BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

Análisis y aplicación de un algoritmo de planificación de procesos en tiempo real en robots móviles.

TESIS PROFESIONAL
para obtener el título de
LICENCIATURA EN INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA
Raúl Cortés Gómez

Director de tesis:
M.C. Mariano Larios Gómez

Noviembre 2018

Dedicatoria:

A mis padres, que siempre han estado presentes en mis momentos de necesidad y angustia, brindando todo el apoyo y cariño que un hijo puede pedir, a ellos por enseñarme que incluso en los momentos más difíciles de la vida debes levantarte y seguir peleando, porque la vida es una pelea constante, por fungir como el faro y la luna que me guiaron en el camino de oscuridad, por ser las bases que me ayudaron a llegar hasta este momento, en el cual dejo una etapa muy importante de mi vida atrás alcanzando una meta más, pero abro las puertas de un futuro incierto y desconocido pero que con sus enseñanzas estoy seguro lograré afrontar con el rostro en alto.

El presente trabajo está dedicado a toda mi familia, a mis padres, a mi hermana, profesores y amigos que me inspiraron tanto en mi vida diaria como en la estudiantil a redactar este trabajo, ellos son quienes me dieron grandes enseñanzas y experiencias, ciertamente convirtiéndose así en los verdaderos protagonistas de este “sueño alcanzado”.

Agradecimientos:

Me van a faltar páginas para agradecer a las personas que se han involucrado en la realización de este trabajo, sin embargo merecen agradecimiento de todo corazón mis padres, familia y amigos, de igual manera mis agradecimientos a la Benemérita Universidad Autónoma de Puebla y a toda la Facultad de Ciencias de la Computación, a mis profesores, en especial al Doctor Mariano Larios Gómez por su paciencia dedicación, motivación, criterio y amistad, el cual es el principal colaborador durante todo este proceso, quien con su dirección, conocimiento, enseñanza y colaboración permitió el desarrollo de este trabajo, Ha hecho fácil esta travesía y un privilegio el trabajar a su lado.

Mi profundo agradecimiento a todas las autoridades y personal que conforman a la Facultad de Ciencias de la Computación.

También deseo aprovechar este momento para agradecer a Dios, por haberme dado la vida y permitirme el haber llegado hasta este momento tan importante de mi formación profesional.

Por último, agradezco al Laboratorio Nacional de Supercómputo del Sureste de México que pertenece a la red de laboratorios nacionales CONACyT, por los recursos computacionales, el apoyo y la asistencia técnica.

"A Todos Muchas Gracias"

Índice general

Índice de Figuras

Indice de Tablas

Reporte de Solicitud de Patente.

Fecha: 24 Sep. 2018

V. 2

Tabla 1. Tabla de reporte de invención.

INFORMACIÓN GENERAL DEL INVENTO	
Título: LOBOBOTS	**Palabras clave (máximo 5): Robótica, Inteligencia Artificial, Sistemas distribuidos, innovación tecnológica**
Inventor(es), datos de contacto y porcentajes de participación: **Raúl Cortés Gómez** **Dirección: Avenida Llanos N°30, Fraccionamiento Real Campestre, Puebla, puebla** **C.P:72310** **Tel:2229099277** **Porcentaje de participación: 50%** **Mariano Larios Gómez** **Dirección: Calle Italia N°24, Fraccionamiento Bosques del Pilar, Puebla. Pue.** **C.P. 72310** **Tel. 2227301221** **Porcentaje de participación: 50%**	
¿Existirá Cotitularidad con otra institución (es) y/o empresa (s)? **Si** **No** [ ] [x] **En caso afirmativo:** Nombre de la institución (es) y/o empresa (s): Porcentaje de cotitularidad (participación): Datos de contacto:	

INFORMACIÓN TÉCNICA

a. Introducción (*máximo dos cuartillas***):**

Existen varios sitios en la naturaleza a los que se puede tener la necesidad de explorar o patrullar con la ayuda de un robot móvil, por ello, el trabajo este trabajo se enfoca en el diseño y desarrollo de un robot móvil como plataforma para la implementación de algoritmos novedosos que permitan la navegación de manera segura a través de un ambiente difícil de acceder por humanos, de tal manera que este evite colisiones contra los objetos. Por tal razón, algunos de nuestros proyectos de robótica móvil requieren nueva tecnología y algoritmia en tarjetas de desarrollo, con una potencia de cálculo mayor, como por ejemplo el de un micro-ordenador portátil, para este tipo de proyectos se opta por tarjetas de desarrollo tales como las de la familia Atmega en especial ARDUINO NANO por la tecnología y su potencia de cómputo y compatibilidad con varios sistemas operativos como Linux, Windows, MacOs, Solaris, entre otros., de distintas librerías OpenSource para el procesamiento del entorno sobre el cual interactúa, además de múltiples dispositivos compatibles con los puertos IO de la tarjeta [9, 10].

En este proyecto de investigación y desarrollo utilizamos las librerías de nativas en el lenguaje C y algunas rutinas a bajo nivel como ensamblador para muestreo de caminos y de localización de obstáculos, en especial los que se encuentran en el medio ambiente o en la naturaleza o pudiera encontrarlos en algún torneo de robótica. Este prototipo también se puede utilizar para las prácticas de hardware y software como un sistema embebido en un ámbito educativo.

b. Descripción Técnica (*Metodología, resultados, conclusiones y bibliografía*):
Incluir gráficas y figuras en un archivo aparte. No hay límite de cuartillas.

Metodología.

En primera instancia se usan algoritmos de planificación de trayectorias basados en la búsqueda y obtención de estrategias de control seguras, para obtener la trayectoria adecuada con un robot móvil y que posea la mayor calidad en desplazamiento. Así como el modelado, diseño y programación en un móvil clásico no tripulado miniatura.

La planificación de trayectorias tiene como fundamento la planificación de movimientos y el seguimiento de rutas. Por tal motivo se contemplan algoritmos como Roadmaps probabilístico, otras variaciones con Lyaponov o Voronoi. Así como, los básicos algoritmos de planificación RRT (Rapidly-Exploring Random Trees) y RRT-Connect propuestos por LaValle [1], sin dejar pasar algoritmos de planificación como las propuestas por Dubins [2] que consisten en planificar exactamente con 3 segmentos de rutas, creando árboles con los métodos LSL, LSR, RSR y RSL aplicados a un robot móvil.

Se modeló y programó un vehículo no identificado, por ejemplo, un robot móvil clásico miniatura de tres ruedas, con un controlador PID (proportional integral derivative) [4]. Esta configuración tiene características importantes que otras configuraciones no tiene, como por ejemplo el móvil de 4 ruedas, móvil con orugas, entre otros [3]. El móvil clásico de tres ruedas tiene mejor velocidad con respecto a la velocidad angular en su eje de yaw, es decir en dar vuelta a la izquierda o derecha, No tiene funciones complejas porque tiene sólo 2 motores. Existen configuraciones de estos móviles que dependen solamente de un motor y una rueda loca.

Resultados.

En este punto se aplicó el uso de los siguientes módulos:

- **Infrarrojo**
- **Ultrasónico**
- **Motor driver**
- **Arduino nano**

Estos trabajando en conjunto con otros materiales permitieron el correcto funcionamiento de nuestro producto final, para eso proponemos la aplicación de propios algoritmos con su respectiva implementación en código C y ensamblador que activa todos los sensores y los hace trabajar detectando objetos a una distancia mayor a 20 cm. Además de detectar el espectro sin reflexión de luz a reflexión de luz realizando una toma de decisiones dependiendo de dichas entradas y salidas de los valores adquiridos por estos módulos.

Durante la etapa de desarrollo se obtuvieron 5 versiones del prototipo las cuales fueron mejorando a su siguiente revisión.

En la tabla 1, se enlistan los materiales requeridos y su costo aproximado por mayoreo, en relación a producto y cantidad, cabe mencionar que dichos materiales deben conseguirse con un proveedor el cual maneje costes de producción directos de fábrica, esto para reducir costos y obtener ganancias para la rentabilidad del proyecto.

PRODUCTO	PRECIO UNIDAD	CANTIDAD	PRECIO FINAL
Motor reductor	22.5	40	900
puente h	16.52	20	330.4
infrarrojo	9.35	20	187
ultrasónico	16.4	20	328
rueda loca	4.35	20	87
Arduino nano	42.05	20	841
AA Case Baterías	9.37	20	187.4

Pegamento	2.4	10	24
caucho	1	30	30
tornillos	100	1	100
ABS	450	1	450
DuPont h/h	0.68625	80	54.9
Switch	2.923076923	13	38
Impresora	10,500	1	10,500
Inversión Total			14,057.7

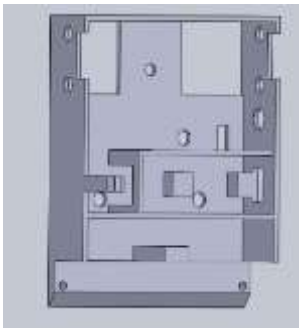
Tabla 1. Costos

Cabe mencionar que si los costos de envió son elevados sin lugar a duda nuestro pedido llegará rápido y con una protección de pedido no solo por parte del proveedor sino también por parte de la agencia de envíos, pero como es fácil apreciar, estos beneficios extra solo son funcionales si el pedido es muy grande, ya que los costes internacionales de envió ascienden arriba de $1,000.00MXN lo cual no sería redituable a menos que en relación a la ganancia y producción las unidades absorban dicho costo extra con pedidos masivos, o que solo se maneje un proveedor lo cual no siempre es la mejor opción.

Prototipo Final

Es la versión actual del prototipo, como se muestra en la figura 1 a) y b), se ilustra la versión donde se culminó la parte de diseño y fabricación siendo este el modelo usado para la promoción del producto presentando sus principales características. Estas se enlistan a continuación:

- **Reducción de paredes en el chasis sin comprometer la calidad del producto**
- **Solución de falla estructural en el posicionamiento del Motor Reductor**
- **Inclusión de ranura interna/externa para Arduino nano Mini USB**
- **Inclusión de soportes extra para Slot de Arduino Nano**
- **Inclusión de perforaciones en la pala del Robot para usos varios (postes de equilibrio)**
- **Inclusión de Ranura para botón Switch.**



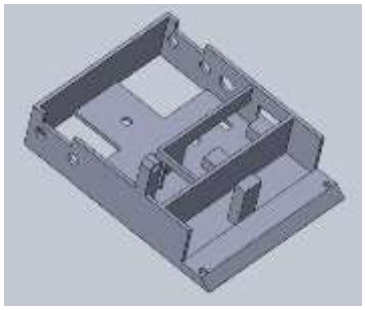
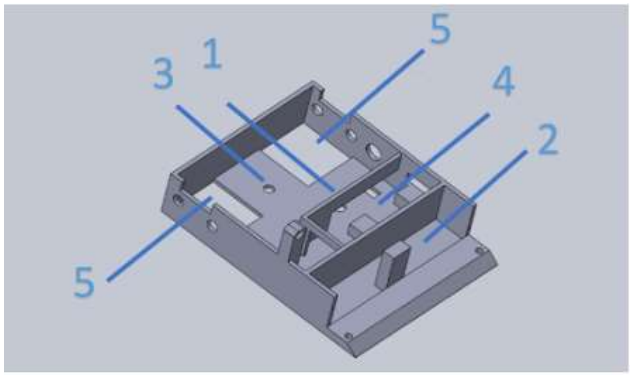


Figura 1. a) Vista superior Prototipo. **b) Vista Lateral Prototipo.**

El producto terminado se muestra en la figura 3 junto con la disposición de todos los módulos, en ellas podemos denotar varias implementaciones del mismo aprovechando su diseño para la distribución de sus componentes internos, en la figura 4 se muestra aplicando una de tantas posibles opciones para su fuente de alimentación, la cual puede ser adaptada según las necesidades del usuario.

1. **Infrarrojo**
2. **Ultrasónico**
3. **Motor driver**
4. **Arduino nano**
5. **Motor-reductores**



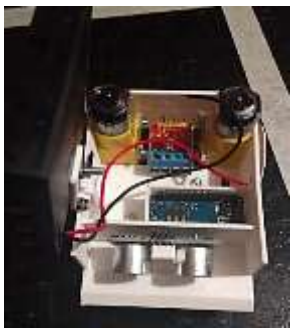


Figura 3. Móvil Vista Superior.

En la figura 4, se muestran 2 vistas distintas de los robots prototipos en funcionamiento. Estos pueden competir para evitar obstáculos en ambientes difíciles de acceder.

La aportación importante de este prototipo fue el proponer un prototipo con fundamentos en diseño y programación de un robot móvil en 3D capaz en usarlo para la educación, competencias de sumobots o en rescates de personas en catástrofes naturales en exploración de lugares difícil de acceder por humanos. Todas las partes fueron realizadas con una impresora en 3D usando una impresora con las características necesarios para poder obtener cada pieza del prototipo, basándonos en las técnicas de impresión y desarrollo en 3D [5, 6, 7, 8].

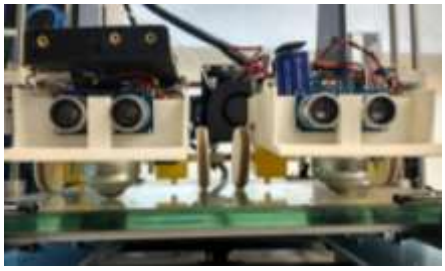


Figura 4. Móvil Vista Frontal.

Conclusiones.

El área de desarrollo de nuevas tecnologías está creciendo de manera exponencial así como, la aparición de nuevos campos de estudio y modificación de modelos educativos los cuales sustituyen a los departamentos de cómputo por una gestión que implique no solo el desarrollo y administración de sistemas de cómputo si no, el desarrollo de hardware y tecnologías capaces de facilitar nuestras labores diarias, es decir, de nuevas tecnologías propias de una sociedad que ha alcanzado una comunión con la ciencia y tecnología.

Por tanto, se decidió enfocarnos en proponer y desarrollar una cultura en el desarrollo de un prototipo basados en tecnologías programables y el aprendizaje tanto de lenguajes de programación como electrónica básica en alumnos de educación media y media superior, preparándolos así para insertarse a

un entorno más profesional, con una dificultad y competitividad mayores, y que estos sean capaces de superar las adversidades que este nuevo entorno estudiantil les presenté.

Bibliografía.

1. **Jaillet L., Siméon T. (2008) Path Deformation Roadmaps. In: Akella S., Amato N.M., Huang W.H., Mishra B. (eds) Algorithmic Foundation of Robotics VII. Springer Tracts in Advanced Robotics, vol 47. Springer, Berlin, Heidelberg**
2. **Lugo-Cárdenas, I., Flores, G., Salazar, S., & Lozano, R. (2014, May). Dubins path generation for a fixed wing UAV. In Unmanned Aircraft Systems (ICUAS), 2014 International Conference on (pp. 339-346). IEEE.**
3. **NVIDIA Inc. Embedded Systems. Jetson TX1 developer kit Full-featured development platform for visual computing. http://www.nvidia.com/object/jetson-tx1-dev-it.html#sthash.xM9kPgjh.dpuf**
4. **henrypadilla1997 (2015). Manipuladores son sistemas mecánicos. Universidad Nacional Autónoma de México. Recuperado de** https://www.coursehero.com/file/p37cltl/Manipuladores-Son-sistemas-mecánicos-multifuncionales-con-un-sencillo-sistema/
5. **TRSD. Soportes e impresión 3D: lo que nunca te cuentan Recuperado de https://impresiontresde.com/soportes-e-impresion-3d-lo-que-nunca-te-cuentan/**
6. **Autor Anonimo (2012). CURA – Software para Impresión 3D – Sprinter Vs Marlin. Recuperado de** https://www.tr3sdland.com/2012/12/software-cura-para-impresion-3d-sprinter-vs-marlin/
7. **Knowledge(2018). Orientación de los modelos. Recuperado de** https://support.formlabs.com/s/article/Model-Orientation?language=es
8. **Impresoras3D.com (2017). Tipos de impresoras 3D. Recuperado de** https://www.impresoras3d.com/tipos-de-impresoras-3d/
9. **Luis Llamas (2018). Ingeniería, informática y diseño. Recuperado de** https://www.luisllamas.es/
10. **Authors, avrchip.com (2015). AVR Microcontroller and Arduino Tutorial. Recuperado de http://avrchip.com/arduino-nano-datasheet-and-tutorial/**

TIPO DE DESARROLLO		
( ) Proceso o metodología	(x) Equipo o dispositivo	( ) Otro (especificar)

MADUREZ DEL DESARROLLO TECNOLÓGICO		
Indique el grado de madurez actual del desarrollo o innovación		
( ) Metodología	( ) Diseño	(x) Prototipo de laboratorio
( ) Diseño industrial o para escalamiento	( ) Modelo escala real	( ) Otro _____

DIVULGACIÓN				
a. Indique si algún aspecto de la invención ha sido publicado en alguno(s) de los siguientes rubros, si así fuese anexe documento comprobatorio:				
( ) Publicado en artículo científico	( ) Exposición en Congreso	( ) Tesis	( ) Otro	(x) Ninguna

b. ¿Existe algún plan para publicar la invención en el futuro?	
(x) Sí *(Especificar)* *La invención forma parte de un conjunto de técnicas, desarrollo de piezas y software los cuales forman parte de un trabajo de tesis para conseguir la Ing. En ciencias de la computación.* *Presentación en un congreso de innovación y talento*	( ) No

1 Introducción

1.1 Resumen.

Existen varios entornos urbanizados o naturales en los que se puede tener la necesidad de explorar o patrullar con la ayuda de un robot móvil; en especial, lugares donde la presencia de obstáculos o imperfecciones de terreno sean comunes, tal como baches, terrenos irregulares, arboles, etc. Por ello, el trabajo se enfoca en el desarrollo y diseño de móviles los cuales sean capaces de procesar algoritmos novedosos que le permitan a un móvil navegar de manera segura a través de ambientes irregulares, de tal manera que éste evite colisiones contra objetos. Por tal razón, algunos de nuestros proyectos de robótica móvil requieren nuevas tecnologías en tarjetas de desarrollo, con una potencia de cálculo mayor, por ejemplo, el de un micro-ordenador portátil, para este tipo de proyectos se opta por tarjetas de la familia hardkernel [12,14], concretamente la ODROID XU4 o ARDUINO NANO por la tecnología y su potencia de cómputo y compatibilidad con varios sistemas operativos como Linux, Windows, entre otros. Así como, el uso de distintas librerías OpenSource para el procesamiento del entorno sobre el cual interactúa. Además, de múltiples dispositivos compatibles con los puertos IO de la tarjeta.

