 and 2 to the UP position. Switch 3 should be either up or down, depending on the battery type being used. Inputs S1 and S2 are configured as analog inputs. The output impedance of the signals fed into the inputs should be less than 10k ohms for best results. If you are using a potentiometer to generate the input signals, a 1k, 5k or 10k linear taper pot is recommended. In all cases, an analog voltage of 2.5V corresponds to no movement. Signals above 2.5V will command a forward motion and signals below 2.5V will command a backwards motion.

There are three operating options for analog input. These are selected with switches 4, 5 and 6. All the options can be used independently or in any combination.

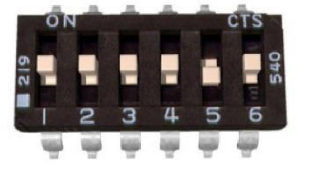
Switch 4: Mixed Mode



Switch 4: Mixed or independent

If switch 4 is in the UP position, the Sabertooth 2x12 is in **Mixed** mode. This mode is designed for easy steering of differential-drive vehicles. The analog signal fed into S1 controls the forward/back motion of the vehicle, and the analog signal fed into S2 controls the turning motion of the vehicle. If Switch 4 is in the DOWN position, the Sabertooth 2x12 is in Independent mode. In Independent mode, the signal fed to S1 directly controls Motor 1 (outputs M1A and M1B) and the signal fed to S2 controls Motor 2.

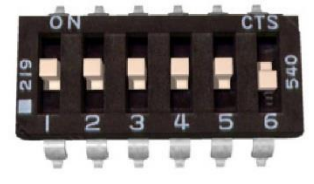
Switch 5: Exponential response



Switch 5: Exponential response

If switch 5 is in the DOWN position, the response to input signals will be exponential. This softens control around the zero speed point, which is useful for control of vehicles with fast top speeds or fast max turning rates. If switch 5 is in the UP position, the response is linear.

Switch 6: 4x sensitivity



Switch 6: 4x sensitivity

If switch 6 is in the UP position, the input signal range is from 0v to 5v, with a zero point of 2.5v.
If switch 6 is in the DOWN position, 4x sensitivity mode is enabled. In this mode, the input signal range is from 1.875V to 3.125V, with a zero point of 2.5v. This is useful for building analog feedback loops

Note on using filtered PWM in Analog Mode

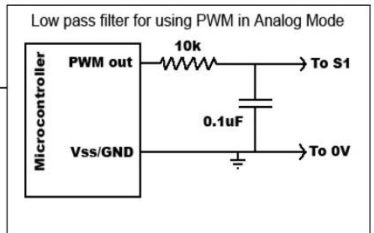


Figure 4.1: Filtered PWM

If you are using a filtered PWM signal from a microcontroller to generate the analog voltage, an R/C filter with component values 10k ohms and at least .1uf is recommended as shown in **Figure 4.1.** Using a larger value filter capacitor such as 1uf or 10uf will result in smoother motor operation, at a cost of slower transient response. A PWM frequency higher than 1000Hz is recommended.

Mode 2: R/C Input

R/C input mode is used with a standard hobby Radio control transmitter and receiver, or a microcontroller using the same protocol. R/C mode is selected by setting switch 1 to the DOWN position and switch 2 to the UP position. If running from a receiver, it is necessary to obtain one or more servo pigtails and hook them up according to figure 5.1. If using a receiver pack, do not connect power to the 5V line of the Sabertooth because the maximum voltage it can tolerate is 6V.

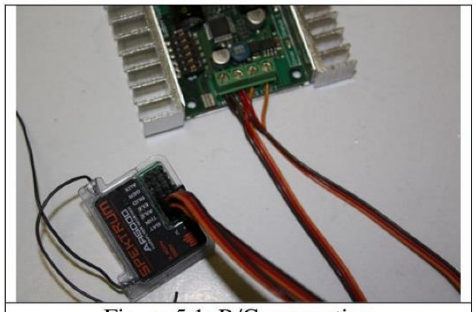
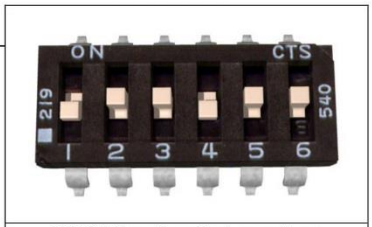


Figure 5.1: R/C connection

There are three operating options for R/C mode. These are selected with switches 4, 5 and 6.

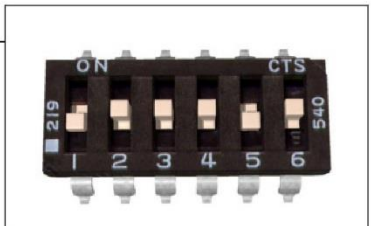
Switch 4: Mixed Mode



R/C Mixed or Independent

When Switch 4 is in the UP position, Mixed mode is selected. In this mode, the R/C signal fed to the S1 input controls the forward/backwards motion of the vehicle. This is usually connected to the throttle channel of a pistol grip transmitter, or the elevator channel of a dual stick transmitter. The R/C signal fed to the S2 input controls the turning of the vehicle.
When switch 4 is in the DOWN position, Independent mode is selected. In this mode, the signal fed to the S1 input directly controls Motor 1 (M1A and M1B) and the signal fed to S2 controls Motor 2.

Switch 5: Exponential response



Exponential mode enabled

If switch 5 is in the UP position, the response is linear.
If switch 5 is in the DOWN position, the response to input signals will be exponential. This softens control around the zero speed point, which is useful for control of vehicles with fast top speeds or fast max turning rates.

Mode 2: R/C Input

R/C input mode is used with a standard hobby Radio control transmitter and receiver, or a microcontroller using the same protocol. R/C mode is selected by setting switch 1 to the DOWN position and switch 2 to the UP position. If running from a receiver, it is necessary to obtain one or more servo pigtails and hook them up according to figure 5.1. If using a receiver pack, do not connect power to the 5V line of the Sabertooth because the maximum voltage it can tolerate is 6V.

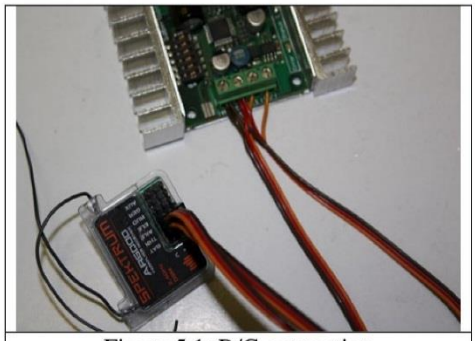
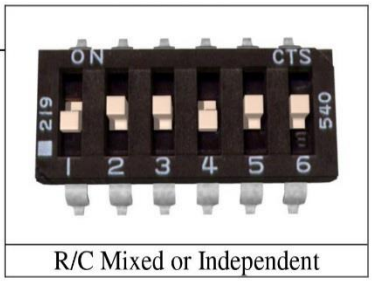


Figure 5.1: R/C connection

There are three operating options for R/C mode. These are selected with switches 4, 5 and 6.

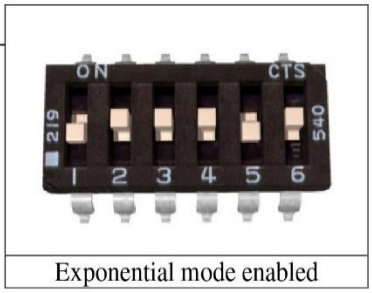
Switch 4: Mixed Mode



R/C Mixed or Independent

When Switch 4 is in the UP position, Mixed mode is selected. In this mode, the R/C signal fed to the S1 input controls the forward/backwards motion of the vehicle. This is usually connected to the throttle channel of a pistol grip transmitter, or the elevator channel of a dual stick transmitter. The R/C signal fed to the S2 input controls the turning of the vehicle.
When switch 4 is in the DOWN position, Independent mode is selected. In this mode, the signal fed to the S1 input directly controls Motor 1 (M1A and M1B) and the signal fed to S2 controls Motor 2.

Switch 5: Exponential response

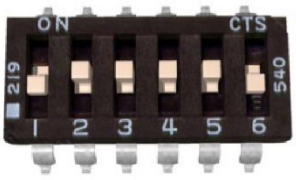


Exponential mode enabled

If switch 5 is in the UP position, the response is linear.
If switch 5 is in the DOWN position, the response to input signals will be exponential. This softens control around the zero speed point, which is useful for control of vehicles with fast top speeds or fast max turning rates.