En este proyecto de investigación de tesis, utilizamos las librerías nativas en lenguaje C y algunas rutinas a bajo nivel para muestreo de caminos y de localización de obstáculos, en especial los que se encuentran en el medio

ambiente o en la naturaleza o pudiera encontrarlos en algún torneo de minirobótica.

1.2 Antecedentes del Proyecto.

Esta tesis surgió de un proyecto admitido en el laboratorio de supercómputo LNS-BUAP, con número 201801076C. Gracias a este proyecto, aceptado, se podrá realizar las pruebas del sistema propuesto, y obtener los resultados deseados.

En este proyecto se requirió el uso de software de reconocimiento de obstáculos en tiempo real y coordenadas en el espacio para el desarrollo de proyectos de robótica, por tanto, usamos tarjetas de desarrollo capaces de programar y modificar trayectorias del robot la cual nos permitiera integrar bibliotecas OpenSource para C/C++, además de elementos electrónicos compatibles con la misma, como tarjetas capturadoras de datos, actuadores y sensores. Esto debido a que las librerías nativas de C son usadas generalmente en el procesamiento de datos en entornos reales computarizados.

1.3 Objetivos.

En primera instancia se diseñó un móvil con herramientas de diseño en 3D y elementos electrónicos, tales como: sensores compatibles con una gran variedad de tarjetas de desarrollo, algunos de estos son el sensor ultrasónico y sensor infrarrojo los cuales en conjunto con algoritmos de planificación de trayectorias basados en la búsqueda y obtención de estrategias de control seguras. Se implementaron junto con todas las funciones de la tarjeta

ARDUINO con un sistema operativo bajo Linux que sea capaz de soportar el lenguaje C/C++ para el desarrollo y correcto funcionamiento del móvil no tripulado.

Objetivos específicos realizados.

- Se obtuvo la mayor calidad en su desplazamiento.
- Se modeló y programó un robot móvil clásico no tripulado miniatura.
- Se adaptaron librerías y programas necesarios para hacer una lectura y escritura de datos mediante el programa en C/C++, que fueron muy útiles para la manipulación de las tarjetas de desarrollo.
- Por último, se obtuvo la lectura de datos de forma óptima para recorrer los caminos previamente detectados.

1.4 Metodología.

En primera instancia se realiza el modelado, diseño y programación de un móvil clásico no tripulado miniatura, el cual es capaz de ejecutar algoritmos de planificación de trayectorias basados en la búsqueda y obtención de estrategias de control seguras, para obtener la trayectoria adecuada en móviles y que posea la mayor calidad en su desplazamiento.

La planificación de trayectorias tiene como fundamento la planificación de movimientos y el seguimiento de rutas. Por tal motivo se menciona que dicho móvil debe ser capaz de ejecutar algoritmos como Roadmaps probabilístico y otras variaciones con Lyaponov o Voronoi [9, 10, 11, 12]. Así como los básicos

algoritmos de planificación RRT (Rapidly-Exploring Random Trees) y RRT-Connect, propuestos por LaValle en [11]. Así como, algoritmos de planificación como las propuestas por Dubins [12, 13], que consisten en realizar la planificación de movimientos exactamente con 3 segmentos de rutas, creando arboles con los métodos LSL, LSR, RSR y RSL aplicados a un móvil.

Se modelo y programó un vehículo no identificado, por ejemplo, un móvil clásico miniatura de tres ruedas, con un controlador básico PID (proportional integral derivative), en este proyecto reducimos el controlador a solo un PD, asignándole al control integral I=0, esto debido al hecho de que cuando el móvil gira a más de 360° el integrador debe reiniciarse en I=0, en caso de ser mayor a 360° puede que el móvil genere efectos extraños como dar una vuelta innecesaria lo cual tiende a la inestabilidad. Esta configuración del robot móvil no tripulado, tiene características importantes que otras configuraciones no tiene, como por ejemplo el móvil de 4 ruedas, móvil con orugas, entre otros. El móvil clásico de tres ruedas tiene mejor velocidad con respecto a la velocidad angular en su eje de yaw, es decir, en dar vuelta a la izquierda o derecha, No tiene funciones complejas porque tiene sólo 2 motores. Existen configuraciones de estos móviles que dependen solamente de un motor y una rueda loca.

1.5 Cronograma de Actividades.

Los resultados reportados se entregaron de forma mensual, revisados por el asesor y se presentaron en un grupo de trabajo conformado por, profesores investigadores y alumnos, estos aportando ideas y ayudando a mejorar

continuamente este trabajo de investigación de tesis. La entrega de avances se muestra en la tabla 2:

Tabla 2. Entrega de resultados mensual.

Avances	Marzo	Abril	Mayo	Junio	Julio	Agosto	Sep	Octubre	Nov
Investigación y documentación del funcionamiento de robots móviles									
Modelado y desarrollo de un robot móvil y su planificación a utilizar									
Desarrollo de una aplicación para reconocimiento de un camino especifico									
Configuración y calibración de sensores									
Mapeo de trayectorias en zonas especificas									
Aplicación de una máquina de aprendizaje a un móvil									
Pruebas y resultados del									

sistema en un robot móvil									

1.6 Infraestructura.

1.6.1 Hardware necesario.

- Arduino Nano: Tarjeta de desarrollo compatible con lenguaje java, Processing o C Native.
- Fuente de poder: Fuente de 5 Volts, para alimentar el procesador de la tarjeta de desarrollo.
- Módulo Infrarrojo FC-51: sensor de detección de objetos por medio de infrarrojo.
- Sensor de ultrasonidos HC-SR04: Sensor ultrasónico capaz de detectar objetos a una distancia de más de 300 cm. de distancia.
- Motor driver L9110S: Módulo para el manejo de motores, mejor conocido como puente H.
- Módulo bluetooth: necesario para realizar conexiones del móvil a la PC.
- Mini USB de 5 Volts y 2Ampers.
- Moto reductores: Motores con engranaje reductor de 12 Volts a 300 RPM.

1.6.2 Software necesario

- SDFormatTool: herramienta para el formateo de unidades de disco.
- 7-zip: usada para extraer la imagen con extensión .xz.
- WinMD5: usado para verificar la integridad de la imagen del sistema operativo.
- Win32DiskImager: herramienta para la carga de un sistema operativo en una tarjeta (eMMC).
- Arduino IDE: software de compilación y programación basado en C++.
- Editor Java: NetBeans, un clásico editor de código en java.
- Librerías Java: Java Developement Kit que son librerías diseñadas para el uso de Lenguajes Orientados a Objetos.

Se consiguió un sistema para el reconocimiento de caminos de difícil y/o peligroso acceso a personas por medio de un robot móvil, así también el trabajar con varios métodos para obtener la ruta o camino en áreas peligrosas o zonas específicas con pequeños caminos.

Este trabajo se enfoca en aplicar algoritmos que le permitan a un robot móvil la navegación de manera segura a través de un ambiente boscoso, de tal manera que este evite colisiones contra los árboles sobre caminos o veredas. Este es un impacto socioeconómico, pues nos permite ayudar a la gente o a un ser vivo poder navegar en zonas pocas seguras. Por ejemplo, para poder ayudar a un animal en zonas poco accesibles a humanos o detectar elementos peligrosos que perjudiquen el bienestar de un ser vivo ya sea animales, plantas o humanos.

Las aportaciones principales consisten en el desarrollo de una plataforma o Framework que manipule y programé un robot móvil. Gracias a esta, se podrá aplicar otros algoritmos que se basan en diferentes áreas como la inteligencia artificial, cómputo distribuido, sistemas de tiempo real, planificación de movimientos de robot, entre otras. De esta forma se podrán realizar experimentos, en trabajos futuros. Otra aportación es la documentación completa del modelado cinemático y dinámico, programación y control de una arquitectura de un móvil tipo clásico miniatura que se aportará a la FCC-BUAP. Por último, se aportamos a la ciencia con la publicación de resultados en una revista científica nacional o internacional [21].

2 Estado del arte.

Los robots han tomado posición en todas las áreas productivas industriales y tecnológicas, incorporándose al proceso productivo y representado uno de los avances más espectaculares de la edad moderna. En poco más de cuarenta años, se ha pasado de aquellos primeros modelos, rudos y limitados, a sofisticadas máquinas capaces de sustituir al hombre en tareas repetitivas o peligrosas de forma más rápida, precisa y económica.

Ningún autor se pone de acuerdo en cuántos y cuáles son los tipos de robots y sus características esenciales por tanto sólo mencionaremos unos cuantos de sus exponentes.

El uso creciente de robots industriales en el proceso productivo, ha dado lugar al desarrollo de controladores industriales rápidos y potentes, basados en microprocesadores, que permiten establecer con exactitud la posición real de los elementos del robot y su desviación o error. Esta evolución ha dado origen a una serie de tipos de robots como los listados a continuación:

Manipuladores.

Son sistemas mecánicos multifuncionales, con un sencillo sistema de control, que permite gobernar el movimiento de sus elementos de forma manual (cuando el operario controla directamente la tarea del manipulador), de secuencia fija (cuando se repite, de forma invariable, el proceso de trabajo preparado previamente) y de secuencia variable.

Estos dispositivos son utilizados generalmente cuando las funciones de trabajo son sencillas y repetitivas [15].

Robots de repetición o aprendizaje.

Son manipuladores que se limitan a repetir una secuencia de movimientos, previamente ejecutada por un operador humano, haciendo uso de un controlador manual o un dispositivo auxiliar. En este tipo de robots, el operario durante la fase de enseñanza se vale de una pistola de programación con diversos pulsadores o teclas, o bien de joysticks, o bien utiliza un maniquí, o desplaza directamente la mano del robot. Actualmente, los robots de aprendizaje son los más conocidos en algunos sectores de la industria, y el tipo de programación que incorporan recibe el nombre de "gestual" [15].

Robots con control por computador.

Son manipuladores o sistemas mecánicos multifuncionales, controlados por un microordenador. El control dispone de un lenguaje específico de programación, compuesto por varias instrucciones adaptadas al hardware del robot, con las que se puede diseñar un programa utilizando solo el ordenador y sin la intervención del manipulador.

Las grandes ventajas que ofrece este tipo de robots, hacen que se vayan imponiendo en el mercado rápidamente, lo que exige la preparación urgente de personal capaz de desarrollar programas de control que permitan el manejo del robot [15].

Robots Inteligentes.

Similares a los anteriormente mencionados, pero tienen la capacidad de poder relacionarse con el mundo que les rodea a través de sensores y de tomar decisiones en función de la información obtenida en tiempo real [15].

Robótica Móvil.

Estos tienen un mayor grado de libertad y movilidad y su principal característica es el movimiento en el espacio físico, es decir, la posibilidad de desplazarse por el entorno para observar e interactuar con él, y de esta forma emular con mayor fidelidad las capacidades de los seres vivos.
Se puede establecer una taxonomía dentro del colectivo de los robots móviles. Si por ejemplo se atiende a sus características estructurales y funcionales se puede establecer la siguiente clasificación:

Robots rodantes.

Son aquellos que se desplazan haciendo uso de ruedas, generalmente montadas por pares en una configuración 2+2 como las de un vehículo por mera simplicidad. Habitualmente sólo dos de sus ruedas presentan tracción y otras dos direcciones, de forma que sea posible maniobrar el robot con un solo servomotor.

También es frecuente encontrar distribuciones de ruedas montadas en modo triciclo, donde una rueda sirve para la dirección y las otras dos aportan la tracción. Otra opción es que la tercera rueda simplemente sea una rueda 'loca' y las otras dos aporten tanto la tracción como la dirección, mediante el método de las orugas [15].

Robots voladores.

Existen dos aproximaciones, en función de su principio de vuelo y estructura:
Helicópteros: habitualmente helicópteros RC (Remote Control) convencionales a los que se les añade la electrónica necesaria para tener visión artificial y capacidad de toma de decisiones autónoma.
Drones o aviones no tripulados: actualmente en uso por el ejército de EEUU para tareas de logística en operaciones militares, apoyo cartográfico, así como tareas de espionaje [15].

3 Diseño y selección de componentes

El siguiente diseño está basado en robots móviles, empleados durante eventos locales y nacionales realizados en la facultad de ciencias de la computación de la Benemérita Universidad Autónoma de Puebla, como se muestra en la Figura 1, 2. Cabe mencionar que se optó por este diseño porque es manejable y usable a los competidores o participantes de dichos eventos. así también se agregaron características extras a los dos modelos propuestos y que participaron en 2 categorías distintas, de los cuales 1 se mantuvo invicto durante 2 años, como se muestra en la Figura 2.

Figura 1. Móviles usados en los eventos.

Figura 2. Premiación BUAP 2016 y 2017.

3.1 Bocetos, croquis y planos de diseño.

Dentro del proceso de diseño de un producto podemos emplear diferentes técnicas de dibujo para conseguir y definir nuestras ideas gráficamente para que puedan ser entendidas por los demás, lleguen a construirse y sean llevadas a un entorno real en 3D.

Derivado del término italiano bozzetto el concepto de boceto refiere al esquema o el proyecto que sirve de bosquejo para cualquier obra. Se trata de una guía que permite volcar y exhibir sobre un papel una idea general antes de arribar al trabajo que arrojará un resultado final.

Por lo general un boceto (definido como layout en idioma inglés) es una ilustración esquemática que carece de detalles y en la mayoría de los casos no posee terminaciones. Como se muestra en la figura 3 podemos ver la vista superior de nuestro primer diseño.

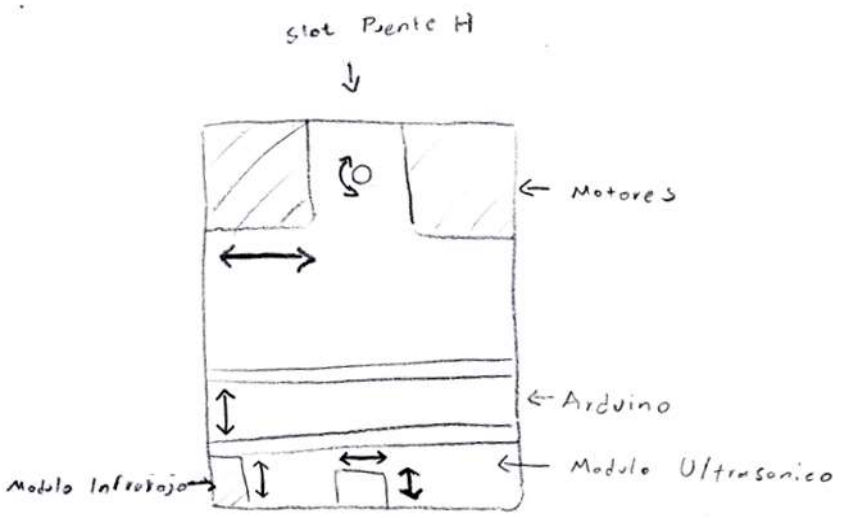


Figura 3. Vista superior del primer boceto.

Su objetivo es simbolizar ideas, pensamientos o conceptos, sin preocuparse por la estética. Por eso generalmente se realiza sobre cualquier clase de hoja y sin necesidad de disponer de instrumentos de dibujo auxiliares.

En este sentido podemos exponer que habitualmente cuando hablamos de boceto nos estamos refiriendo a un dibujo que se realiza a lápiz, papel, a mano alzada y sin entrar en ningún tipo de detalles, simplemente se trata de ideas básicas para acometer la posterior obra definitiva.

De esta manera nos encontramos documentos gráficos de este tipo en campos como, por ejemplo, arquitectura o moda. Y es que es frecuente que los diseñadores y los modistos se encargan de plasmar de una manera previa las ideas que les van surgiendo para luego acometer de una forma mucho más exhaustiva y compleja el conjunto de sus confecciones que conformarán los modelos de sus nuevas colecciones.

Igualmente podemos subrayar que en el mundo del cómic también es frecuente que los dibujantes una vez reciben los guiones de las historias se encarguen de llevar a cabo estos bocetos para establecer la mejor manera de representar lo que son los episodios que se cuentan en las tramas.

Los bocetos pueden ser considerados como un estudio previo de otra clase de trabajos. Por ejemplo: un dibujo puede constituir el primer paso de una obra de arquitectura o de una escultura.

Existen tres grandes tipos de bocetos: el burdo, el comprensivo y el dummy. El primer boceto sencillo supone la representación en papel de una primera idea, desprovista de detalles y contenidos técnicos como se muestra en la figura 3, figura 4 y figura 5.

El boceto comprensivo incluye ciertos ajustes a dicha idea a fin de mejorar su calidad y hacerla más comprensible, véase la figura 7. Para eso se emplean diversas herramientas técnicas. Por último, el boceto dummy se caracteriza por tener un elevado grado de precisión y calidad en todos los efectos visuales que se aprovecharán para la obra final.

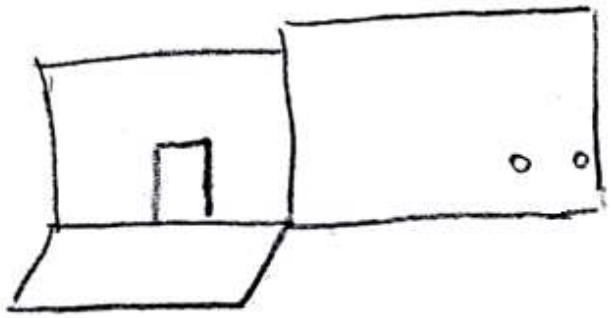


Figura 4. Boceto vista lateral.

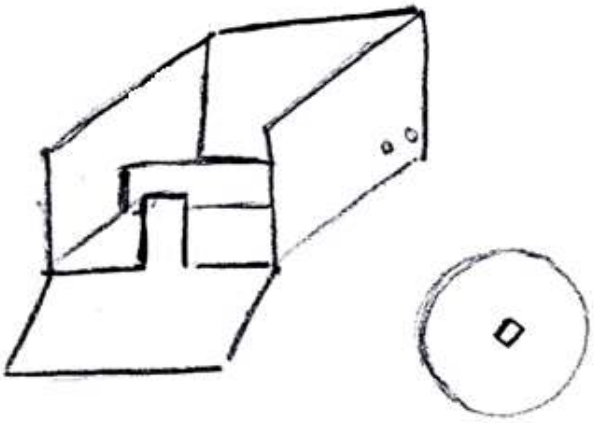


Figura 5. Vista en diagonal.

Un croquis es una representación de un lugar, de tal manera que a través de una serie de trazos se dibuja un espacio concreto. Hay que indicar que el croquis viene a ser una versión simplificada de un plano (figura 6), por lo que no se trata de dibujar todos los detalles de un espacio sino de ofrecer una imagen aproximada.

Este tipo de representación gráfica sirve para ciertas actividades cotidianas. No es lo mismo un croquis que un plano. Si bien ambos términos tienen una evidente semejanza, es conveniente no confundirlos. El croquis es mucho más simple y en el mismo no es relevante la escala ni la exactitud de la información (figura 7, figura 8). El croquis puede ser un primer paso para realizar posteriormente un plano. De esta manera, se podría decir que el croquis es un boceto inicial de un posible plano como se muestra en la figura 9. En esta figura se muestra un diseño más detallado de dicho concepto.

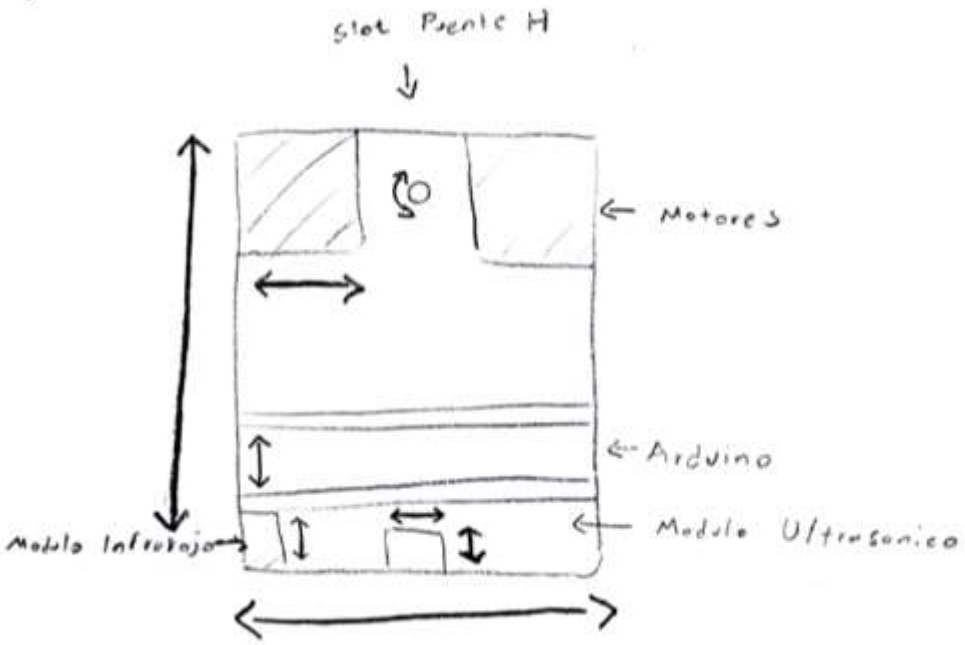


Figura 6. Croquis vista superior.

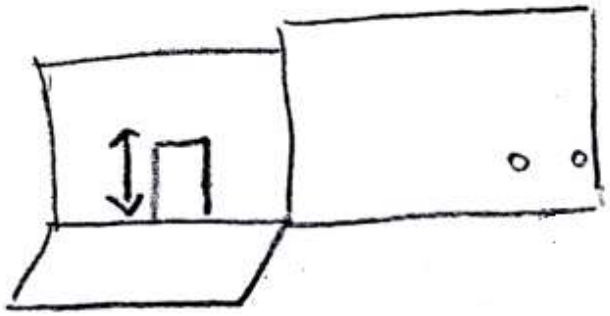


Figura 7. Croquis vista lateral.

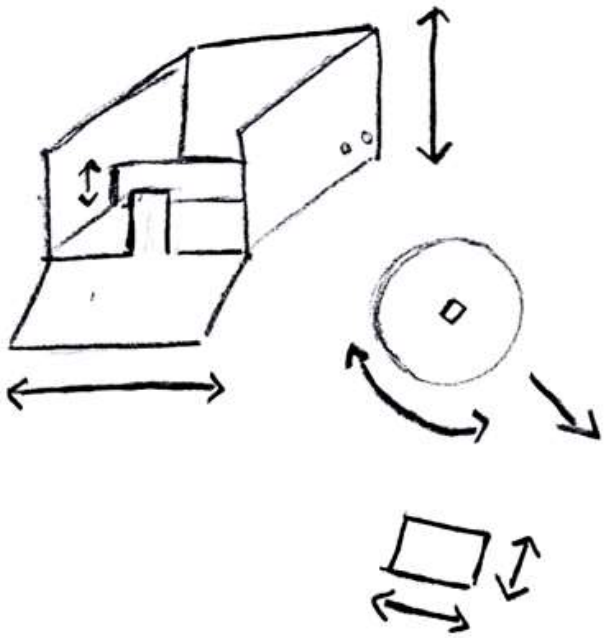


Figura 8. Croquis vista en diagonal.

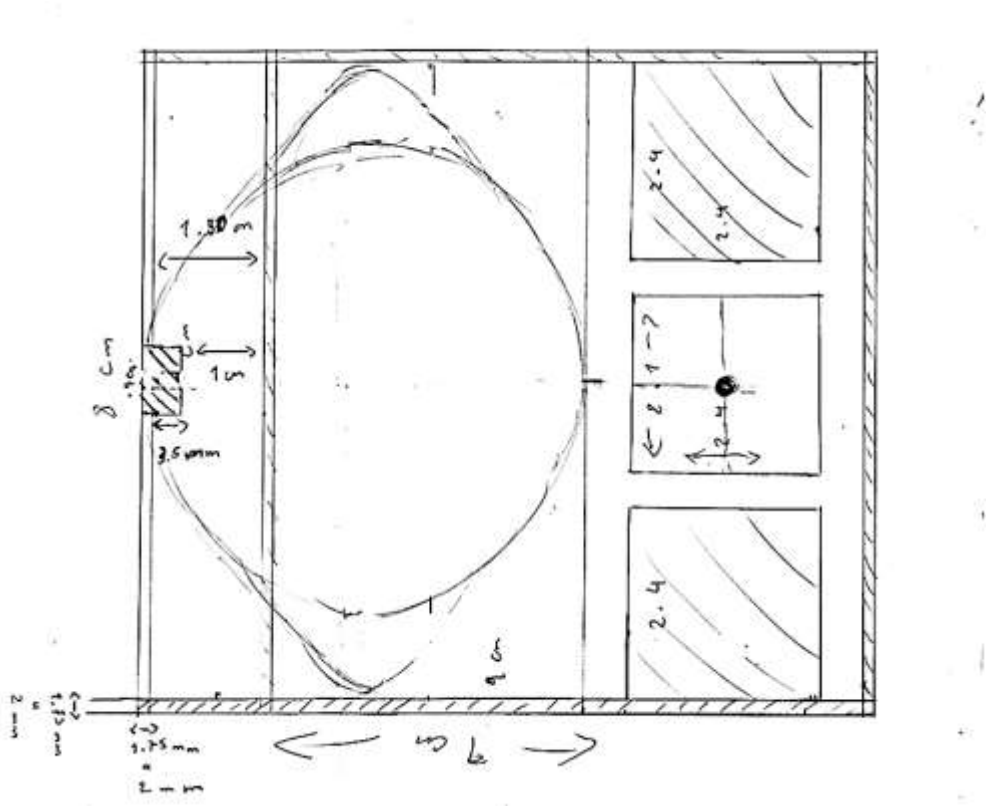


Figura 9. Vista superior croquis.

Un plano, por otra parte, es una representación esquemática y a una cierta escala de una construcción, un terreno, una población, una máquina u otra cosa, en este sentido la figura 11 se puede apreciar un plano del diseño y se puede denotar que es un elemento que sólo cuenta con dos dimensiones y que alberga varios puntos y rectas, aunque este puede contar con estos elementos de forma infinita. Los planos son dibujos delineados, se realizan con ayuda de instrumentos de dibujo de alta precisión (escuadra, cartabón, regla, compás, etc.), para conseguir una representación de un dibujo cuyas medidas están en proporción con el objeto en la realidad (Escala) figura 10.

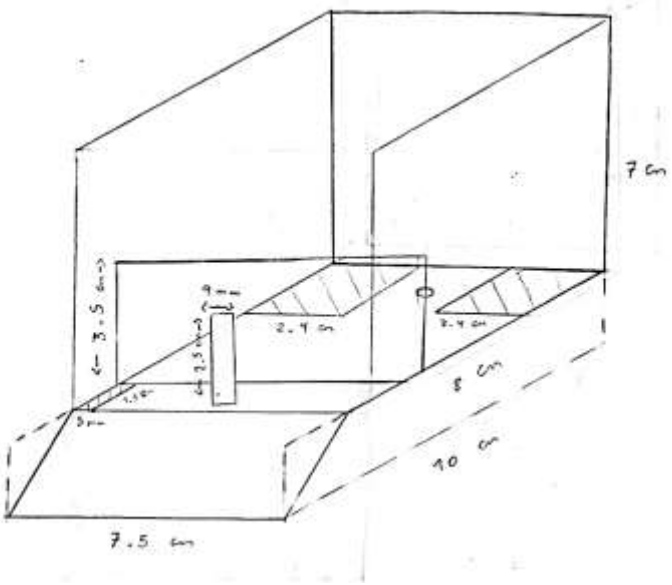


Figura 10. Plano en diagonal.

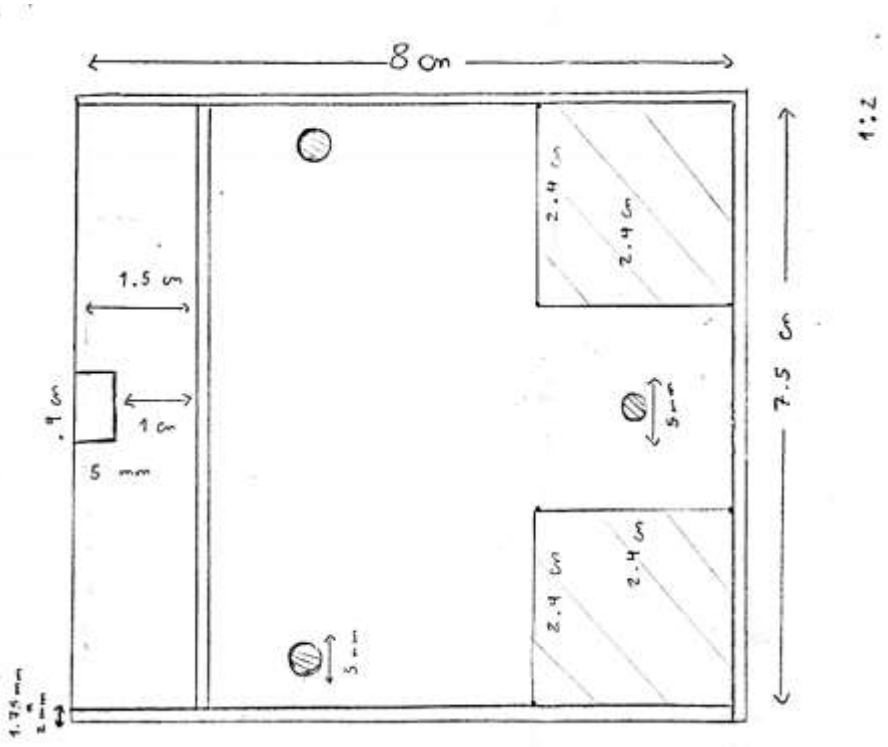


Figura 11. Plano vista superior.

3.2 Modelo 3D (CAD).

El diseño asistido por computadoras, más conocido por sus siglas inglesas CAD (computer-aided design), es aquel en el que se utilizan diferentes programadores gráficos para lograr crear una serie de imágenes que conjuntas crean una imagen de mayor amplitud o más conocida como dibujo (figura 13). El CAD es también utilizado como un medio de expresión mediante un ordenador y un gestor gráfico; a su vez, se puede decir que también es considerado como relativamente una nueva técnica de dibujo revolucionaria, con la cual se pueden realizar dibujos y planos, también se puede llegar a encontrar denotado con las siglas CADD (computer-aided design and drafting), que significan "bosquejo y diseño asistido por computadora".

Estas herramientas se pueden dividir básicamente en programas de dibujo 2D y de modelado 3D. Las herramientas de dibujo en 2D se basan en entidades geométricas vectoriales como puntos, líneas, arcos y polígonos, con las que se puede operar a través de una interfaz gráfica. Los modeladores en 3D añaden superficies y sólidos.

El CAD fue principalmente inventado por un francés, Pierre Bézier, ingeniero de los Arts et Métiers ParisTech. El ingeniero desarrolló los principios fundamentales del CAD con su programa UNISURF en 1966.

El usuario puede asociar a cada entidad una serie de propiedades como color, capa, estilo de línea, nombre, definición geométrica, material, etc., que permiten manejar la información de forma lógica. Además se pueden renderizar a través de diferentes motores o softwares como V-Ray, Maxwell Render, Lumion, Flamingo, entre los que son pagados, hay algunos de licencia free and

OpenSource, como por ejemplo el Kerkythea y Acis, entre los más usados, son modeladores 3D para obtener una previsualización realista del producto, aunque a menudo se prefiere exportar los modelos a programas especializados en visualización y animación, como Autodesk Maya, Autodesk Inventor, Solidworks, Bentley MicroStation, Softimage XSI o Cinema 4D y la alternativa libre y gratuita Blender, capaz de modelar, animar y realizar videojuegos.

Para el diseño e impresión de piezas se optó por un software ampliamente conocido el cual se basa en el diseño CAD, hablamos de SolidWorks, aunque existen otras opciones de pago como freeware las cuales son capaces de manipular y crear archivos STL (Standard Triangle Language) que es un formato de archivo informático de diseño asistido por computadora (CAD), como se muestra en la figura 12, que define geometría de objetos 3D excluyendo información como color, texturas o propiedades físicas que sí incluyen otros formatos CAD figura 14.

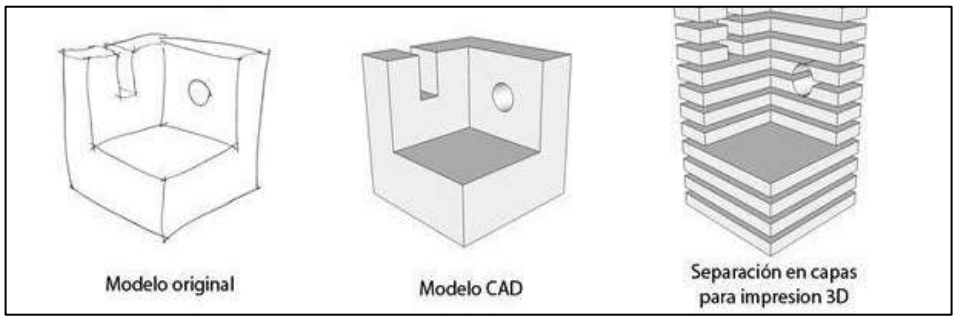


Figura 12. Diseños e impresión.

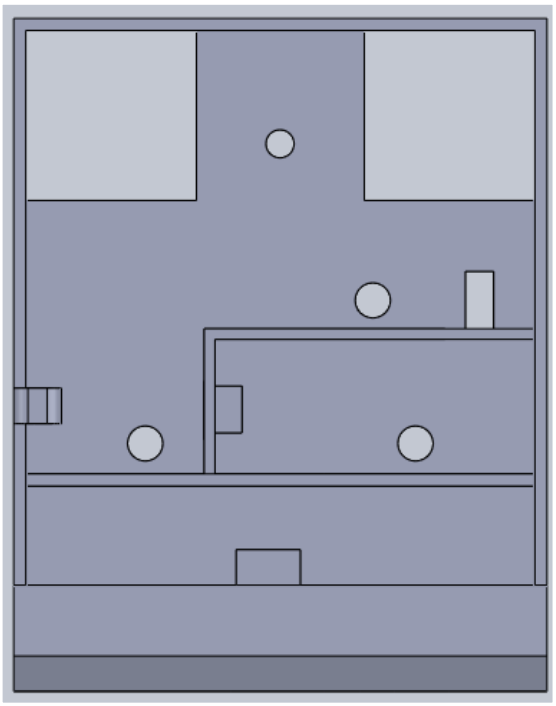


Figura 13. Plano vista superior.