Switch 6: R/C Mode/Microcontroller mode select

If switch 6 is in the UP position, then the Sabertooth is in standard R/C mode. This mode is designed to be used with a hobby-style transmitter and receiver. It automatically calibrates the control center and endpoints to maximize stick usage. It also enables a Timeout Failsafe, which will shut down the motors if the Sabertooth stops receiving correct signals from the receiver.



Microcontroller mode selected

If switch 6 is set in the DOWN position, then Microcontroller mode is enabled. This disables the Timeout Failsafe and auto-calibration. This means that the Sabertooth will continue to drive the motor according to the last command until another command is given. If the control link is possible unreliable – like a radio - then this can be dangerous due to the robot not stopping. However, it is extremely convenient if you are controlling the Sabertooth from a microcontroller. In this case, commanding the controller can be done with as little as three lines of code.

Output_High(Pin connected to S1)
Delay(1000us to 2000us)
Output_Low(Pin connected to S1)

A note on certain microprocessor receivers

Some receivers, such as the Spektrum AR6000, will output servo pulses before a valid transmitter signal is present. This will cause the Sabertooth to autocalibrate to the receiver's startup position which may not correspond to the center stick position, depending on trim settings. This may cause the motors to move slowly, even when the transmitter stick is centered. If you encounter this, either consult your receiver manual to reprogram the startup position, or adjust your transmitter trims until the motors stop moving. As a last resort, you can enter R/C microcontroller mode which will disable Sabertooth's autocalibration.

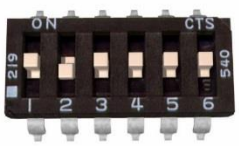
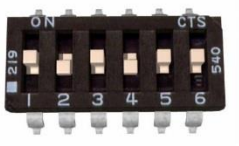
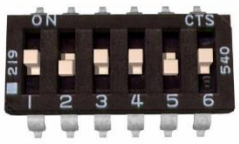
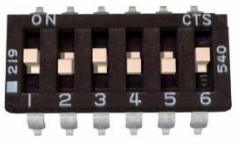
Mode 3: Simplified Serial Mode

Simplified serial uses TTL level single-byte serial commands to set the motor speed and direction. This makes it easy to interface to microcontrollers and PCs, without having to implement a packet-based communications protocol. Simplified serial is a one-direction only interface. The transmit line from the host is connected to S1. The host's receive line is not connected to the Sabertooth. Because of this, multiple drivers can be connected to the same serial transmitter. If using a true RS-232 device like a PC's serial port, it is necessary to use a level converter to shift the –10V to 10V rs-232 levels to the 0v-5v TTL levels the Sabertooth is expecting. This is usually done with a Max232 type chip. If using a TTL serial device like a microcontroller or a USB-to-TTL serial converter, the TX line of the microcontroller may be connected directly to S1.

Because Sabertooth controls two motors with one 8 byte character, when operating in Simplified Serial mode, each motor has 7 bits of resolution. Sending a character between 1 and 127 will control motor 1. 1 is full reverse, 64 is stop and 127 is full forward. Sending a character between 128 and 255 will control motor 2. 128 is full reverse, 192 is stop and 255 is full forward. Character 0 (hex 0x00) is a special case. Sending this character will shut down both motors.

Baud Rate Selection

Simplified Serial operates with an 8N1 protocol – 8 data bytes, no parity bits and one stop bit. The baud rate is selected by switches 4 and 5 from the following 4 options:

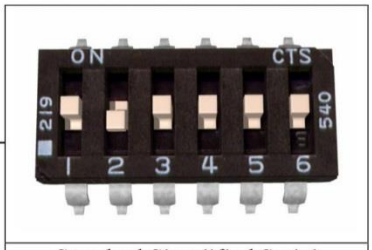
	
2400 Baud: 01x00x	9600 Baud: 01x10x
	
19200 Baud: 01x01x	38400 Baud: 01x11x

What baud rate to use is dependent on what your host can provide and the update speed necessary. 9600 baud or 19200 baud is recommended as the best starting points. If communication is unreliable, decrease the baud rate. If communications are reliable, you may increase the baud rate. The maximum update speed on the Sabertooth is approximately 2000 commands per second. Sending characters faster than this will not cause problems, but it will not increase the responsiveness of the controller either.

The baud rate may be changed with power on by changing the DIP switch settings. There is no need to reset or cycle power after a baud rate change.

There are 2 operating options for Simplified Serial, selected by the position of Switch 6.

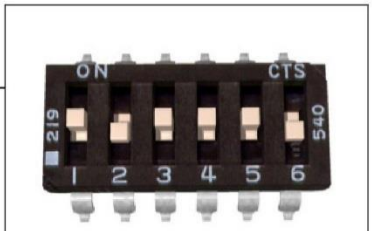
Option 1: Standard Simplified Serial Mode



Standard Simplified Serial

Serial data is sent to input S1. The baud rate is selected with switches 4 and 5. Commands are sent as single bytes. Sending a value of 1-127 will command motor 1 Sending a value of 128-255 will command motor 2. Sending a value of 0 will shut down both motors.

Option 2: Simplified Serial with Slave Select



Simplified Serial with Slave Select

This mode is used when it is desirable to have multiple Sabertooth motor drivers running from the same serial transmitter, but you do not wish to use packetized serial. A digital signal (0v or 5v) is fed to the S2 input. This is controlled by the host microcontroller. If the signal on S2 is logic high (5v) when the serial command is sent, the driver will change to the new speed. If the signal on S2 is not high when the command is sent, then command will be ignored. Pseudo-code demonstrating this is shown below. After sending the signal, allow about 50 us before commanding the Slave Select line to a logic LOW to allow time for processing. A hookup diagram and example pseudo-code are shown in **Figures 6.2** and **6.3**.

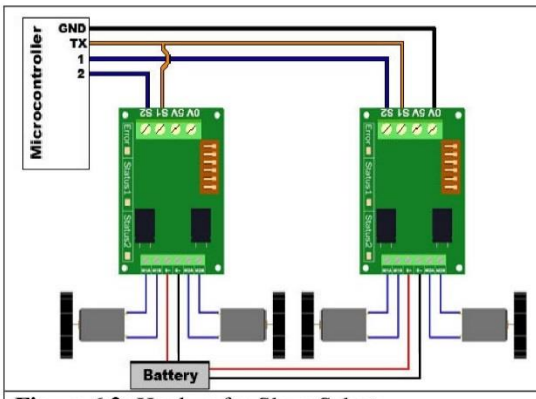


Figure 6.2: Hookup for Slave Select

```
//set controller 1's speed
Output_High (S2 pin on controller 1)
USART_TX(controller 1 speed, 0 to 255)
Delay_us(50)
Output_Low (S2 pin on controller 1)

//set controller 2's speed
Output_High (S2 pin on controller 2)
USART_TX(controller 2 speed, 0 to 255)
Delay_us(50)
Output_Low (S2 pin on controller 2)
```

Figure 6.3: Pseudocode for Slave Select

Mode 4: Packetized Serial Mode

Packetized Serial uses TTL level multi-byte serial commands to set the motor speed and direction. Packetized serial is a one-direction only interface. The transmit line from the host is connected to S1. The host's receive line is not connected to the Sabertooth. Because of this, multiple Sabertooth 2x12 motor drivers can be connected to the same serial transmitter. It is also possible to use SyRen and Sabertooth motor drivers together from the same serial source, as well as any other serial device, as long as it will not act on the packets sent to the Sabertooth. If using a true RS-232 device like a PC's serial port, it is necessary to use a level converter to shift the –10V to 10V RS-232 levels to the 0V to 5V TTL. Packetized serial uses an address byte to select the target device.

Packet Overview

The packet format for the Sabertooth consists of an address byte, a command byte, a data byte and a seven bit checksum. Address bytes have value greater than 128, and all subsequent bytes have values 127 or lower. This allows multiple types of devices to share the same serial line.