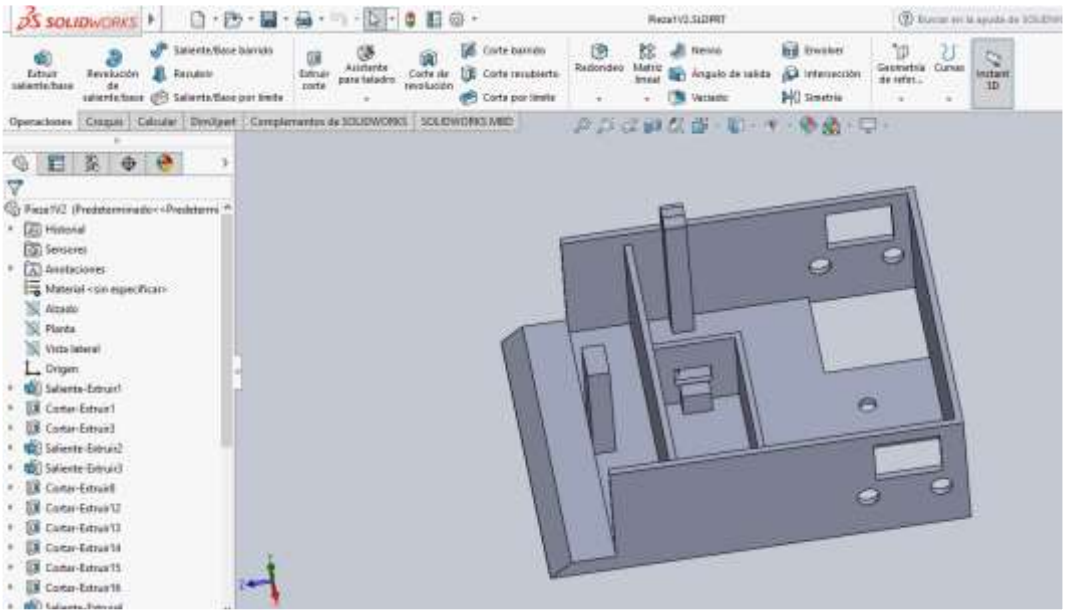


Figura 14. Software CAD.

3.3 Herramienta para impresión del prototipo CURA.

3.3.1 Software de impresión.

Cada software de diseño CAD utiliza el formato que su fabricante crea más conveniente al momento de realizar diseños, pero una vez tenemos el diseño terminado habrá que exportarlo a un archivo común que utilizan casi todas las impresoras 3D, el formato STL que se encarga de definir la geometría del objeto en 3D excluyendo colores, texturas y demás información que no es útil a la hora de imprimir, este tipo de archivos contiene una cabecera con información, velocidad, temperatura de cabezal y máquina, que la impresora 3D extrae para producir la pieza.

Toda esta parametrización se hace con software especializado en este sector, el más famoso o por lo menos el más usado es el Cura de Ultimaker.

3.3.2 CURA.

Cura es un software que nos va a permitir convertir los archivos STL que contienen nuestro diseño 3D en piezas físicas en un solo entorno de trabajo donde podremos no solo configurar parámetros para una impresión de baja o alta calidad, si no también, nos permite pre-visualizar las piezas y su manipulación por medio de herramientas basadas en la geometría básica.

Originalmente, Cura se llamaba SkeinPyPy. Este nombre proviene de la combinación de Skeinforge con PyPy. Skeinforge es un software que realiza el corte de las piezas 3D en capas (slicer) generando el código para que nuestras

impresoras lo comprendan. Sin embargo, Skeinforge es bastante lento y le puede llevar más de una hora “cortar” modelos complejos.

Skeinforge está hecho en Python, así que lo combinaron con el complemento PyPy el cual lo hacía correr más rápido, la combinación de los dos dio lugar a “SkeinforgePyPy” que más tarde se llamó “SkeinPyPy”.

SkeinPyPy no era más que un paquete que combinaba Skeinforge con PyPy y algunos parches para hacerlo más fácil de usar. El paquete venía preparado para ejecutarlo directamente, sin instalar ningún otro complemento, incluyendo PrintRun para la comunicación con nuestra impresora, ya estábamos en las versiones ALPHA, pero todavía era Skeinforge y el interfaz de usuario de Skeinforge, con 260 opciones de configuración, es un poco difícil de usar, muchas de las opciones no se aplican a la impresión 3D y los cambios de configuración no son fáciles. Así, que se inició el proyecto NewUI. Con el aporte de muchas personas, se creó un interfaz fácil y sencilla de usar. Aún sigue utilizando Skeinforge como backend, pero sin mostrar ninguna de las horribles pantallas de usuario de Skeinforge. En este punto, SkeinPyPy comenzó a evolucionar para hacer Skeinforge más rápido y conseguir un todo en uno, fácil de usar en un solo paquete. Estos paquetes son las versiones Beta del proyecto. En este punto, se decidió cambiar el nombre del paquete de SkeinPyPy por CURA, debido a que superó obejtivo inicial de “Skeinforge con PyPy”, el nombre de CURA se eligió dentro de la comunidad Ultimaker [17].

Finalmente, en octubre de 2012, Ultimaker contrató al creedor de este software para evolucionarlo y hacer otros proyectos relacionados con el software de impresión en 3D. Su autor nos garantiza que CURA seguirá siendo gratuito, de código abierto y que apoyará a las máquinas RepRap, así como a Ultimaker.

Algunas de las funciones más relevantes que utilizamos en el diseño de piezas son:

Orientación en los ejes XYZ.

Podemos cambiar la orientación de los modelos en los ejes X, Y y Z de distintas maneras; en las sugerencias en las siguientes secciones encontraremos recomendaciones sobre la mejor forma de orientar los modelos. La manera más fácil de rotar una pieza es haciendo clic sobre sobre el control esférico que se superpone al modelo y arrastrarlo. Rota los modelos con mayor precisión en uno de los tres ejes principales arrastrando uno de los anillos circulares del control o introduciendo ángulos específicos en las casillas de la barra de herramientas llamada “Orientación”. La rotación del eje Z se puede ajustar en cualquier momento desde la barra de herramientas “Orientación” o utilizando el anillo de rotación del eje Z sin afectar a los soportes. No obstante, si se ajustan los ángulos en los ejes X y/o Y, se deberán volver a generar los soportes, como se ilustra en la figura 15.

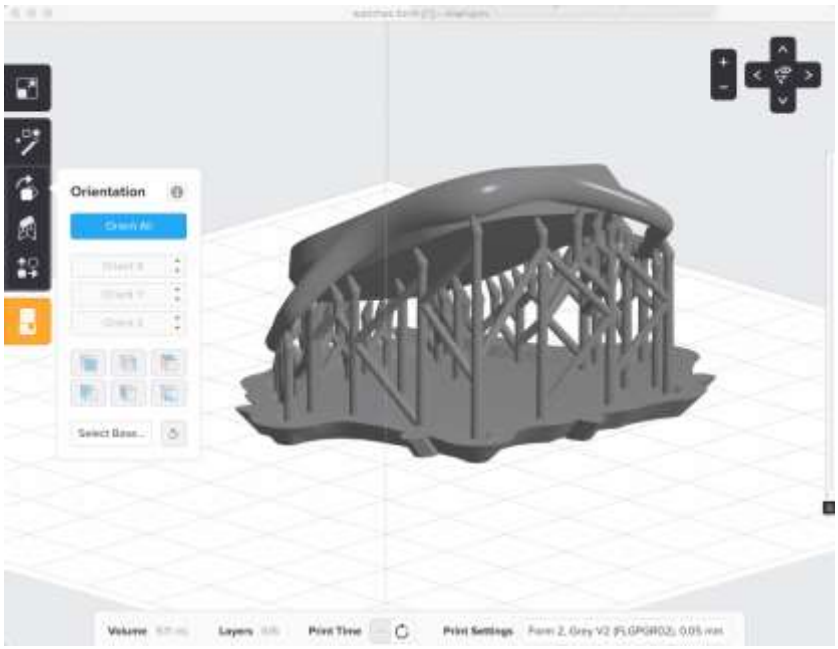


Figura 15. Orientación del modelo.

Los ángulos de rotación X, Y y Z se restablecen a cero grados después de realizar una rotación. Esto se debe a que las herramientas de orientación del PreForm son relativas, no absolutas. Es decir, cualquier cambio de orientación se aplica en relación a la posición actual de la pieza, en vez de a su posición de origen. Para ver un ejemplo de este comportamiento, cargamos un modelo y creamos un duplicado del mismo para identificar dicho efecto. Seleccionar la primera copia y aplicar una rotación en el eje X de 90 grados además, realizar una rotación en el eje Z de 90 grados. Seleccionamos la segunda copia y aplicamos las mismas rotaciones en el orden inverso: rotación en el eje Z de 90 grados y a continuación, rotación en el eje X de 90 grados. Veremos resultados diferentes en ambas figuras, como se realizó en [18].

Selección de la base.

La herramienta de orientación "Seleccionar base" permite seleccionar la cara de la pieza del modelo que quieres que mire hacia abajo en la base de impresión. Hacemos clic en "Seleccionar base" y a continuación clic en un punto específico de la pieza. PreForm girará automáticamente la pieza hasta la posición requerida.

Una vez haciendo clic en "Seleccionar base", el cursor adquirirá la forma de una flecha naranja cuando pase por encima del modelo.
El modelo rotará para que el área donde hayas hecho clic quede frente a la base de impresión.

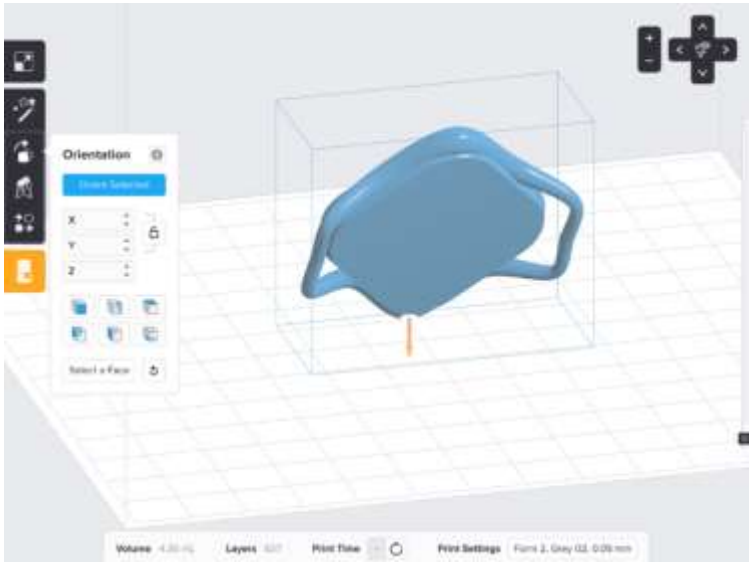


Figura 16. Selección de base.

Esta es una forma rápida de orientar los modelos cuando se tiene definida la "parte frontal" de los mismos. Seleccionar la parte inferior como base puede ayudar a minimizar el trabajo durante el proceso de pos impresión, ya que las marcas de los soportes no estarán en las partes estéticas del modelo que se esta trabajando [18].

Soportes e impresión 3D.

A menudo, los “soportes” en impresión 3D no “soportan” nada. En los procedimientos de sinterizado láser, con base de polvo, por ejemplo, los soportes suelen desempeñar más bien un papel de “tirantes” y de disipadores del calor.

Actúan sobre todo como puntos de tensión para evitar que la pieza se desprenda hacia abajo por las tensiones generadas durante el proceso de fusión láser. También se usan para proporcionar vías de enfriamiento a algunas partes de la pieza que se está imprimiendo, para evitar un enfriamiento excesivamente lento, dando malos resultados en cuanto al acabado superficial de la pieza y el material de la misma.

Figura 17. Soportes.

En la figura 17 podemos ver cómo los soportes se usan a menudo para anclar la pieza al lecho de impresión para que no se despegue debido al estrés residual [16].

3.3.3 Tipos de impresoras 3D.

Existen varios tipos de tecnologías desarrolladas para la impresión de objetos en 3D de los cuales se pueden destacar diferentes gamas o tipos de impresoras, lo hacemos principalmente en función a la tecnología que usan para llevar a cabo la impresión, teniendo esto en cuenta, podemos distinguir los tipos más comunes de impresoras 3D:

Impresoras 3D por Estereolitografía (SLA).

Esta técnica fue la primera en utilizarse. Consiste en la aplicación de un haz de luz ultravioleta a una resina líquida (contenida en un cubo) sensible a la luz. La luz UV va solidificando la resina capa por capa.
La base que soporta la estructura se desplaza hacia abajo para que la luz vuelva a ejercer su acción sobre el nuevo baño, así hasta que el objeto alcance la forma deseada.

Con este método se consiguen piezas de altísima calidad, aunque, por sacar un inconveniente, se desperdicia cierta cantidad de material en función del soporte que sea necesario fabricar.

Algunos ejemplos de impresoras 3D que funcionan por estereolitografía son: Project 1500, 1200 ó 3510 de 3D Systems [19].

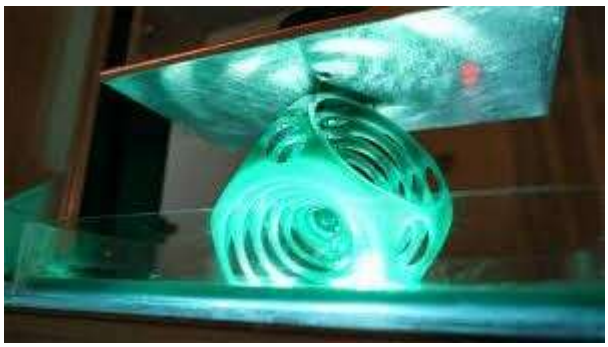


Figura 18. Impresión 3D por estereolitografía.

Impresoras 3D de Sinterización Selectiva por Láser (SLS).

También conocido en inglés como Selective Laser Sintering (SLS), esta tecnología se nutre del láser para imprimir los objetos en 3D.
Nació en los años 80, y pese a tener ciertas similitudes con la tecnología SLA, ésta permite utilizar un gran número de materiales en polvo (cerámica, cristal, nylon, poliestireno, etc.). El láser impacta en el polvo, funde el material y se solidifica. Todo el material que no se utiliza se almacena en el mismo lugar donde inició la impresión por lo que, no se desperdicia nada.

Una de las impresoras 3D más famosas que utilizan esta tecnología de impresión 3D es la EOS.
Con las dos últimas tecnologías se consigue una mayor precisión de las piezas impresas y mayor velocidad de impresión [19].

Figura 19. Impresión 3D de sinterización selectiva por láser.

Impresoras 3D por Inyección.

Este es el sistema de impresión 3D más parecido a una impresora habitual (de tinta en folio), pero en lugar de inyectar gotas de tinta en el papel, inyectan capas de fotopolímero líquido que se pueden curar en la bandeja de construcción. Como ejemplo de impresoras 3D por inyección destacamos X60 de 3D Systems o la Zprint 450 [19].

Figura 20. Impresoras 3D por inyección.

Impresión por deposición de material fundido (FDM).

También conocida por FFF (Fused Filament Fabrication, término registrado por Stratasys), la técnica aditiva del modelado por deposición fundida es una tecnología que consiste en depositar polímero fundido sobre una base plana, capa a capa. El material, que inicialmente se encuentra en estado sólido almacenado en rollos, se funde y es expulsado por la boquilla en minúsculos hilos que se van solidificando conforme van tomando la forma de cada capa. Se trata de la técnica más común en cuanto a impresoras 3D de escritorio y usuarios domésticos se refiere. Aunque los resultados pueden ser muy buenos, no suelen ser comparables con los que ofrecen las impresoras 3D por SLA, por ejemplo. La ventaja principal es que esta tecnología ha permitido poner la impresión 3D al alcance de cualquier persona con impresoras como la CubeX, Prusa o cualquier impresora de RepRap. Actualmente se utilizan una gran variedad de materiales, entre los que predominan ABS y PLA [19].

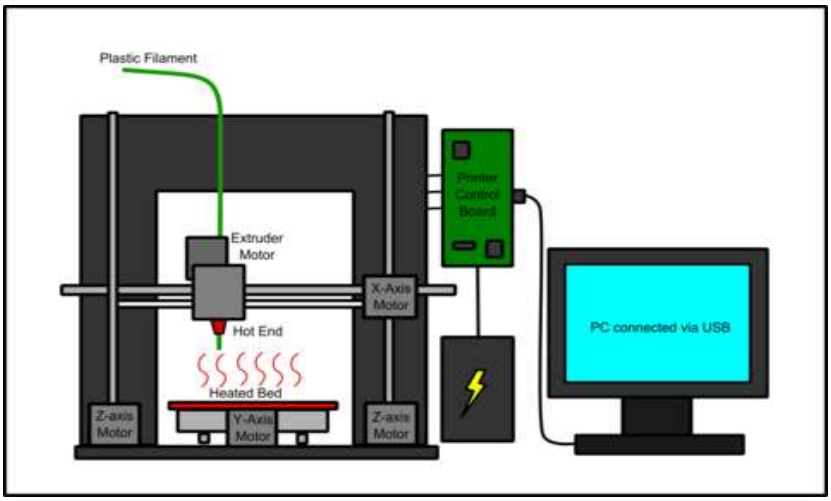


Figura 21. Impresión por deposición de material fundido.