An example packet and pseudo-code to generate it are shown in **Figures 7.1** and **7.2**

Packet
Address: 130
Command : 0
Data: 64
Checksum: 66
	void DriveForward(char address, char speed)
	{
	Putc(address);
	Putc(0);
	Putc(speed);
	Putc((address + 0 + speed) & 127);
	}

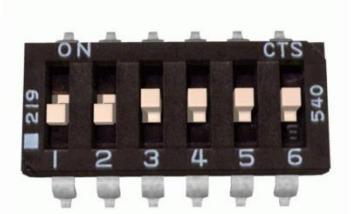
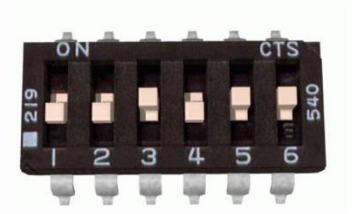
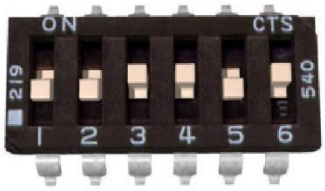
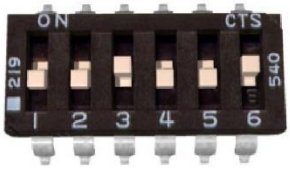
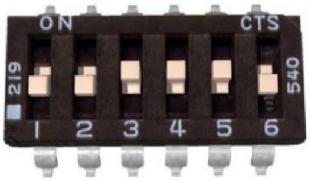
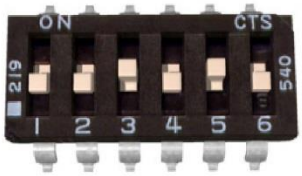
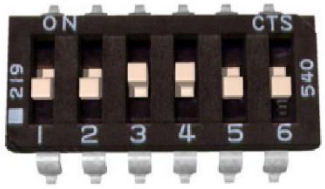
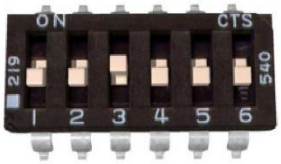
Figure 7.1: Example 50% forward	Figure 7.2: Pseudocode to generate 7.1

Baud Rate Selection

Packetized Serial operates with an 8N1 protocol – 8 data bytes, no parity bits and one stop bit. The baud rate is set at **9600 baud** from the factory. This value can be changed by sending the proper baud rate selection packet once the unit has powered on. Changed baud rates will be active after a power cycle. Once you set it, it stays that way until you change the rate again. See the 'Setting Commands' page for further details on how to change the baud rate.

Address Byte Configuration

Address bytes are set by switches 4, 5 and 6. Addresses start at 128 and go to 135. The switch settings for the addresses are shown in the chart below

	
Address: 128	Address: 129
	
Address: 130	Address: 131
	
Address: 132	Address: 133
	
Address: 134	Address: 135

Commands

The command byte is the second byte of the packet. There are four possible commands in packetized serial mode. Each is followed by one byte of data

0: Drive forward motor 1 (decimal 0, binary 0b00000000, hex 0h00)

This is used to command motor 1 to drive forward. Valid data is 0-127 for off to full forward drive. If a command of 0 is given, the Sabertooth will go into power save mode for motor 1 after approximately 4 seconds.

1: Drive backwards motor 1 (decimal 1, binary 0b00000001, hex 0h01)

This is used to command motor 1 to drive backwards. Valid data is 0-127 for off to full reverse drive. If a command of 0 is given, Sabertooth will go into power save mode for motor 1 after approximately 4 seconds.

2: Min voltage (decimal 2, binary 0b00000010, hex 0h02)

This is used to set a custom minimum voltage for the battery feeding the Sabertooth. If the battery voltage drops below this value, the output will shut down. This value is cleared at startup, so much be set each run. The value is sent in .2 volt increments with a command of zero corresponding to 6v, which is the minimum. Valid data is from 0 to 120. The function for converting volts to command data is

Value = (desired volts-6) x 5

3: Max voltage (decimal 3, binary 0b0000011, hex 0h03)

This is used to set a custom maximum voltage. If you are using a power supply that cannot sink current such as an ATX supply, the input voltage will rise when the driver is regenerating (slowing down the motor) Many ATX type supplies will shut down if the output voltage on the 12v supply rises beyond 16v. If the driver detects an input voltage above the set limit, it will put the motor into a hard brake until the voltage drops below the set point again. This is inefficient, because the energy is heating the motor instead of recharging a battery, but may be necessary. The driver comes preset for a maximum voltage of 30V. The range for a custom maximum voltage is 0v-25v. The formula for setting a custom maximum voltage is

Value = Desired Volts*5.12

If you are using any sort of battery, then this is not a problem and the max voltage should be left at the startup default.

4: Drive forward motor 2 (decimal 4, binary 0b00000100, hex 0h04)

This is used to command motor 2 to drive forward. Valid data is 0-127 for off to full forward drive. If a command of 0 is given, the Sabertooth will go into power save mode for motor 2 after approximately 4 seconds.

5: Drive backwards motor 2 (decimal 5, binary 0b00000101, hex 0h05)

This is used to command motor 2 to drive backwards. Valid data is 0-127 for off to full reverse drive. If a command of 0 is given, the Sabertooth will go into power save mode after approximately 4 seconds.

6: Drive motor 1 7 bit (decimal 6, binary 0b00000110, hex 0h06)
This command is used to drive motor 1. Instead of the standard commands 0 and 1, this one command can be used to drive motor 1 forward or in reverse, at a cost of lower resolution. A command of 0 will correspond to full reverse, and a command of 127 will command the motor to drive full forward. A command of 64 will stop the motor.

7: Drive motor 2 7 bit (decimal 7, binary 0b00000111, hex 0h07)
This command is used to drive motor 2. Instead of the standard commands 4 and 5, this one command can be used to drive motor 1 forward or in reverse, at a cost of lower resolution. A command of 0 will correspond to full reverse, and a command of 127 will command the motor to drive full forward. A command of 64 will stop the motor.

Mixed mode commands

Sabertooth can also be sent mixed drive and turn commands. When using the mixed mode commands, please note that the Sabertooth requires valid data for both drive and turn before it will begin to operate. Once data for both has been sent, then each may be updated as needed, it is not necessary to send both data packets each time you with to update the speed or direction. You should design your code to either use the independent or the mixed commands. Switching between the command sets will cause the vehicle to stop until new data is sent for both motors.

8: Drive forward mixed mode (decimal 8, binary 0b00001000, hex 0h08)
This is used to command the vehicle to drive forward in mixed mode. Valid data is 0-127 for off to full forward drive.

9: Drive backwards mixed mode (decimal 9, binary 0b00001001, hex 0h09)
This is used to command the vehicle to drive backwards in mixed mode. Valid data is 0-127 for off to full reverse drive.

10: Turn right mixed mode (decimal 10, binary 0b00001010, hex 0h0a)
This is used to command the vehicle to turn right in mixed mode. Valid data is 0-127 for zero to maximum turning speed.

11: Drive turn left mixed mode (decimal 11, binary 0b00001011, hex 0h0b)
This is used to command the vehicle to turn leftt in mixed mode. Valid data is 0-127 for zero to maximum turning speed.

12: Drive forwards/back 7 bit (decimal 12, binary 0b00001100, hex 0h0c)
This is used to command the vehicle to move forwards or backwards. A command of 0 will cause maximum reverse, 64 will cause the vehicle to stop, and 127 will command full forward.

13: Turn 7 bit (decimal 13, binary 0b00001101, hex 0h0d)
This is used to command the vehicle turn right or left. A command of 0 will cause maximum left turn rate, 64 will cause the vehicle to stop turning , and 127 will command maximum right turn rate.

Setting Commands

Several parameters of the Sabertooth 2x12 can be changed using Packetized Serial mode. Some of these changes persist when the unit is powercycled and some persist when it is switched to other modes.

14: Serial Timeout (decimal 14, binary 0b00001110, hex 0h0e)

This setting determines how long it takes for the motor driver to shut off if it has not received a command recently. Serial Timeout is off by default. A command of 0 will disable the timeout if you had previously enabled it. The timeout scales 1 unit per 100ms of timeout, so a command of 10 would make a timeout of 1000ms. This setting does **not** persist through a power cycle or in any mode other than Serial.

15: Baud Rate (decimal 15, binary 0b00001111, hex 0h0f)

This value remains until it is changed and does persist through a power cycle. The values are:
1: 2400 baud
2: 9600 baud (default)
3: 19200 baud
4: 38400 baud

16: Ramping (decimal 16, binary 0b00010000, hex 0h10)

This adjusts or disables the ramping feature found on the Sabertooth 2x50. This adjustment applies to all modes, even R/C and analog mode. Values between 1 and 10 are **Fast Ramp**; values between 11 and 20 are **Slow Ramp**; values between 21 and 80 are **Intermediate Ramp**. **Fast Ramping** is a ramp time of 256/(~1000xCommand value). Ramp time is the delay between full forward and full reverse speed.
1: 1/4 second ramp (default)
2: 1/8 second ramp
3: 1/12 second ramp
Slow and Intermediate Ramping are a ramp time of 256/[15.25x(Command value – 10)]
See **Figures 8.1 and 8.2** in the Appendix for a graph of values.

17: Deadband (decimal 17, binary 0b00010001, hex 0h11)

This determines the extent of the Sabertooth's deadband – the range of commands close to "stop" that will be interpreted as stop. This setting applies to all modes and persists through a power cycle. The commands range from 0 to 127 and the formula is as follows:
127-command < motors off < 128+command
Thus, a command of 3 would shut the motors off with speed commands between 124 and 131.

A command of 0 sets the deadband to its default, which is 124 < off < 131 in serial mode.

Checksum

To prevent data corruption, each packet is terminated with a checksum. If the checksum is not correct, the data packet will not be acted upon. The checksum is calculated as follows:

Checksum = address byte + command byte + data byte

The checksum should be added with all unsigned 8 bit integers, and then ANDed with the mask 0b01111111 (decimal 127) in an 8 bit system.

Example of Packetized Serial

The following is an example function for commanding two Dimension Engineering motor drivers using Packetized Serial Mode. **Figure 7.3** shows an example hookup and **Figure 7.4** shows an example function.

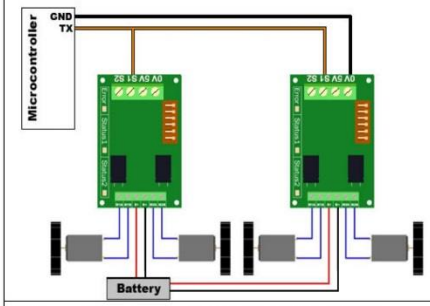


Figure 7.3: Packetized serial hookup

```
void DriveForward(char address, char speed)
{
  Putc(address);
  Putc(0);
  Putc(speed);
  Putc((address + 0 + speed) & 127);
}
```

Figure 7.4: Packetized Serial Function

Example: So in this function, if address is 130, command is 0 (for driving forward), speed is 64, the checksum should calculate as follows:
130+0+64 = 194
194 in binary is 0b11000010
0b11000010 & 0b01111111 = 0b01000010

Once all the data is sent, this will result in the Sabertooth with address 130 driving forward at roughly half throttle.

Emergency Stop

In Packetized Serial mode, the S2 input is configured as an active-low emergency stop. It is pulled high internally, so if this feature isn't needed, it can be ignored. If an emergency stop is desired, all the S2 inputs can be tied together. Pulling the S2 input low will cause the driver to shut down. This should be tied to an emergency stop button if used in a device that could endanger humans.

Appendix

Figure 8.1: Fast and Intermediate Ramp

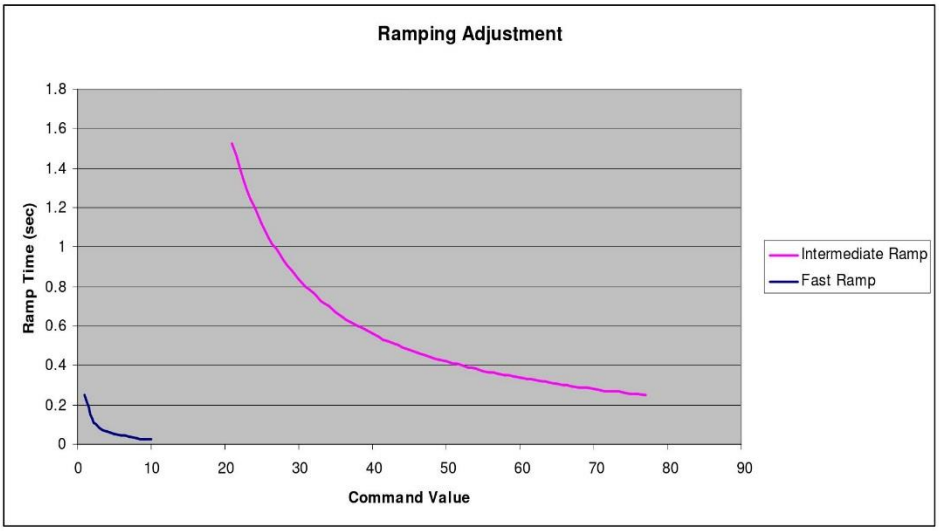
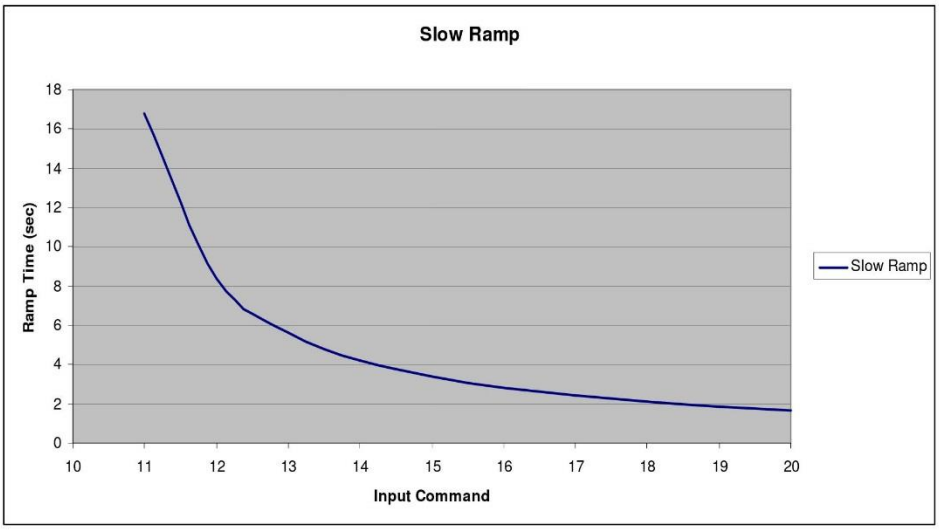


Figure 8.2: Slow Ramp



8.5. Anexo 5. Microcontrolador 2560

Features

- High Performance, Low Power Atmel® AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 135 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16MHz
  - On-Chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 64K/128K/256KBytes of In-System Self-Programmable Flash
  - 4Kbytes EEPROM
  - 8Kbytes Internal SRAM
  - Write/Erase Cycles:10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/ 100 years at 25°C
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
    - Endurance: Up to 64Kbytes Optional External Memory Space
- Atmel® QTouch® library support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix® acquisition
  - Up to 64 sense channels
- JTAG (IEEE std. 1149.1 compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - Four 16-bit Timer/Counter with Separate Prescaler, Compare- and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four 8-bit PWM Channels
  - Six/Twelve PWM Channels with Programmable Resolution from 2 to 16 Bits (ATmega1281/2561, ATmega640/1280/2560)
  - Output Compare Modulator
  - 8/16-channel, 10-bit ADC (ATmega1281/2561, ATmega640/1280/2560)
  - Two/Four Programmable Serial USART (ATmega1281/2561, ATmega640/1280/2560)
  - Master/Slave SPI Serial Interface
  - Byte Oriented 2-wire Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 54/86 Programmable I/O Lines (ATmega1281/2561, ATmega640/1280/2560)
  - 64-pad QFN/MLF, 64-lead TQFP (ATmega1281/2561)
  - 100-lead TQFP, 100-ball CBGA (ATmega640/1280/2560)
  - RoHS/Fully Green
- Temperature Range:
  - -40°C to 85°C Industrial
- Ultra-Low Power Consumption
  - Active Mode: 1MHz, 1.8V: 500µA
  - Power-down Mode: 0.1µA at 1.8V
- Speed Grade:
  - ATmega640V/ATmega1280V/ATmega1281V:
    - 0 - 4MHz @ 1.8V - 5.5V, 0 - 8MHz @ 2.7V - 5.5V
  - ATmega2560V/ATmega2561V:
    - 0 - 2MHz @ 1.8V - 5.5V, 0 - 8MHz @ 2.7V - 5.5V
  - ATmega640/ATmega1280/ATmega1281:
    - 0 - 8MHz @ 2.7V - 5.5V, 0 - 16MHz @ 4.5V - 5.5V
  - ATmega2560/ATmega2561:
    - 0 - 16MHz @ 4.5V - 5.5V