3.4 Impresión del prototipo.

Para la impresión del modelo se usa una impresora 3D del tipo FDM capaz de imprimir figuras con un volumen superior a los 10 cm^3, creando un objeto con 3 dimensiones lo cual se consigue construyendo capas sucesivamente por medio del proceso aditivo, propio de la impresora. Cabe mencionar que para este trabajo se utilizó una impresora modelo Anet A8, como se muestra en la figura 22. La cual resulta muy económica, pero de alta calidad y fácil de ensamblar, además de usar Filamento de impresión del tipo ABS o PLA, en este caso se usó PLA porque es fácil para la impresora la fundición y manipulación de este material. De esta forma se obtuvo el resultado físico del diseño en cuestión.

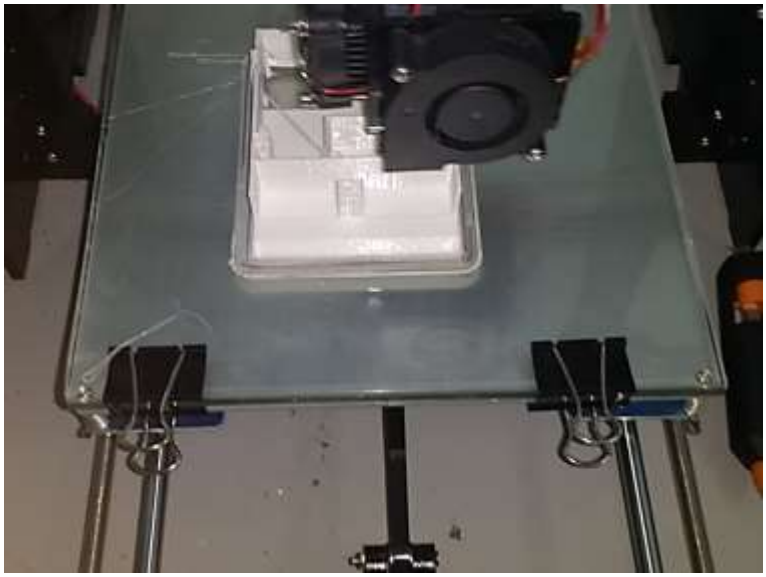


Figura 22. Impresión de diseño físico.

3.5 Módulos y componentes necesarios.

3.5.1 Atmega

Arduino Nano véase la figura 23, 24 y 25, es un módulo de funciones que es muy pequeño con muchas ventajas. Es adecuado para hacer proyectos de dimensiones reducidas, Pudiendo conectar otros componentes electrónicos en un breadborad con ayuda del microcontrolador avr ATmega328, como el visto en la figura 24.

Arduino nano no tiene una toma de CC para que se suministre corriente, por tanto, la alimentación de este se realiza a través del puerto mini-B USB o directamente al pin VCC y GND como se puede ver en el esquema de la figura 24, y se puede suministrar un voltaje de alimentación de 6 a 20 V, a través del puerto USB mini-B. Así también, se puede proporcionar un voltaje de 5 V en el pin 30 (este voltaje no será ajustado por el regulador, así que específicamente debe ser un voltaje de 5 V) [22].

Tabla 3. Especificaciones de Arduino Nano.

Microcontrolador	Atmel ATmega168 o ATmega328
Voltaje de funcionamiento (nivel lógico)	5 V
Voltaje de entrada (recomendado)	7-12 V
Voltaje de entrada (límites)	6-20 V
Pines de E / S digitales	14 (de los cuales 6 proporcionan salida PWM)
Clavijas de entrada analógica	8
Corriente DC por Pin E / S	40 mA

Memoria flash	16 KB (ATmega168) o 32 KB (ATmega328) de los cuales 2 KB utilizados por el gestor de arranque
SRAM	1 KB (ATmega168) o 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) o 1 KB (ATmega328)
Velocidad de reloj	16 MHz
Dimensiones	0.73 "x 1.70"
Longitud	45 mm
Anchura	18 mm
Peso	5 g

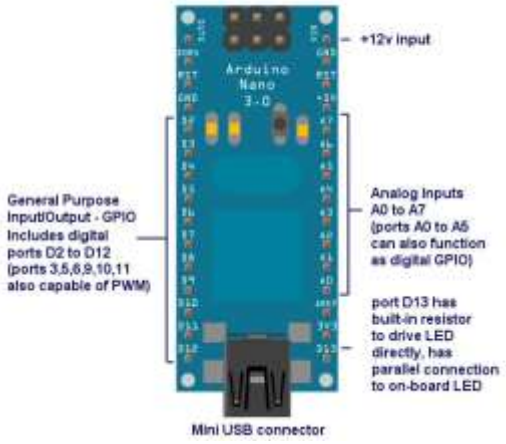


Figura 23. Módulo Arduino Nano.

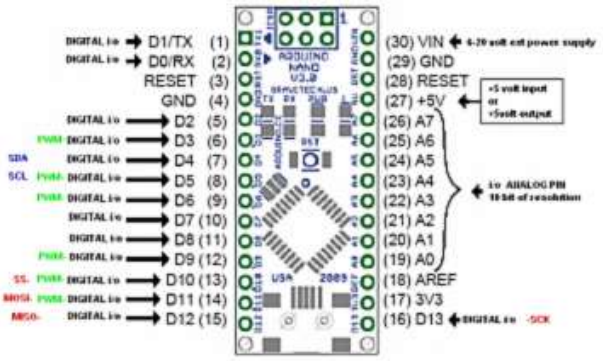


Figura 24. Puerto digitales y analógicos.

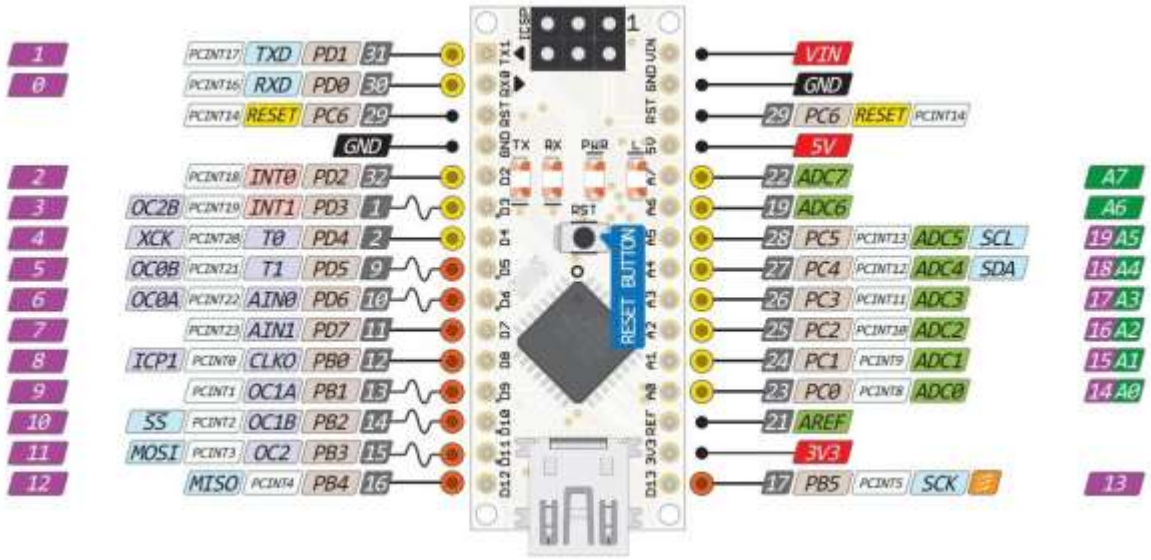


Figura 25. Esquema general de puertos.

3.5.2 Módulo Infrarrojo FC-51.

Un detector de obstáculos infrarrojo es un dispositivo que detecta la presencia de un objeto mediante la reflexión que produce la luz. El uso de luz infrarroja (IR) es simplemente para que ésta no sea visible para el ojo humanos.

Constitutivamente son sensores sencillos, que disponen de un LED emisor de luz infrarroja y de un fotodiodo (tipo BPV10NF o similar) que recibe la luz reflejada por un posible obstáculo como se muestra en la figura 26.

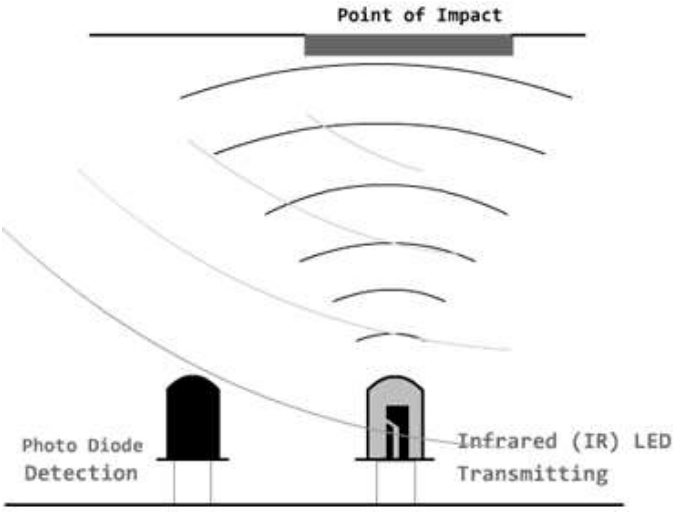


Figura 26. Funcionamiento del IR.

Los detectores de obstáculo suelen proporcionarse con una placa de medición estándar con el comparador LM393, que permite obtener la lectura como un valor digital cuando se supera un cierto umbral, que se regula a través de un potenciómetro ubicado en la placa.

Este tipo de sensores actúan a distancias cortas, típicamente de 5 a 20 mm. Además, la cantidad de luz infrarroja recibida depende del color, material, forma y posición del obstáculo, por lo que no disponen de una precisión suficiente para proporcionar una estimación de la distancia al obstáculo.
Pese a esta limitación, son ampliamente utilizados para la detección de obstáculos en pequeños vehículos o robots. Su bajo coste hace que sea frecuente ubicarlos en el perímetro, de forma que detectemos obstáculos en varias direcciones.

También son útiles en otro tipo de aplicaciones como, por ejemplo, detectar la presencia de un objeto en una determinada zona, determinar si una puerta está abierta o cerrada, ó si una máquina ha alcanzado un cierto punto en su desplazamiento.

Precio.

Los sensores de obstáculos infrarrojos son realmente baratos (figura 27). Podemos encontrar detectores infrarrojos, incluida la placa de medición, por 0.80 US(dólares) como vendedores internacionales de eBay y AliExpress.

Figura 27. Módulo IR FC-51.

Al ser un sensor sencillo también podemos montarlo nosotros mismos. En general no merece la pena ya que sólo los componentes nos costarían más, sin contar el tiempo necesario y la calidad que podríamos obtener, por lo que lo normal es que usemos un modelo comercial.

Esquema eléctrico

El montaje es sencillo. Alimentamos el módulo a través del pin Vcc y GND conectándolos, respectivamente, a la salida de 5V y GND en Arduino.

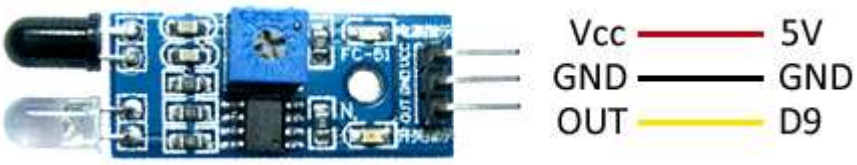


Figura 28. Pins I/O.

Finalmente, conectamos la salida digital del sensor a una entrada digital para leer el estado del sensor como se muestran en las figuras 28 y 29.

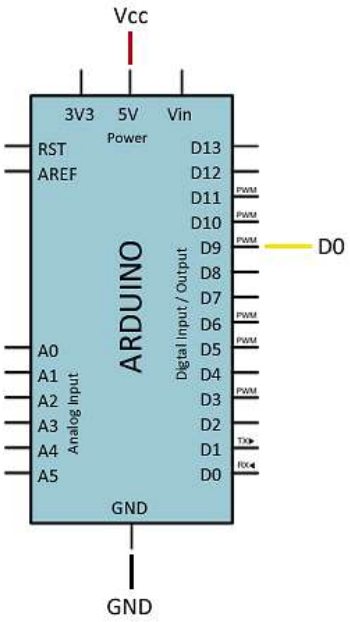


Figura 29. Esquema eléctrico.

Opcionalmente, calibramos el umbral de disparo acercando un objeto al detector de obstáculos y regulando la salida digital con el potenciómetro. Si se desea se puede omitir este paso, para esto dejaremos el potenciómetro en un valor medio. Para hacer uso de nuestro sensor debemos de cargar el código mostrado a continuación. Este es un Algoritmo Analizado, modelado e implementado en la tarjeta de desarrollo que utilizamos (Arduino nano) [20]:

```
1. const int sensorPin = 9;
2.
3. void setup() {
4.   Serial.begin(9600);    //iniciar puerto serie
5.   pinMode(sensorPin , INPUT);   //definir pin como
   entrada
6. }
7.
8. void loop(){
9.   int value = 0;
10.       value = digitalRead(sensorPin );       //lectura
   digital de pin
11.
12.       if (value == HIGH) {
13.           Serial.println("Detectado obstaculo");
14.       }
15.       delay(1000);
16.     }
```

3.5.3 Sensor de ultrasonidos HC-SR04

Un sensor de ultra sonidos (figura 30) es un dispositivo para medir distancias. Su funcionamiento se base en el envío de un pulso de alta frecuencia, no audible por el ser humano. Este pulso rebota en los objetos cercanos y es reflejado hacia el sensor, que dispone de un micrófono adecuado para esa frecuencia.

Midiendo el tiempo entre pulsos, conociendo la velocidad del sonido, podemos estimar la distancia del objeto contra cuya superficie impacto el impulso de ultrasonidos

Los sensores de ultrasonidos son sensores baratos, y sencillos de usar. El rango de medición teórico del sensor HC-SR04 es de 2 cm. a 400 cm., con una resolución de 0.3 cm. En la práctica, sin embargo, el rango de medición real es mucho más limitado, en torno a 20 cm. a 2 metros.

Los sensores de ultrasonidos son sensores de baja precisión. La orientación de la superficie a medir puede provocar que la onda se refleje, falseando la medición. Además, no resultan adecuados en entornos con gran número de objetos, dado que el sonido rebota en las superficies generando ecos y falsas mediciones. Tampoco son apropiados para el funcionamiento en el exterior y al aire libre.

Pese a esta baja precisión, que impide conocer con precisión la distancia a un objeto, los sensores de ultrasonidos son ampliamente empleados. En robótica es habitual montar uno o varios de estos sensores, por ejemplo, para detección de obstáculos, determinar la posición del robot, crear mapas de entorno, o resolver laberintos.

En aplicaciones en que se requiera una precisión superior en la medición de la distancia, suelen acompañarse de medidores de distancia por infrarrojos y sensores ópticos.
Existen otros sensores para detectar distancias. En distancias medias y largas tenemos el sensor óptico Sharp GP2Y0A02YK0 y en muy cortas distancias el detector de obstáculos infrarrojos.

Precio.

El sensor de ultrasonidos HC-SR04 es un sensor barato. Podemos encontrar unidades por 0,65 € en AliExpress.

Figura 30. Módulo ultrasónico.

Funcionamiento.

El sensor se basa simplemente en medir el tiempo entre el envío y la recepción de un pulso sonoro. Sabemos que la velocidad del sonido es 343 m/s en condiciones de temperatura 20 ºC, 50% de humedad, presión atmosférica a nivel del mar. Transformando unidades como se muestra en la ecuación (1).

$$343\frac{m}{s} * 100\frac{cm}{m} * \frac{1}{1000000}\frac{s}{\mu s} = \frac{1}{29.2}\frac{cm}{\mu s} \quad (1)$$

Es decir, el sonido tarda 29,2 microsegundos en recorrer un centímetro. Por tanto, podemos obtener la distancia a partir del tiempo entre la emisión y recepción del pulso mediante el uso de la ecuación (2).

$$Distancia(cm) = \frac{Tiempo(\mu s)}{29.2*2} \quad (2)$$

El motivo de dividir por dos el tiempo (además de la velocidad del sonido en las unidades apropiadas, que hemos calculado antes) es porque hemos medido el tiempo que tarda el pulso en ir y volver, por lo que la distancia recorrida por el pulso es el doble de la que queremos medir.

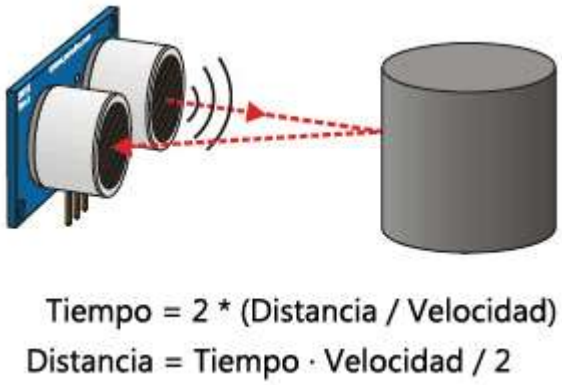


Figura 31. Funcionamiento.

Esquema eléctrico.

El esquema eléctrico del módulo ultrasónico se muestra a detalle en la figura 32.

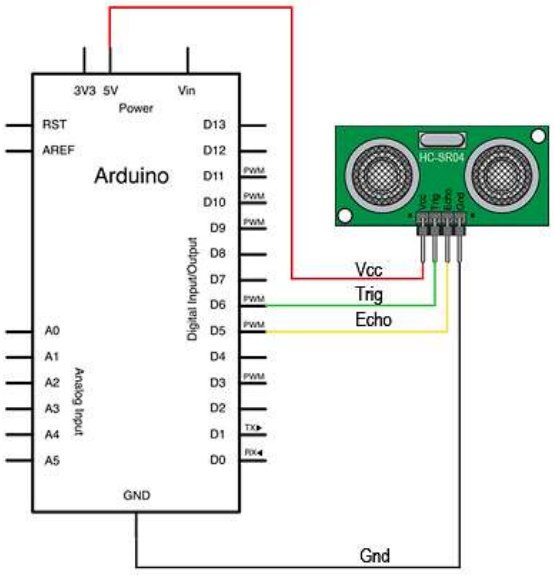


Figura 32. Esquema eléctrico.