8-bit Atmel Microcontroller with 64K/128K/256K Bytes In-System Programmable Flash

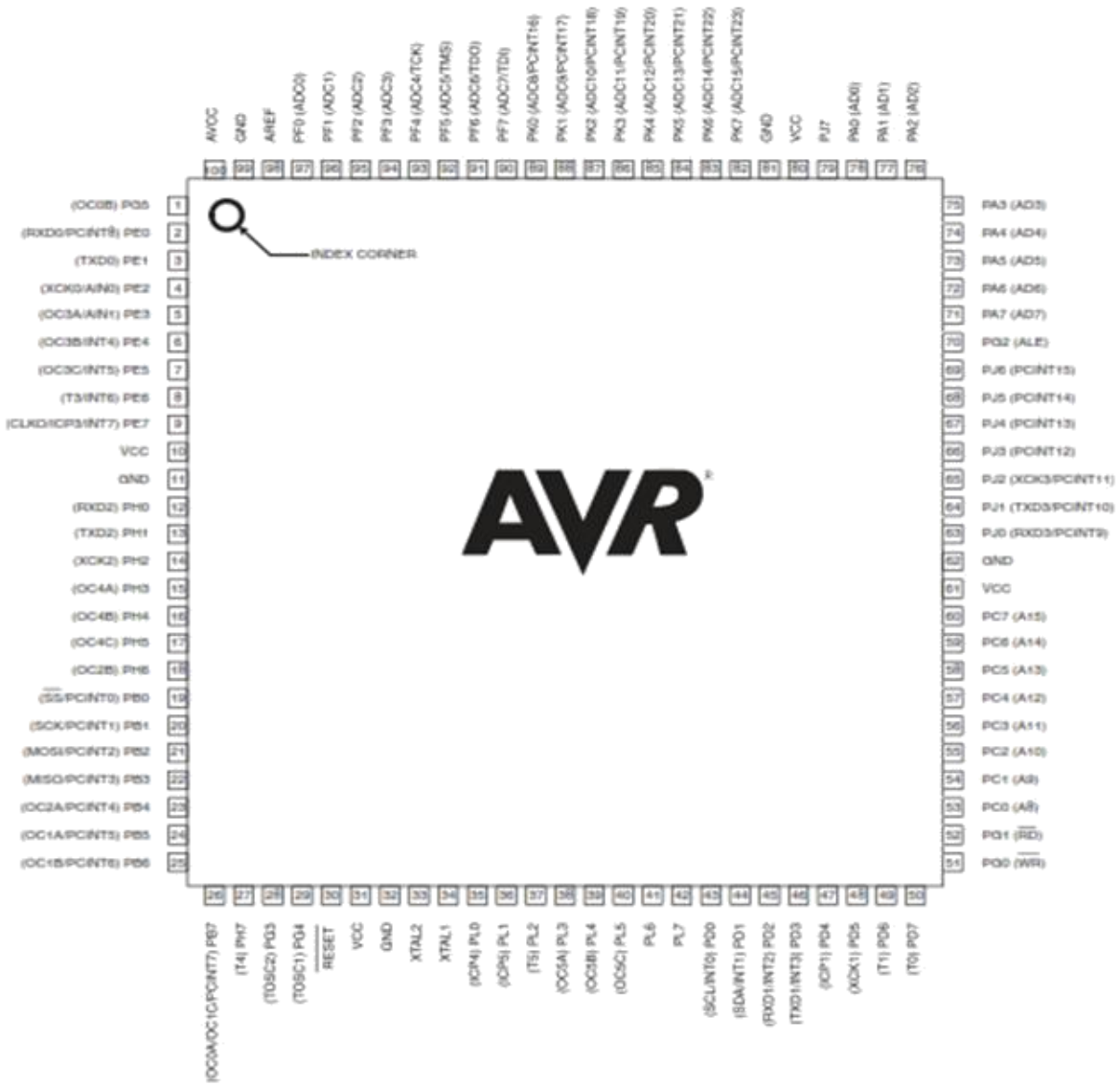
ATmega640/V
ATmega1280/V
ATmega1281/V
ATmega2560/V
ATmega2561/V

Preliminary Summary

2549NS–AVR–05/11

1. Pin Configurations

Figure 1-1. TQFP-pinout ATmega640/1280/2560

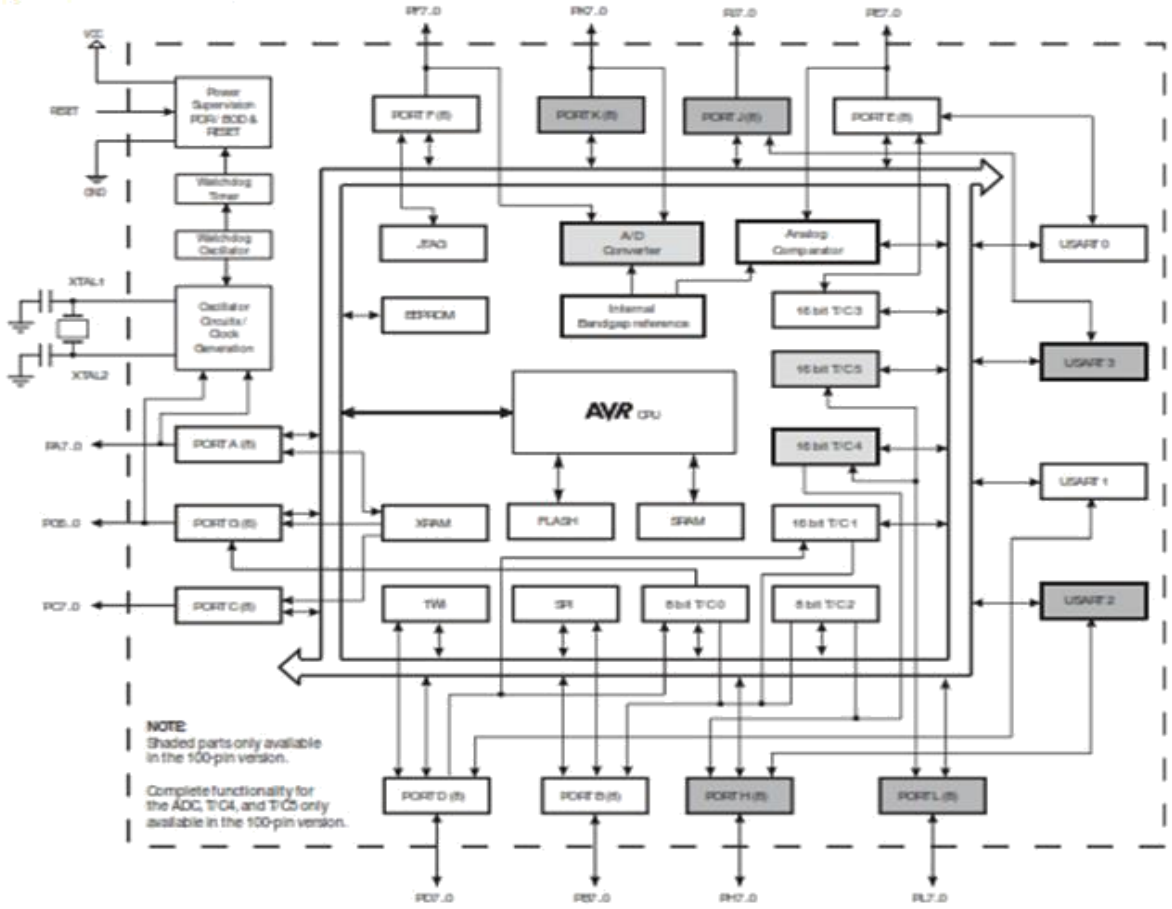


2. Overview

The ATmega640/1280/1281/2560/2561 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega640/1280/1281/2560/2561 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