Esquema de montaje.

El montaje en una Breadboard se muestra en la figura 33.

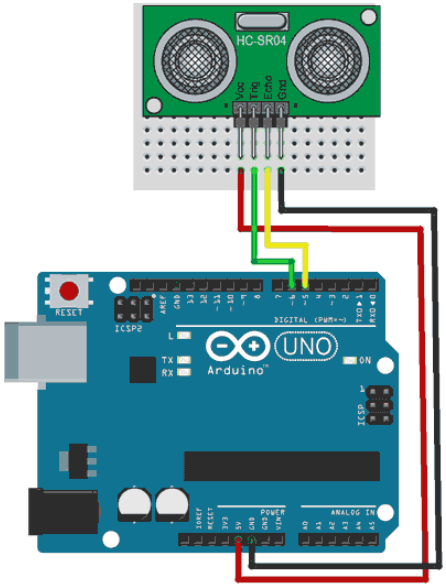


Figura 33. Montado físico.

Código (sin librerías).

Para activar el sensor necesitamos generar un pulso eléctrico en el pin Trigger (disparador) de al menos 10 µs. Previamente, pondremos el pin a Low durante 4 µs para asegurar un disparo limpio.

Posteriormente usamos la función “pulseIn” para obtener el tiempo requerido por el pulso para volver al sensor. Finalmente, convertirnos el tiempo en distancia mediante la ecuación correspondiente.

Observar que intentamos emplear siempre aritmética de enteros, evitando usar números en coma flotante. Esto es debido a que las operaciones en coma

flotante ralentizan mucho el procesador, y suponen cargar un gran número de librerías en memoria [20].

Una vez que cumplimos con el esquema de montaje, cargamos un código de prueba muy similar al siguiente:

```
1. const int EchoPin = 5;
2. const int TriggerPin = 6;
3. const int LedPin = 13;
4.
5. void setup() {
6.    Serial.begin(9600);
7.    pinMode(LedPin, OUTPUT);
8.    pinMode(TriggerPin, OUTPUT);
9.    pinMode(EchoPin, INPUT);
10.     }
11.
12.     void loop() {
13.        int cm = ping(TriggerPin, EchoPin);
14.        Serial.print("Distancia: ");
15.        Serial.println(cm);
16.        delay(1000);
17.     }
18.
19.     int ping(int TriggerPin, int EchoPin) {
20.        long duration, distanceCm;
21.
22.        digitalWrite(TriggerPin, LOW);        //para
  generar un pulso limpio ponemos a LOW 4us
23.        delayMicroseconds(4);
24.        digitalWrite(TriggerPin, HIGH);       //generamos
  Trigger (disparo) de 10us
25.        delayMicroseconds(10);
26.        digitalWrite(TriggerPin, LOW);
27.
28.        duration = pulseIn(EchoPin, HIGH);    //medimos
  el tiempo entre pulsos, en microsegundos
29.
30.        distanceCm = duration * 10 / 292/ 2;           /
  /convertimos a distancia, en cm
```

```
31.         return distanceCm;
32.     }
```

Código con librería NewPing.

Otra opción es emplear una librería para facilitarnos el proceso, como por ejemplo la librería NewPing disponible en el gestor de librerías del IDE de Arduino.

La librería NewPing aporta funciones adicionales, como la opción de realizar un filtro de mediana para eliminar el ruido, o emplear el mismo pin como trigger y echo, lo que nos permite ahorrar muchos pines en caso de tener múltiples sensores de ultrasonidos.

La librería proporciona diversos ejemplos para ilustrar su uso. En la tabla 6 se muestra el funcionamiento con un único pin para trigger y echo [20], aquí un ejemplo de cómo aplicar dicha librería:

```
1. #include <NewPing.h>
2. const int UltrasonicPin = 5;
3. const int MaxDistance = 200;
4.
5. NewPing sonar(UltrasonicPin, UltrasonicPin,
   MaxDistance);
6.
7. void setup() {
8.   Serial.begin(9600);
9. }
10.
11.     void loop() {
12.       delay(50);                              // esperar
   50ms entre pings (29ms como mínimo)
```

```
13.       Serial.print(sonar.ping_cm());          // obtener
  el valor en cm (0 = fuera de rango)
14.       Serial.println("cm");
15.     }
```

3.5.4 Módulo L9110S (motor driver).

El L9110 es un circuito impreso compuesto por 2 chipset modelo L9110S, que trabajan entre 2,5 v y 12 v (recomendados 5 a 12 voltios) y con un amperaje de 800 mA. Las dimensiones de este driver son de 3.1 cm x 2.2 cm x 1.2 cm y un peso de 7 g. El coste de driver ronda los 3,53 €

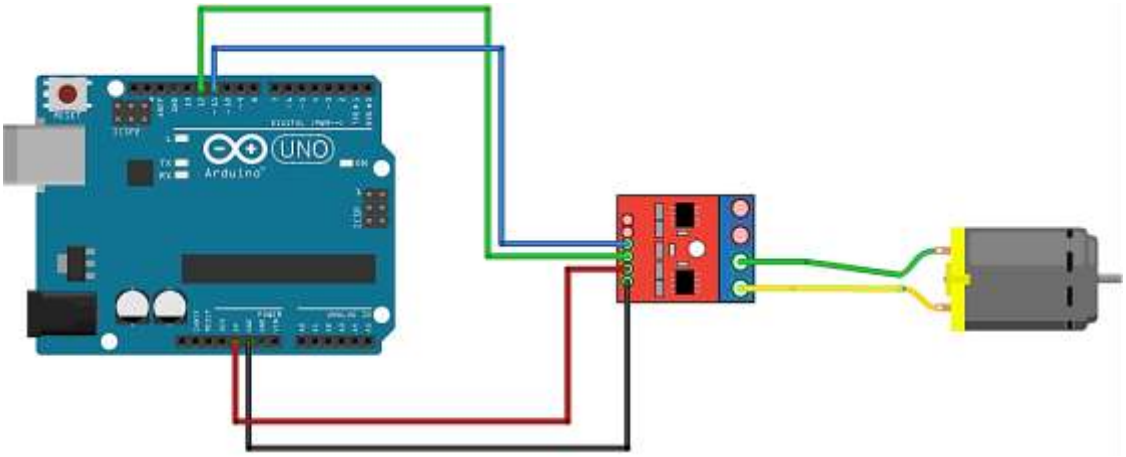


Figura 34. Montado del driver.

Es importante mencionar, que al estar alimentando nuestro motor a través de nuestro driver como se muestra en la figura 34.

El driver L9110S puede trabajar hasta 12v (pero nunca a más de 1 amperio para evitar daños al driver). Para conseguir llegar hasta 12v tendremos que suministrar este voltaje a través de una fuente externa [20].

Código.

Una vez realizado las conexiones entre nuestro módulo y la tarjeta arduino, cargamos un código similar al siguiente:

```
1. int M1_Izq = 12;          //Dirección
2. int M1_Derecha = 11;       //Dirección
3.
4. void setup()
5. {
6.   pinMode(M1_Izq, OUTPUT);
7.   pinMode(M1_Derecha, OUTPUT);
8. }
9.
10.     void loop(){
11.       girar (1);
12.       delay(1000);  //1 sg
13.
14.       stop();
15.       delay(250);   //250ms
16.
17.       girar (2);
18.       delay(1000);  //1 seg
19.
20.       stop();
21.       delay(250);   //250ms
22.     }
23.     void girar(int direccion)
24.     {
25.       boolean inPin1 = LOW;
26.       boolean inPin2 = HIGH;
27.
```

```
28.       if(direccion == 1){
29.         inPin1 = HIGH;
30.         inPin2 = LOW;
31.       }
32.         digitalWrite(M1_Izq, inPin1);
33.         digitalWrite(M1_Derecha, inPin2);
34.     }
35.
36.     void stop(){
37.         digitalWrite(M1_Izq, LOW);
38.         digitalWrite(M1_Derecha, LOW);
39.     }
```

Como podemos observar en el código, el eje del motor se moverá en 2 direcciones a través de un booleano que se debe definir en los 2 datos que le enviáis a través de la salida digital " digitalWrite (M1_Izq, inPin1); digitalWrite (M1_Derecha, inPin2);".

3.5.5 Motores geared down.

Un motor geared down es un motor de corriente continua que incorpora un reductor interno. Esto aumenta el par del motor y reduce su velocidad. Velocidades de giro habituales son 60, 120, 240 y 480 rpm, entre otras.

Figura 35. Interior de un motorreductor.

Es frecuente que algunos motores geared down incorporan un encoder interno. Este encoder suele estar aplicado en el lado de alta velocidad, por lo que la precisión es superior a añadir una encoder acoplado al eje.

Los motores geared down son frecuentes para accionar ruedas de robots y vehículos.

Figura 36. Tipos de motorreductor.

3.5.6 Rueda Loca (caster wheel)

Es una rueda sin tracción, simple o doble, que puede girar libremente y que generalmente está situada en la parte inferior de una estructura. Se utiliza en carros de la compra, sillas de oficina o móviles.

Existen dos tipos de rueda loca:

- Rígida Sujeta a la superficie con una estructura rígida que le permite sin embargo girar adelante y atrás (figura 37).

Figura 37. Rueda loca rígida.

- De pivote o Rotatoria Sujeta a la superficie con una estructura que tiene un eje en su centro, anclado a la rueda que puede girar libremente.

Figura 38. Rueda loca de pivote.

Este último tipo de rueda tiene una ventaja fundamental, y es que puede girar en todas direcciones, sin embargo, esta facilidad puede convertirse también un problema, conocido como aleteo. El aleteo es ese característico movimiento de vaivén o zigzag que experimentan, por ejemplo, los carritos del súper mercado cuando no tienen peso.

3.5.7 Cable DuPont

Un cable puente para prototipos (o simplemente puente para prototipos), es un cable con un conector en cada punta (o a veces sin ellos), que se usa normalmente para interconectar entre sí los componentes en una placa de pruebas. P.E.: se utilizan de forma general para transferir señales eléctricas de cualquier parte de la placa de prototipos a los pines de entrada/salida de un microcontrolador.

Los cables puente se fijan mediante la inserción de sus extremos en los agujeros previstos a tal efecto en las ranuras de la placa de pruebas, la cual debajo de su superficie tiene unas planchas interiores paralelas que conectan las ranuras en grupos de filas o columnas según la zona. Los conectores se insertan en la placa de prototipos, sin necesidad de soldar, en los agujeros que convengan para el conexionado del diseño.

Hay distintos tipos de cables puente, por ejemplo

- Con pinzas cocodrilo

  Los hay que llevan pinzas cocodrilo en lugar de conectores terminales que, entre otras aplicaciones, se utilizan temporalmente para puentear los sensores, botones y otros elementos de los prototipos entre sí y con los microcontroladores.

- Con terminales aislados

En el tipo con terminales aislados la disposición de los elementos y la facilidad de insertar los "conectores aislados" de los "cables puente" sobre la placa de pruebas permite el incremento de la densidad de montaje de ambos (componentes y puentes) sin temor a los cortocircuitos. Los cables puente varían en tamaño y color para distinguir las señales con las que se está trabajando.

Variación de cables puente con terminales esmaltados, según las combinaciones macho-hembra:

- Macho - macho
- Macho - hembra
- Hembra – hembra

Figura 39. Cable DuPont.

4 Diseño y creación de la política.

En la tabla 8, se enlistan los materiales requeridos y su costo en relación a producto y cantidad, cabe mencionar que dichos materiales deben conseguirse con un proveedor el cual maneje costes de producción directos de fábrica, esto para reducir costos y obtener ganancias para la rentabilidad del proyecto.

Tabla 4. Tabla de balance de costos.

PRODUCTO	PRECIO UNIDAD	CANTIDAD	PRECIO FINAL
Motor reductor	22.5	40	900
puente h	16.52	20	330.4
infrarrojo	9.35	20	187
ultrasónico	16.4	20	328
rueda loca	4.35	20	87
arduino nano	42.05	20	841
AA Case Baterías	9.37	20	187.4
Pegamento	2.4	10	24
caucho	1	30	30
tornillos	100	1	100
ABS	450	1	450
DuPont h/h	0.68625	80	54.9
switch	2.923076923	13	38
Inversión Total			3557.7

Como se puede observar muchos de los materiales son de fácil adquisición, pero de costos elevados si no se busca el correcto proveedor, por tanto, se optó por hacer la compra con uno o varios proveedores extranjeros los cuales garantizan

precios bajos y una buena calidad en sus productos lo cual nos ayuda a disminuir los costes lo suficiente como para obtener ganancias cercanas al 300% $9,442.3 MXN de ganancia en relación a la inversión inicial de $3557.00 MXN.
Cabe destacar que los inconvenientes de ésta estrategia de compra son:

- Costos de envío elevados=tiempos de entrega cortos
- Costos de envió económicos=tiempo de entrega largos

Sí, los costos de envío son elevados, sin lugar a duda nuestro pedido llegará más rápido y con una protección de pedido, no sólo por parte del proveedor, sino también por parte de la agencia de envíos, pero como es fácil apreciar, estos beneficios extras solo son funcionales si los pedidos son muy grandes, ya que los costes internacionales de envío ascienden arriba de $1,000.00MXN lo cual no sería redituable a menos que en relación a la ganancia y producción las unidades absorban dicho costo extra con pedidos masivos, o que solo se maneje un proveedor lo cual no siempre es la mejor opción. En el caso de los costos bajos hablamos de una carencia por parte de la empresa que realiza el envío y solo quedamos con la protección del proveedor lo cual supone ciertas dificultades en el caso de que el envío no llegue a su destino. Además, se tiene un tiempo excesivamente largo de espera, el cual puede variar de un mes hasta 2 meses, y en el peor de los casos el producto llegara a su destino, por diversas situaciones, la más común como ser retenido en aduana o perdida por el servicio de paqueterías.

5 Resultados

5.1 Software

En este punto se aplicó el uso de todos los módulos mencionados en el desarrollo de este documento los cuales permitieron el correcto funcionamiento de nuestro producto final, para eso implementamos un código que activa todos los sensores y los hace trabajar en conjunto detectando objetos a una distancia mayor a 20cm y detectando el espectro sin reflexión de luz a reflexión de luz realizando una toma de decisiones dependiendo de dichas entradas y salidas de los valores adquiridos por estos módulos. Una vez montado todo correctamente y verificado las conexiones, cargamos el siguiente código:

```
1. const int sensorPin = 9;
2. int IN3 = 3;
3. int IN4 = 4;
4. int IN6 = 6;
5. int IN5 = 5;
6. const int Trigger = 7;          //Pin digital 7 para el
   Trigger del sensor
7. const int Echo = 8;     //Pin digital 8 para el Echo del
   sensor
8.
9. void setup() {
10.       Serial.begin(9600);           //iniciar puerto
   serie
11.       pinMode(sensorPin , INPUT);    //definir pin como
   entrada
12.       pinMode (IN4, OUTPUT);        // Input4 conectada
   al pin 4
```

```
13.       pinMode (IN3, OUTPUT);           // Input3
  conectada al pin 3
14.       pinMode (IN5, OUTPUT);         // Input4 conectada
  al pin 5
15.       pinMode (IN6, OUTPUT);         // Input3 conectada
  al pin 6
16.       pinMode(Trigger, OUTPUT);      //pin como salida
17.       pinMode(Echo, INPUT);           //pin como entrada
18.       digitalWrite(Trigger, LOW);    //Inicializamos el
  pin con 0
19.     }
20.
21.     void loop(){
22.       long t;         //tiempo que demora en llegar el
  eco
23.       long d;        //distancia en centímetros
24.       digitalWrite(Trigger, HIGH);
25.       delayMicroseconds(10);                  //Enviamo
  s un pulso de 10us
26.       digitalWrite(Trigger, LOW);
27.       t = pulseIn(Echo, HIGH);       //obtenemos el
  ancho del pulso
28.       d = t/59;               //escalamos el tiempo a
  una distancia en cm
29.
30.       int value = 0;
31.       value = digitalRead(sensorPin );        //lectura
  digital de pin
32.     //BUSCAR OBJETIVO
33.       if ((value == HIGH)&&(d >20)) {
34.       Serial.println("zona oscura");
35.       digitalWrite (IN3, LOW);
36.       digitalWrite (IN4, HIGH);
37.       digitalWrite (IN5, HIGH);
38.       digitalWrite (IN6, LOW);
39.       delay(500);
40.     }
41.             //EVITAR GOLPE
42.     if ((value == LOW)&&(d >20)){
43.       Serial.println("zona blanca");
44.       digitalWrite (IN3, HIGH);
45.       digitalWrite (IN4, LOW);
46.       digitalWrite (IN5, LOW);
47.       digitalWrite (IN6, HIGH);
```

```
48.       delay(500);
49.             //AVANZAR AL FRENTE
50.       digitalWrite (IN3, HIGH);
51.       digitalWrite (IN4, LOW);
52.       digitalWrite (IN5, HIGH);
53.       digitalWrite (IN6, LOW);
54.      delay(100);
55.     }
56.             //EMPUJAR OBJETO SOBRE LINEA
57.     if ((value == LOW)&&(d <20)){
58.       Serial.println("zona blanca");
59.      digitalWrite (IN3, HIGH);
60.       digitalWrite (IN4, LOW);
61.       digitalWrite (IN5, HIGH);
62.       digitalWrite (IN6, LOW);
63.      delay(200);
64.     }
65.             //EMPUJAR OBJETIVO
66.     if ((value == HIGH)&&(d <20)){
67.        digitalWrite (IN3, HIGH);
68.       digitalWrite (IN4, LOW);
69.       digitalWrite (IN5, HIGH);
70.       digitalWrite (IN6, LOW);
71.      delay(2000);
72.     }
73.
74.       delay(20);
75.     }
```

Durante la etapa de desarrollo se obtuvieron 5 versiones del producto las cuales fueron mejorando a su siguiente revisión, a continuación, se enlistan todas las versiones y sus principales características.