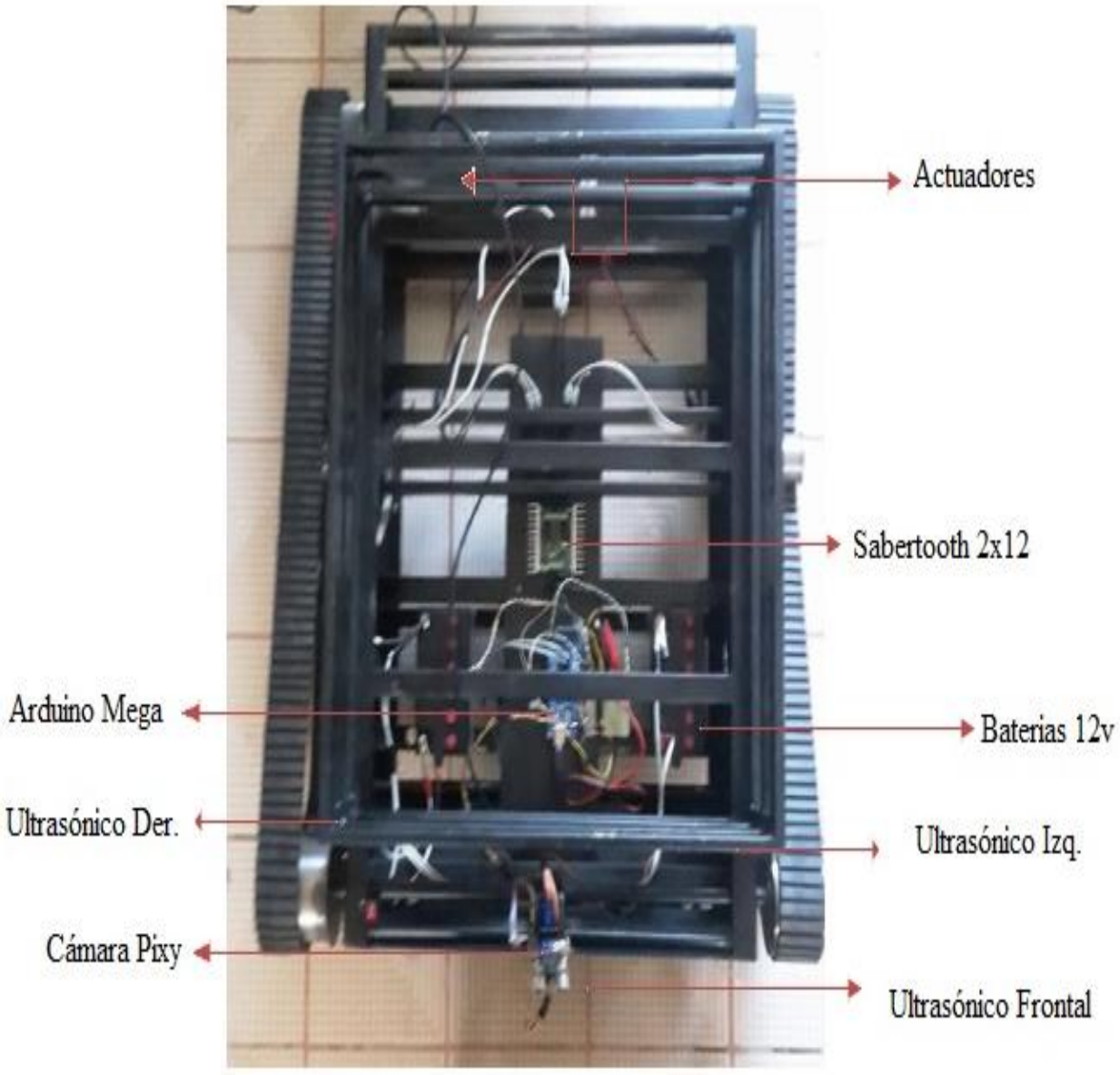
8.6. Anexo 6. Distribución chi-cuadrado

TABLA 3-Distribución Chi Cuadrado χ^2

P = Probabilidad de encontrar un valor mayor o igual que el chi cuadrado tabulado, v = Grados de Libertad

v/p	0,001	0,0025	0,005	0,01	0,025	0,05	0,1	0,15	0,2	0,25	0,3	0,35	0,4	0,45	0,5
1	10,8274	9,1404	7,8794	6,6349	5,0239	3,8415	2,7055	2,0722	1,6424	1,3233	1,0742	0,8735	0,7083	0,5707	0,4549
2	13,8150	11,9827	10,5965	9,2104	7,3778	5,9915	4,6052	3,7942	3,2189	2,7726	2,4079	2,0996	1,8326	1,5970	1,3863
3	16,2660	14,3202	12,8381	11,3449	9,3484	7,8147	6,2514	5,3170	4,6416	4,1083	3,6649	3,2831	2,9462	2,6430	2,3660
4	18,4662	16,4238	14,8602	13,2767	11,1433	9,4877	7,7794	6,7449	5,9886	5,3853	4,8784	4,4377	4,0446	3,6871	3,3567
5	20,5147	18,3854	16,7496	15,0863	12,8325	11,0705	9,2363	8,1152	7,2893	6,6257	6,0644	5,5731	5,1319	4,7278	4,3515
6	22,4575	20,2491	18,5475	16,8119	14,4494	12,5916	10,6446	9,4461	8,5581	7,8408	7,2311	6,6948	6,2108	5,7652	5,3481
7	24,3213	22,0402	20,2777	18,4753	16,0128	14,0671	12,0170	10,7479	9,8032	9,0371	8,3834	7,8061	7,2832	6,8000	6,3458
8	26,1239	23,7742	21,9549	20,0902	17,5345	15,5073	13,3616	12,0271	11,0301	10,2189	9,5245	8,9094	8,3505	7,8325	7,3441
9	27,8767	25,4625	23,5893	21,6660	19,0228	16,9190	14,6837	13,2880	12,2421	11,3887	10,6564	10,0060	9,4136	8,8632	8,3428
10	29,5879	27,1119	25,1881	23,2093	20,4832	18,3070	15,9872	14,5339	13,4420	12,5489	11,7807	11,0971	10,4732	9,8922	9,3418
11	31,2635	28,7291	26,7569	24,7250	21,9200	19,6752	17,2750	15,7671	14,6314	13,7007	12,8987	12,1836	11,5298	10,9199	10,3410
12	32,9092	30,3182	28,2997	26,2170	23,3367	21,0261	18,5493	16,9893	15,8120	14,8454	14,0111	13,2661	12,5838	11,9463	11,3403
13	34,5274	31,8830	29,8193	27,6882	24,7356	22,3620	19,8119	18,2020	16,9848	15,9839	15,1187	14,3451	13,6356	12,9717	12,3398
14	36,1239	33,4262	31,3194	29,1412	26,1189	23,6848	21,0641	19,4062	18,1508	17,1169	16,2221	15,4209	14,6853	13,9961	13,3393
15	37,6978	34,9494	32,8015	30,5780	27,4884	24,9958	22,3071	20,6030	19,3107	18,2451	17,3217	16,4940	15,7332	15,0197	14,3389
16	39,2518	36,4555	34,2671	31,9999	28,8453	26,2962	23,5418	21,7931	20,4651	19,3689	18,4179	17,5646	16,7795	16,0425	15,3385
17	40,7911	37,9462	35,7184	33,4087	30,1910	27,5871	24,7690	22,9770	21,6146	20,4887	19,5110	18,6330	17,8244	17,0646	16,3382
18	42,3119	39,4220	37,1564	34,8052	31,5264	28,8693	25,9894	24,1555	22,7595	21,6049	20,6014	19,6993	18,8679	18,0860	17,3379
19	43,8194	40,8847	38,5821	36,1908	32,8523	30,1435	27,2036	25,3289	23,9004	22,7178	21,6891	20,7638	19,9102	19,1069	18,3376
20	45,3142	42,3358	39,9969	37,5663	34,1696	31,4104	28,4120	26,4976	25,0375	23,8277	22,7745	21,8265	20,9514	20,1272	19,3374
21	46,7963	43,7749	41,4009	38,9322	35,4789	32,6706	29,6151	27,6620	26,1711	24,9348	23,8578	22,8876	21,9915	21,1470	20,3372
22	48,2676	45,2041	42,7957	40,2894	36,7807	33,9245	30,8133	28,8224	27,3015	26,0393	24,9390	23,9473	23,0307	22,1663	21,3370
23	49,7276	46,6231	44,1814	41,6383	38,0756	35,1725	32,0069	29,9792	28,4288	27,1413	26,0184	25,0055	24,0689	23,1852	22,3369
24	51,1790	48,0336	45,5584	42,9798	39,3641	36,4150	33,1962	31,1325	29,5533	28,2412	27,0960	26,0625	25,1064	24,2037	23,3367
25	52,6187	49,4351	46,9280	44,3140	40,6465	37,6525	34,3816	32,2825	30,6752	29,3388	28,1719	27,1183	26,1430	25,2218	24,3366
26	54,0511	50,8291	48,2898	45,6416	41,9231	38,8851	35,5632	33,4295	31,7946	30,4346	29,2463	28,1730	27,1789	26,2395	25,3365
27	55,4751	52,2152	49,6450	46,9628	43,1945	40,1133	36,7412	34,5736	32,9117	31,5284	30,3193	29,2266	28,2141	27,2569	26,3363
28	56,8918	53,5939	50,9936	48,2782	44,4608	41,3372	37,9159	35,7150	34,0266	32,6205	31,3909	30,2791	29,2486	28,2740	27,3362
29	58,3006	54,9662	52,3355	49,5878	45,7223	42,5569	39,0875	36,8538	35,1394	33,7109	32,4612	31,3308	30,2825	29,2908	28,3361

8.7. Anexo 7. Ubicación de los dispositivos

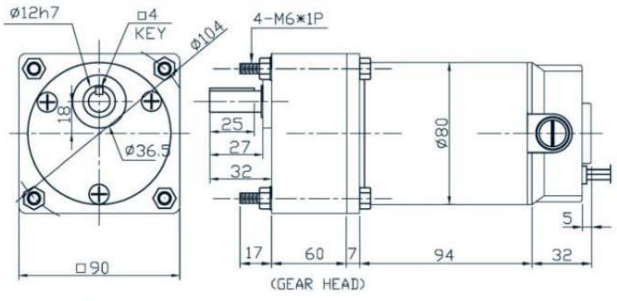


8.8. Anexo 8. Especificaciones técnicas King Motores sf7152

SPUR GEARED MOTOR — KING RIGHT MOTOR

Motor dia Φ80mm
Motor original torque : 26 - 36 Ncm
Gear ratio (1/ i) : 1/3 - 1/180

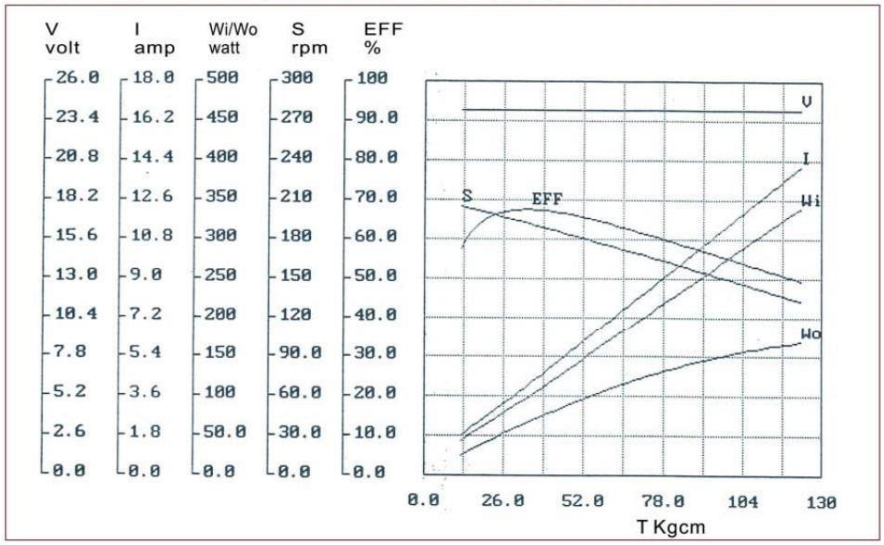
SF7152



RATIO	3	5	10	12.5	15	18	20	30	50	75	100	150	180

N.W 3.4 KGS/PCS
6PCS A CTN/N.W. 20.5KGS
G.W.21.5 KGS CFT.2.4

24VDC No Load 210rpm (RATIO：1/12.5)Torque/Speed Performance Curve

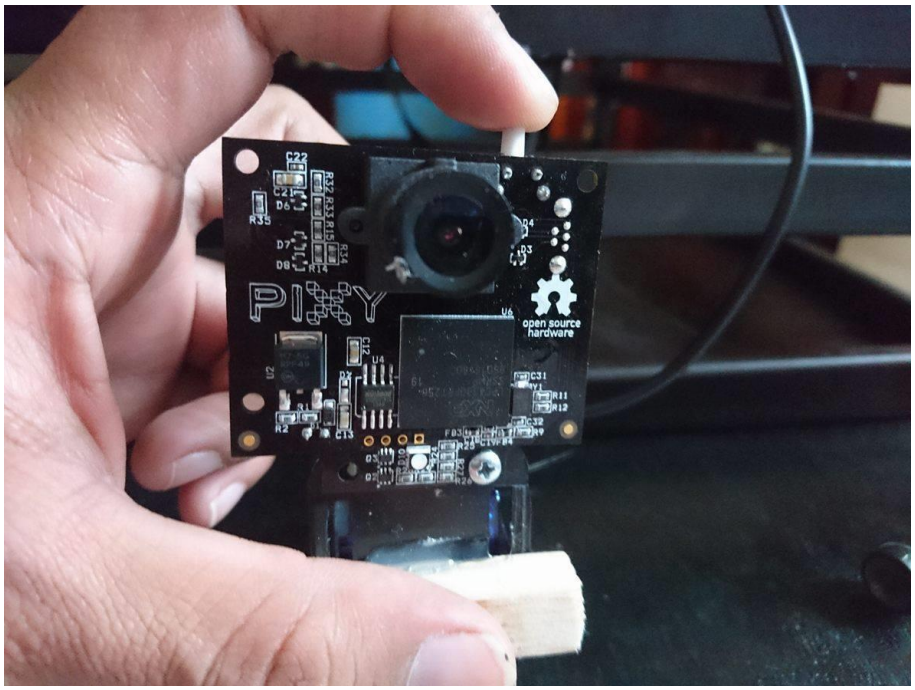


SPUR GEAR

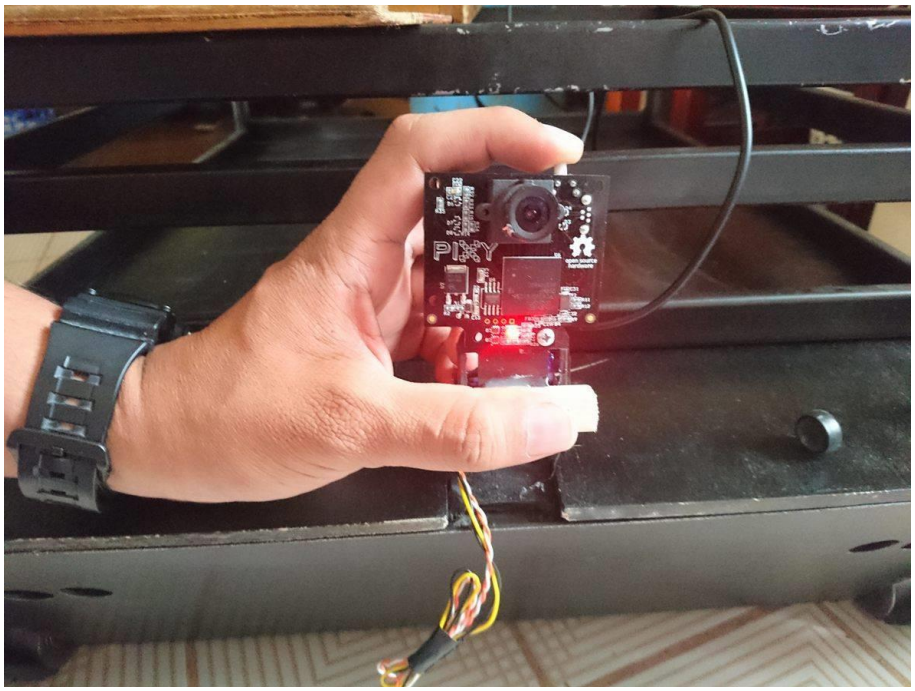
NO	R.P.M.	VOLT(V)	I(AMP)	INPUT(W)	TORQUE(KG-CM)	OUTPUT(W)	EFF(%)
1	206.1	24.11	1.76	42.38	12.00	25.40	59.93
2	202.7	24.11	2.21	53.37	16.00	33.30	62.39
3	198.8	24.10	2.89	69.63	22.00	44.90	64.48
4	196.4	24.14	3.43	82.71	27.00	54.40	65.77
5	191.1	24.13	4.15	100.15	35.00	68.60	68.50
6	186.4	24.12	5.02	121.12	42.00	80.30	66.30
7	180.7	24.11	6.05	145.99	51.00	94.60	64.80
8	174.2	24.12	7.24	174.71	63.00	112.60	64.45
9	166.8	24.10	8.70	209.67	76.00	130.10	62.05
10	155.2	24.09	10.51	253.11	90.00	143.30	56.62
11	143.3	24.06	12.27	295.30	106.00	155.80	52.76
12	132.0	24.06	14.26	343.08	125.00	169.30	49.35