5.2 Versiones del diseño

5.2.1 Prototipo 1

La versión original (figura 40), era más robusta y consumía más material de impresión, estaba planeada para recubrir todos los componentes internos del móvil (figura 43, 44 y 45) y contaba con 4 piezas impresas para su ensamblaje:

- 1 Chasis (figura 40).
- 1 Tapa de chasis (figura 42).
- 2 Ruedas (figura 41).

Ventajas: es una versión más completa en estructura respecto a las versiones posteriores, tiene robustez y hace uso óptimo de su espacio para almacenar componentes en su interior.

Desventajas: al ser una versión tan robusta los costes de producción son muy altos y ciertamente innecesarios, en su diseño se encontraron errores de espacio, soportes y ranuras para los dispositivos, teniendo de esta manera una posible incompatibilidad con algunos de sus componentes.

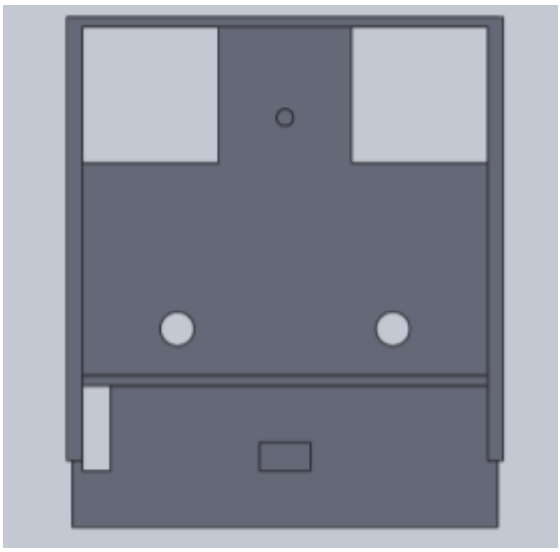


Figura 40. Chasis prototipo 1.

Figura 41. Llanta prototipo 1.

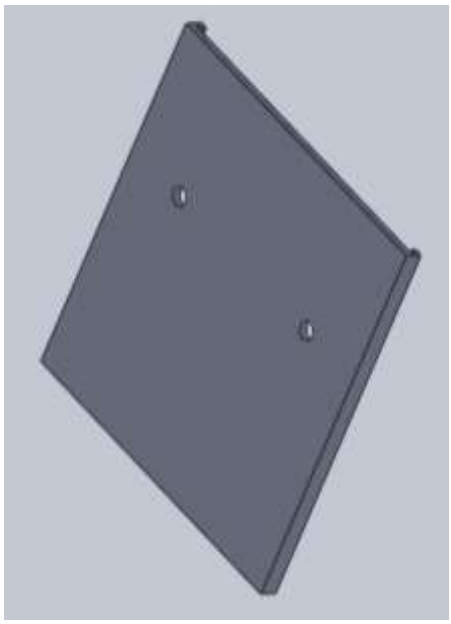


Figura 42. Tapa de chasis prototipo 1.

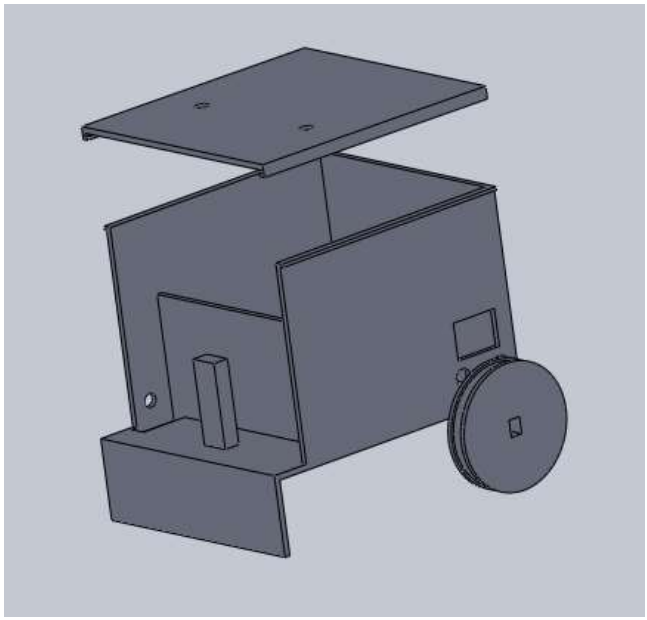


Figura 43. Vista lateral prototipo 1.

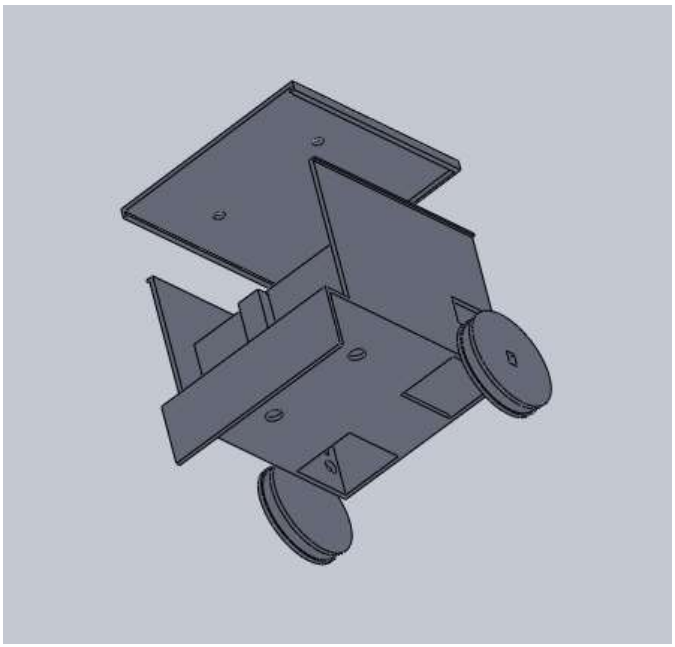


Figura 44. Vista inferior prototipo 1.

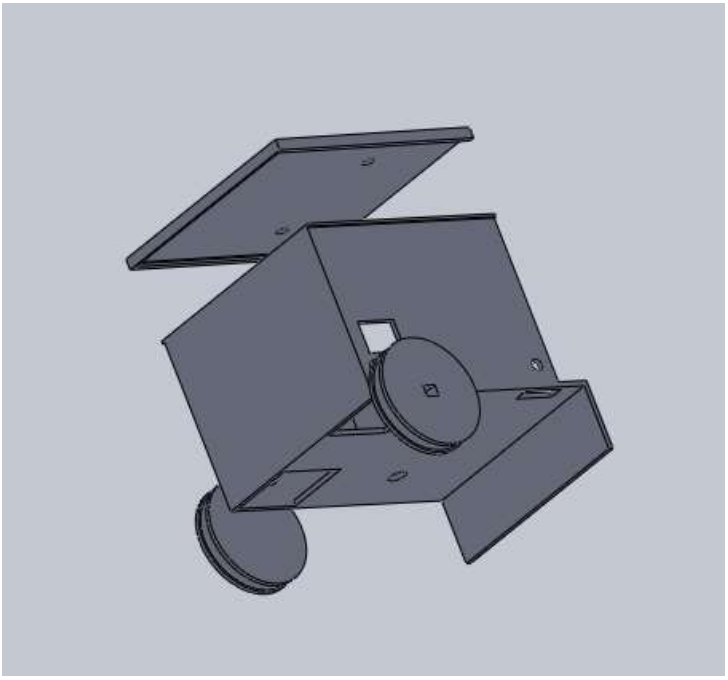


Figura 45. Vista trasera prototipo 1.

5.2.2 Prototipo 2

Este diseño es una mejora sustancial del primer prototipo, su diseño es más estético y menos robusto, algunas de sus ventajas y desventajas son:

- Ventajas: además de contar con compartimentos especiales diseñados para una fácil colocación de la placa arduino nano, también se agregaron perforaciones extra para acoplar sensores de forma externa y facilitar el cableado al interior del chasis, se incluyó una ranura interna para Arduino nano Mini USB, además de poner un pilar de sujeción para acomodar una bisagra en lugar de la tapa del prototipo 1, la cual está pensada para acomodar nuestro Battery Case a modo de tapa, y por último se modificaron las llantas mostrando una parte lisa en la cara frontal además de un riel menos amplio para la adición de caucho para la tracción.

- Desventajas: Aunque se ha mejorado el diseño se encontraron errores de espaciado en el posicionamiento de los motorreductores además de que esta versión sigue consumiendo bastante material en su fabricación debido a las altas paredes del modelo y por ultimo cabe mencionar que el diseño cuadrado de las bisagras no era el óptimo para hacer girarlas.

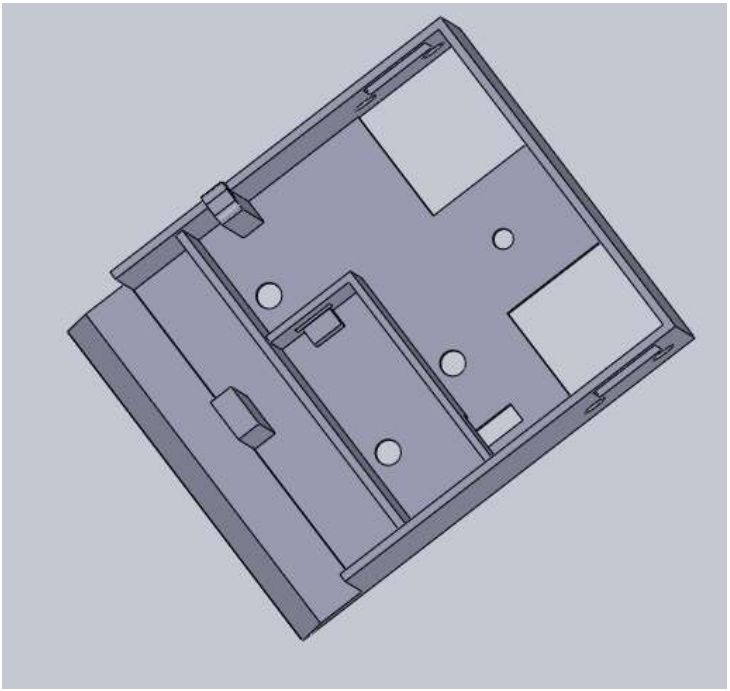


Figura 46. Vista superior prototipo 2.

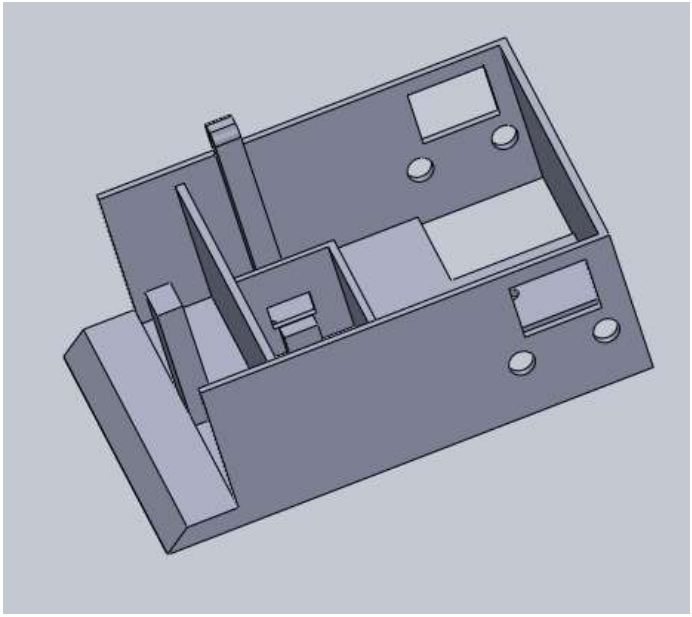


Figura 47. Vista lateral prototipo 2.

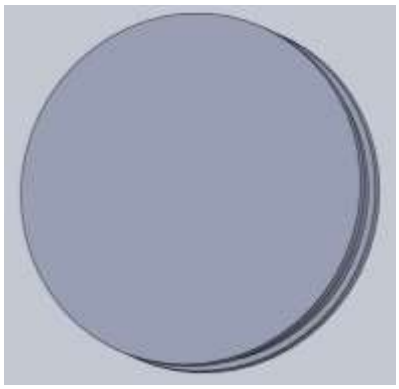


Figura 48. Vista frontal de rueda.

Figura 49. Vista trasera de rueda.

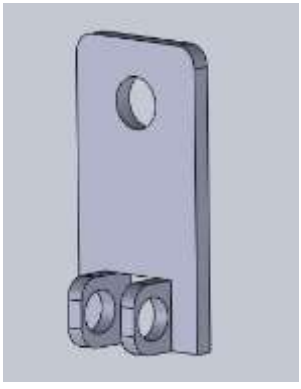


Figura 50. Bisagra vista frontal.

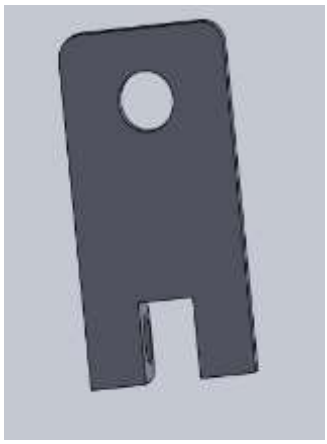


Figura 51. Bisagra vista trasera.

5.2.3 Prototipo 3

En esta versión del Prototipo se agregaron las siguientes modificaciones con sus ventajas y desventajas:

- Ventajas: Inclusión de 3 pilares para dar un soporte extra al Battery Case además de que se disminuyó el tamaño de las paredes del prototipo, también se corrigió el error de las bisagras, además de que se redujo un poco el tamaño del riel de las ruedas para evitar mucho espaciado en la posición del caucho y se ajustó un poco más la ranura para el eje del motor reductor, cabe mencionar que esta sería la versión definitiva de las llantas.
- Desventajas: La mayor desventaja de este prototipo es que se sigue consumiendo mucho material en la fabricación del chasis y sigue manteniendo muchos de los errores de diseño del mismo, la bisagra también sufre de fragilidad por el grosor de las paredes en su diseño.

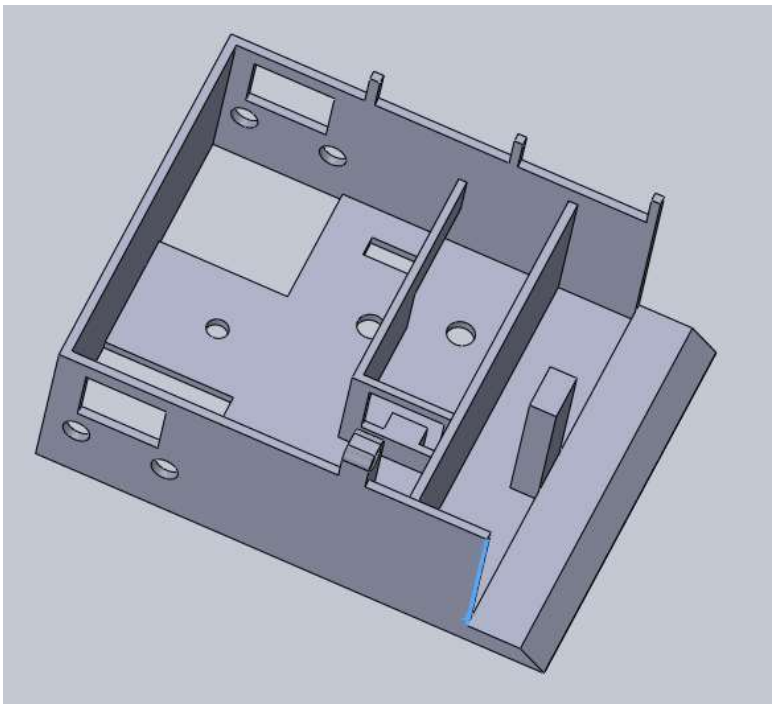


Figura 52. Vista lateral prototipo 3.

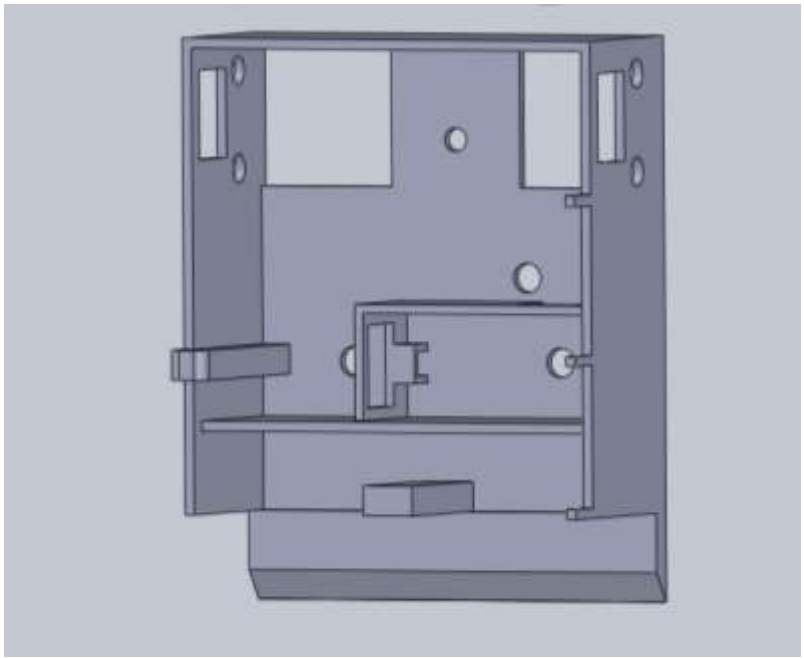


Figura 53. Vista superior prototipo 3.