8.9. Anexo 9. Captura de Imagen con Cámara Pixy

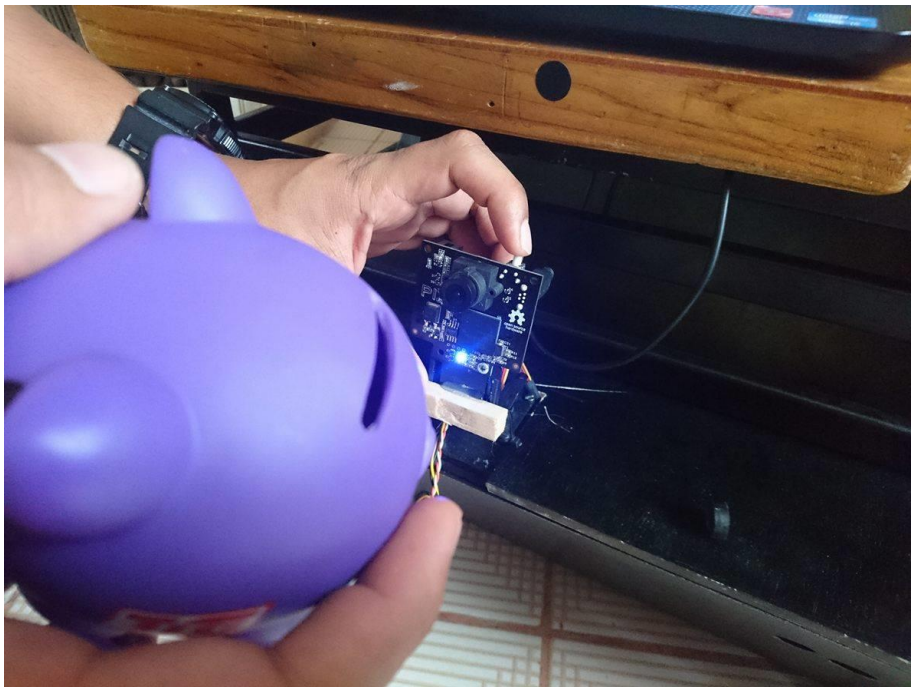
Para la captura del objeto con la cámara Pixy se puede realizar de dos maneras: La primera es por medio de aplastar el botón de la cámara hasta cuando el led de la cámara indique el primer color de captura que es el rojo, en este momento la cámara nos indica que esta lista para realizar la captación de la imagen.



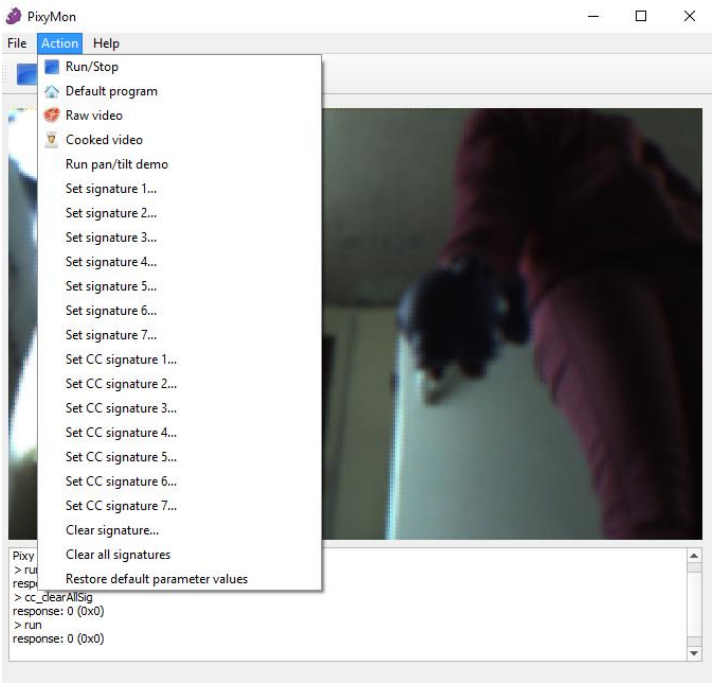
Se procede a colocar el objeto frente a la cámara, cuando el led cambia de color significa que el objeto ha sido fijado.



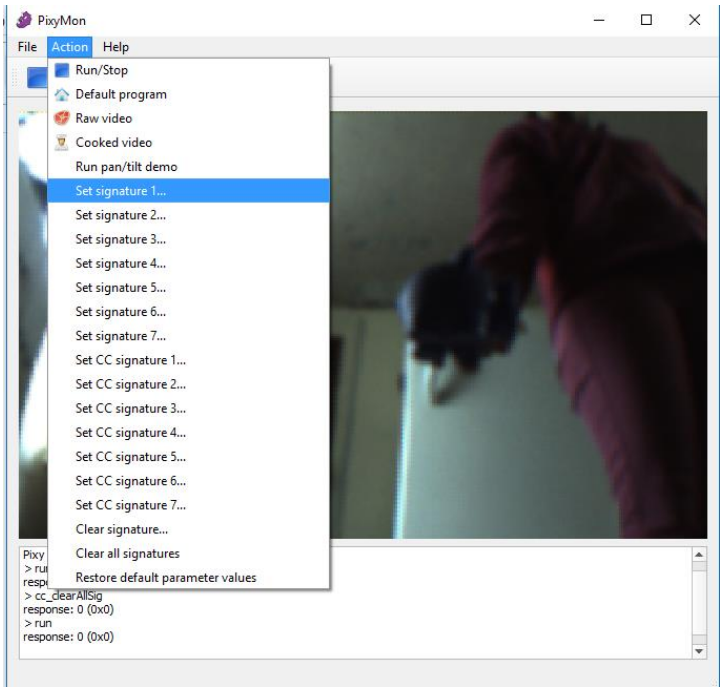
Luego presionamos el botón de la cámara y la imagen se encuentra capturada, el led cambia de color cuando un objeto se encuentra frente a la cámara.



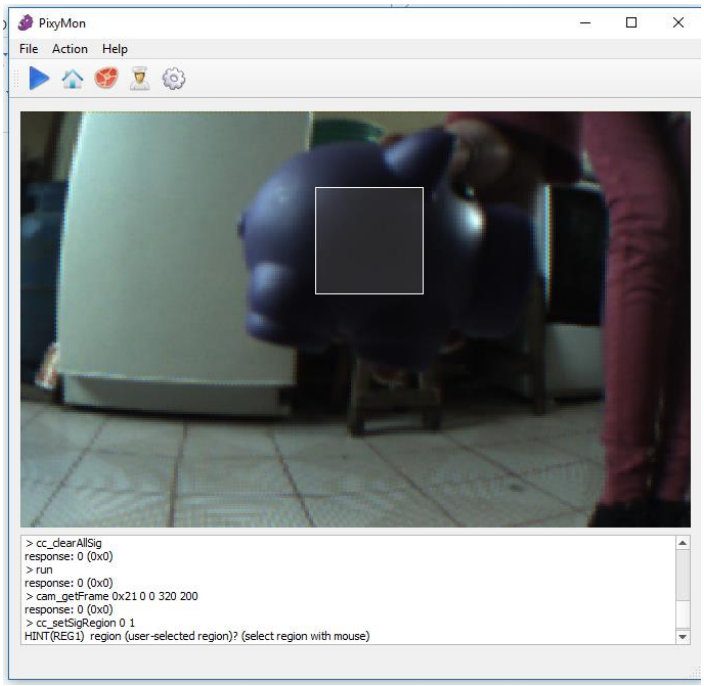
La segunda forma de realizar la captura de la imagen es por medio del software en el cual ingresamos a la opción Action



Escogemos la opción Set signature 1 y la cámara paraliza la imagen para realizar la captura.



Seleccionamos parte de la figura que vamos a seguir y tenemos la captura de la imagen



Y tenemos el objeto capturado