Figura 54. Vista trasera llanta.

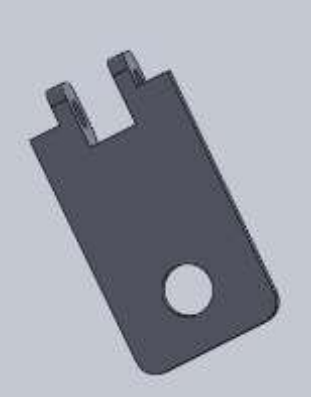


Figura 55. Vista trasera bisagra.

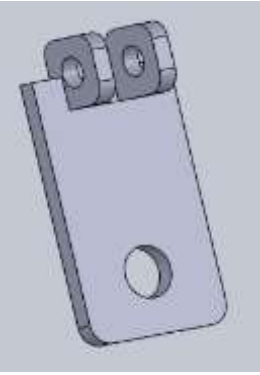


Figura 56. Vista frontal bisagra.

5.2.4 Prototipo 4

- Ventajas: se redujeron los pilares extras para el battery case a 2 evitando consumo extra de material en su fabricación y cumpliendo así la misma función, se redujo las paredes del chasis reduciendo todavía más el costo de producción y mejorando su relación calidad precio, las bisagras se hicieron más robustas en sus paredes para resistir la carga del battery case, siendo esta su última versión.

- Desventajas: existe cierta fragilidad en el diseño del posicionamiento de los motorreductores causando quiebres además que los pilares se notan antiestéticos y poco funcionales, también se planteó el agregado de un orificio para botón switch que no estaba previsto en ninguno de los diseños anteriores.

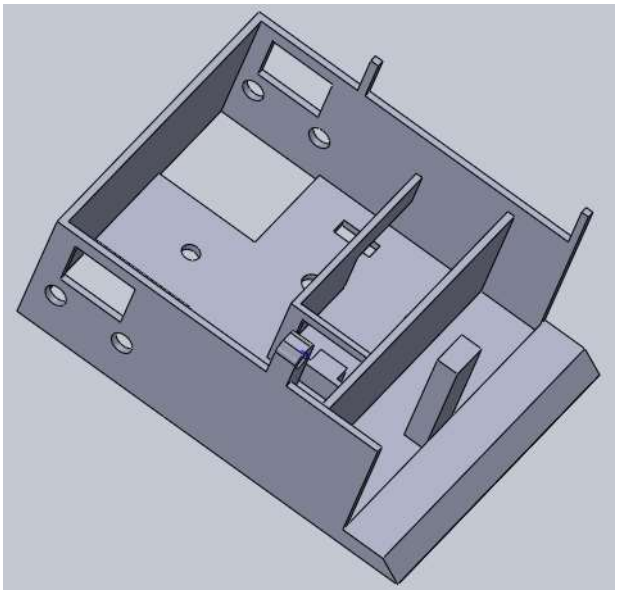


Figura 57. Vista lateral prototipo 4.

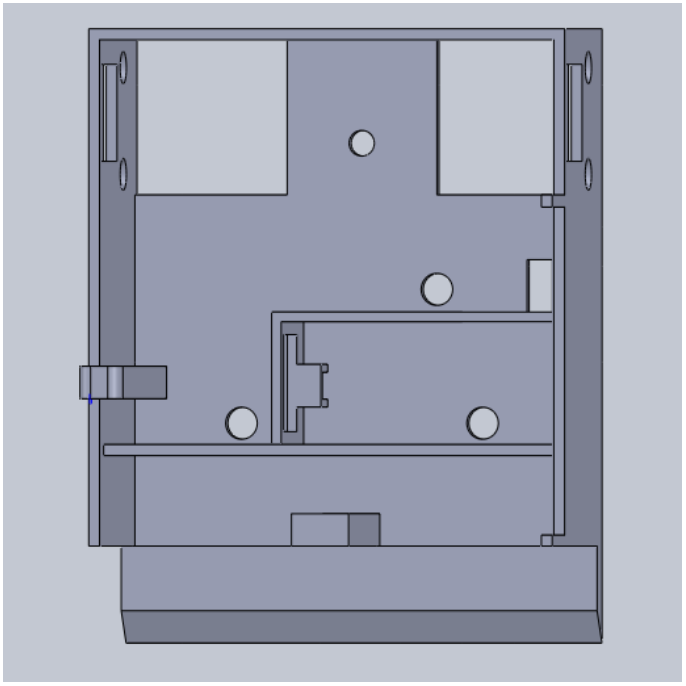


Figura 58. Vista superior prototipo 4.

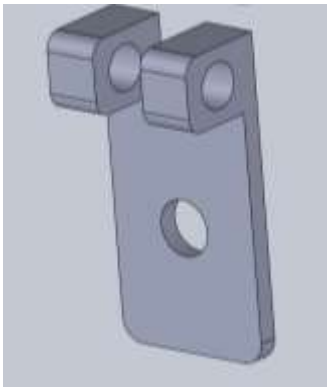


Figura 59. Vista frontal bisagra.

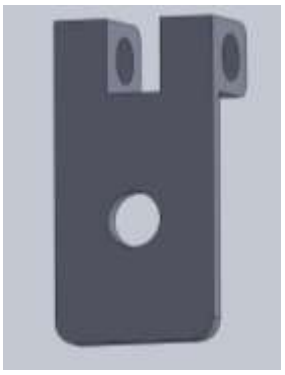


Figura 60. Vista trasera bisagra.

5.2.5 Prototipo 5

Es la versión más actual del prototipo, con esta versión se culminó la parte de diseño y fabricación siendo este el modelo usado para la promoción del producto sus principales características son:

- Reducción de paredes en el chasis sin comprometer la calidad del producto
- Solución de falla estructural en el posicionamiento del Motor Reductor
- Inclusión de ranura interna/externa para Arduino nano Mini USB
- Inclusión de soportes extra para Slot de Arduino Nano
- Inclusión de perforaciones en la pala del Robot para usos varios (postes de equilibrio)
- Inclusión de Ranura para botón Switch.

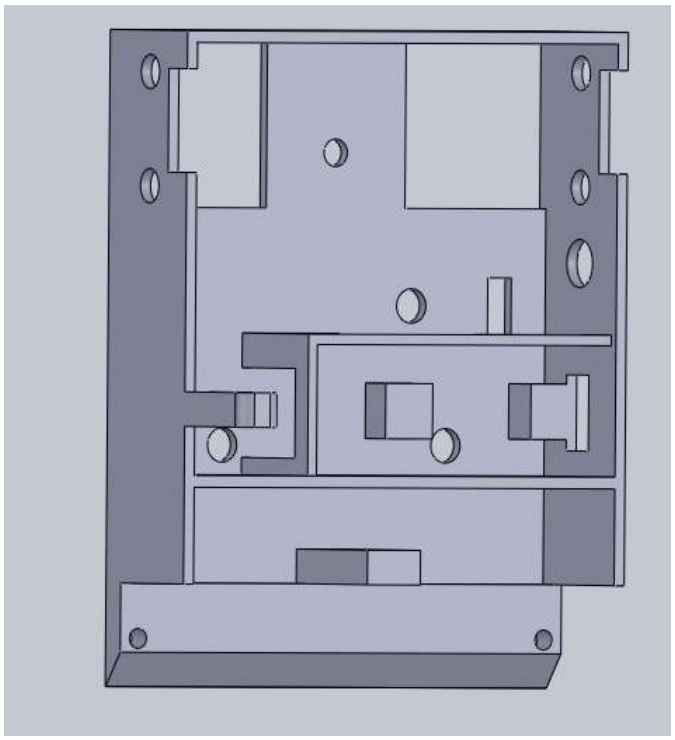


Figura 61. Vista superior prototipo 5.

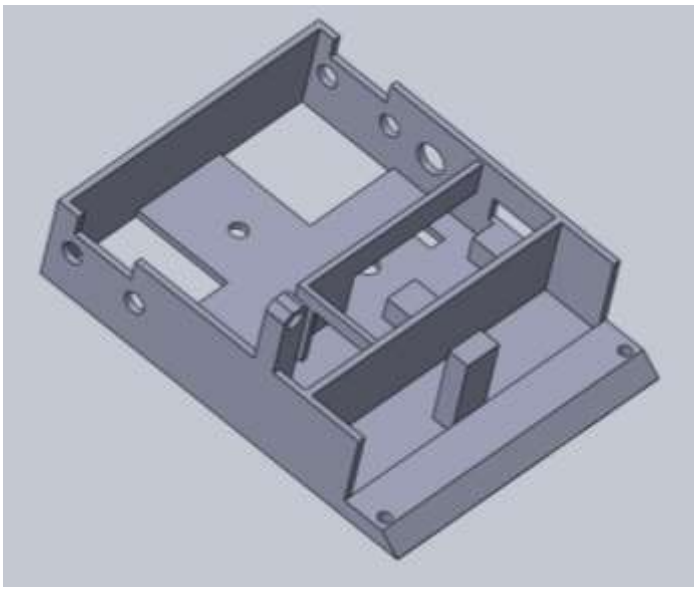


Figura 62. Vista lateral prototipo 5.

5.3 Resultados

El producto fue terminado como se muestra en las figuras 63 y 64, y en ellas podemos denotar varias implementaciones del mismo aprovechando su diseño para la distribución de sus componentes internos, además de su fuente de alimentación, la cual puede ser adaptada según las necesidades del usuario.

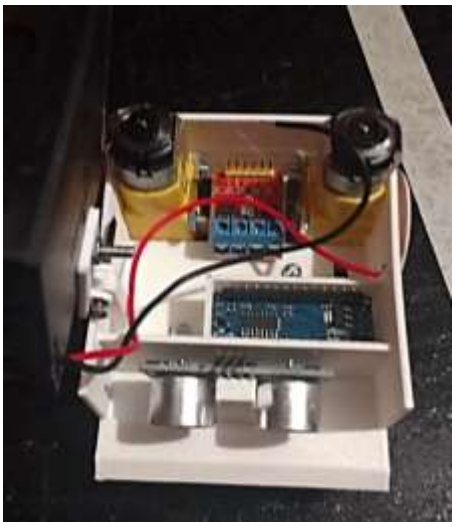


Figura 63. Móvil vista superior.

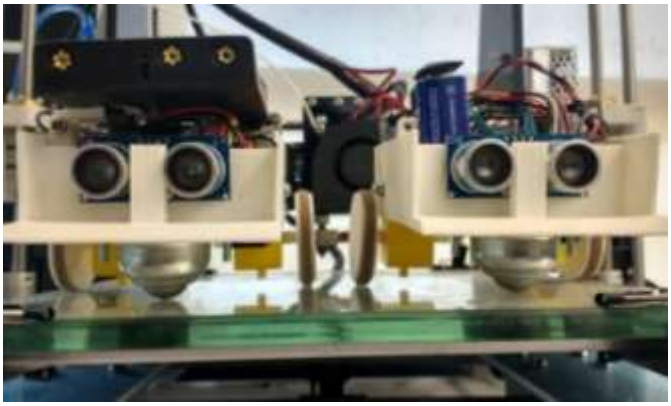


Figura 64. Móvil vista frontal.

Una vez concluidas todas las etapas se realizó la comercialización del producto (figura 65) por medio de internet y carteles para una propagación media.

Las pruebas del producto fueron exitosas y aunque se tuvo problema con algunos materiales a la hora de hacer pruebas en general los usuarios que hicieron uso de los robots quedaron satisfechos, lo cual nos lleva a la etapa de distribución y fabricación de más piezas, ahora aplicadas con un enfoque educativo.

Figura 65. Propaganda.

6 Conclusiones.

Las conclusiones de este trabajo de tesis consisten en el desarrollo de una plataforma o Framework que manipulé y programé un robot móvil. Gracias a esta, se podrá aplicar otros algoritmos que se basan en diferentes áreas como la inteligencia artificial, cómputo distribuido, sistemas de tiempo real, planificación de movimientos de robot, entre otras. De esta forma se podrán realizar experimentos, en trabajos futuros. Otra aportación es la documentación completa del modelado cinemático y dinámico, programación y control de una arquitectura de un móvil tipo clásico miniatura que se aportará a la FCC-BUAP. Por último, aportamos a la Ingeniería participando en la convocatoria de patentes de la BUAP.

El área de desarrollo de nuevas tecnologías está creciendo de manera exponencial así como la aparición de nuevos campos de estudio y modificación de modelos educativos los cuales sustituyen a los departamentos de cómputo por una gestión que implique no solo el desarrollo y administración de sistemas computacionales, si no el desarrollo de hardware y tecnologías capaces de facilitar nuestras labores diarias, es decir de nuevas tecnologías propias inyectadas en una sociedad que ha alcanzado una comunión con la tecnología.

Por tanto, se decidió enfocar ese trabajo en una iniciativa, desarrollando una cultura de propuestas innovadoras y tecnológicas, así también aportamos al aprendizaje tanto de lenguajes de programación como electrónica básica en alumnos de educación media, media superior y superior, preparándolos así para insertarse a un entorno más profesional, con una dificultad y competitividad

mayores, y que estos sean capaces de superar las adversidades que este nuevo entorno estudiantil les presenté.

7 Bibliografía.

1. odroid. (7 de marzo de 2017). odroid. Obtenido de http://odroid.com/dokuwiki/doku.php?id=en:odroid_flashing_tools.
2. nanuyo. (27 de julio de 2016). GitHub, Inc. Obtenido de https://github.com/nanuyo/opencvsh_for_ubuntu_mate/blob/master/opencv_install_to_ubuntu_mate_16.04.sh
3. odriod. (27 de abril de 2017). ameriDroid.com. Obtenido de http://ameridroid.com/
4. Alessandro Giusti et all. A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots. Browse Journals & Magazines IEEE Robotics and Automation. Volume: 1 Issue: 2. December 2015.
5. P. Santana, L. Correia, R. Mendonça, N. Alves and J. Barata, "Tracking natural trails with swarm-based visual saliency", J. Field Rob., vol. 30, no. 1, pp. 64-86, 2013.
6. Aufrere, R., Chapuis, R., & Chausse, F. (2001). A model-driven approach for real-time road recognition. Machine Vision and Applications, 13(2), 95-107.
7. Zavala-Río, A., Fantoni, I., & Lozano, R. (2003). Global stabilization of a PVTOL aircraft model with bounded inputs. International Journal of Control, 76(18), 1833-1844.
8. Duda, R. O., & Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. Communications of the ACM, 15(1), 11-15.
9. Latombe, Jean-Claude. Robot Motion Planning. Kluwer Academic Publishers 1991.
10. Carrillo, L. R. G., Flores Colunga, G. R., Sanahuja, G., & Lozano, R. (2014). Quad rotorcraft switching control: An application for the task of path following. Control System s Technology, IEEE Transactions on, 22(4), 1255-1267.
11. Frazzoli, E., Dahleh, M. A., & Feron, E. (2002). Real-time motion planning for agile autonomous vehicles. Journal of Guidance, Control, and Dynamics, 25(1), 116-129.
12. Choset, H. M. (2005). Principles of robot motion: theory, algorithms, and implementation. MIT press. L.E.Dubins, “On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents,” American Journal of Mathematics, vol. 79, no. 3, pp. 497–516, jul 1957.
13. Lugo-Cárdenas, I., Flores, G., Salazar, S., & Lozano, R. (2014, May). Dubins path generation for a fixed wing UAV. In Unmanned Aircraft Systems (ICUAS), 2014 International Conference on (pp. 339-346). IEEE.
14. NVIDIA Inc. Embedded Systems. Jetson TX1 developer kit Full-featured development platform for visual computing. http://www.nvidia.com/object/jetson-tx1-dev-it.html#sthash.xM9kPgjh.dpuf

15. henrypadilla1997 (2015). Manipuladores son sistemas mecánicos. Universidad Nacional Autónoma de México. Recuperado de https://www.coursehero.com/file/p37cltl/Manipuladores-Son-sistemas-mecánicos-multifuncionales-con-un-sencillo-sistema/
16. TRSD. Soportes e impresión 3D: lo que nunca te cuentan Recuperado de https://impresiontresde.com/soportes-e-impresion-3d-lo-que-nunca-te-cuentan/
17. Autor Anonimo (2012). CURA – Software para Impresión 3D – Sprinter Vs Marlin. Recuperado de https://www.tr3sdland.com/2012/12/software-cura-para-impresion-3d-sprinter-vs-marlin/
18. Knowledge(2018).Orientación de los modelos. Recuperado de https://support.formlabs.com/s/article/Model-Orientation?language=es
19. Impresoras3D.com (2017). Tipos de impresoras 3D. Recuperado de https://www.impresoras3d.com/tipos-de-impresoras-3d/
20. Luis Llamas (2018). Ingeniería, informática y diseño. Recuperado de https://www.luisllamas.es/
21. Larios Gómez M., Hernández Beristain A., Martínez Mirón E. (2017). Scheduling: A graphical interface for applying a process scheduling algorithm. Research in Computing Science. Vol. 145, pp. 119-125.
22. Authors,avrchip.com (2015). AVR Microcontroller and Arduino Tutorial. Recuperado de http://avrchip.com/arduino-nano-datasheet-and-tutorial/
23. Larios Gómez M. Migliolo Carrera J., Anzures García M., Aldama Díaz A. (2018). A scheduling algorithm for a platform in real time. CCIS 948 9th international supercomputing conference in Mexico ISUM 2018. Springer International Publishing AG.