Universidad Católica de Santa María

Facultad de Ciencias e Ingenierías Físicas y Formales

Escuela Profesional de Ingeniería Electrónica

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA ELECTRÓNICO DIDÁCTICO APLICADO A LA ENSEÑANZA Y APRENDIZAJE DE LA ELECTRÓNICA Y LA ROBÓTICA EDUCATIVA UTILIZANDO MICROCONTROLADORES

Tesis presentada por el Bachiller:

Valencia Rodriguez, Bruno Fernando

Para optar el Título Profesional de:

Ingeniero Electrónico con Especialidad en Telecomunicaciones

Asesor:

Ing. Zegarra Gago, Henry Christian

Arequipa-Perú

2021

DICTAMEN APROBATORIO DEL BORRADOR DE TESIS

UCSM-ERP

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

INGENIERIA ELECTRONICA

TITULACIÓN CON TESIS

DICTAMEN APROBACIÓN DE BORRADOR

Arequipa, 29 de Marzo del 2021

Dictamen: 002397-C-EPIE-2021

Visto el borrador del expediente 002397, presentado por:

2010200281 - VALENCIA RODRIGUEZ BRUNO FERNANDO

Titulado:

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA ELECTRÓNICO DIDÁCTICO APLICADO A LA ENSEÑANZA Y APRENDIZAJE DE LA ELECTRÓNICA Y LA ROBÓTICA EDUCATIVA UTILIZANDO MICROCONTROLADORES

Nuestro dictamen es:

APROBADO

1475 - MALAGA CHAVEZ CESAR EDUARDO
DICTAMINADOR

1843 - URRUTIA ESPINOZA MARIO WILLIAM
DICTAMINADOR

2465 - ZEGARRA GAGO HENRY CHRISTIAN
DICTAMINADOR

DEDICATORIA

A mis padres Fernando y Eleana por haberme forjado como la persona que soy en la actualidad; muchos de mis logros se los debo a ustedes entre los que se incluye este. Me formaron con reglas y con algunas libertades, pero al final de cuentas, me motivaron constantemente para alcanzar mis anhelos.

A mi hermano Andrés por su cariño y apoyo, durante todo este proceso, por estar conmigo en todo momento gracias.

A mi enamorada Daniela quien me apoyo y alentó para continuar, cuando parecía que me iba a rendir.

AGRADECIMIENTO

Me van a faltar páginas para agradecer a las personas que se han involucrado en la realización de este trabajo, sin embargo merecen reconocimiento especial mi madre y mi padre que con su esfuerzo y dedicación me ayudaron a culminar mi carrera universitaria y me dieron el apoyo suficiente para no decaer cuando todo parecía complicado e imposible.

Asimismo, agradezco infinitamente a mi hermano que con sus palabras me hacía sentir orgulloso de lo que soy y de lo que le puedo enseñar.

De igual forma, agradezco a mi asesor de tesis ing. Henry Zegarra, gracias a sus consejos y correcciones hoy puedo culminar este trabajo. A los ingenieros que me han visto crecer como persona, gracias a sus conocimientos hoy puedo sentirme dichoso y contento.

A mis amigos que gracias a su apoyo moral me permitieron permanecer con empeño, dedicación y cariño, y a todos quienes contribuyeron con un granito de arena para culminar con éxito este proyecto.

RESUMEN

La Robótica Educativa ha sido concebida, sistematizada curricularmente y modelada como un sistema educativo. Proponiendo el desarrollo de Talleres dentro de las horas de libre disponibilidad contempladas en el plan de estudios. Dichos talleres contribuyen al logro de determinados aprendizajes considerados prioritarios, o de especial importancia para las necesidades específicas de los estudiantes, mejorando los resultados de aprendizaje en las instituciones que se ha aplicado.

Lastimosamente no todas las instituciones tienen acceso a este tipo de educación o la capacitación suficiente para utilizar de forma correcta los módulos.

Es por ello que, en la presente tesis se propone diseñar un módulo de robótica educativa capaz de cumplir con los requerimientos necesarios para que el alumno pueda introducirse en el mundo de la electrónica y la robótica de una forma amigable, resultando en un interés hacia el estudio de una carrera de ciencias e ingeniería.

Por lo que, se construyó un módulo programable capaz de realizar distintas tareas, compatible con una gran variedad de sensores disponibles en el mercado. Basado en un lenguaje de programación de alto nivel de código abierto y con hardware personalizable gracias al uso de las tecnologías de fabricación aditiva,

El módulo fue empleado en estudiantes de nivel escolar secundario como parte de un curso de introducción a la robótica, consiguiendo los resultados esperados y despertando el interés por parte de los alumnos hacia las áreas de ciencia e ingeniería.

Palabras claves: Sistema electrónico didáctico, enseñanza y aprendizaje, robótica educativa, microcontroladores

ABSTRACT

Educational Robotics has been conceived, curricularly systematized and modeled as an educational system. Proposing the development of workshops within the hours of free availability contemplated in the study plan. These workshops contribute to the achievement of certain learning considered priorities, or of special importance for the specific needs of students, improving the learning results in the institutions that they have been applied.

Unfortunately, not all institutions have access to this type of education or sufficient training to use the modules correctly.

That is why, in this thesis, it is proposed to design an educational robotics module capable of meeting the necessary requirements so that the student can enter the world of electronics and robotics in a friendly way, resulting in an interest towards the study of a career in science and engineering.

Therefore, a programmable module capable of performing different tasks was built, compatible with a wide variety of sensors available on the market. Based on a high-level open source programming language and customizable hardware thanks to the use of additive manufacturing technologies,

The module was used in secondary school students as part of an introductory robotics course, achieving the expected results and awakening the interest of the students towards the areas of science and engineering.

Keywords: electronic didactic system, teaching and learning, educational robotics, microcontrollers

INTRODUCCION

El presente trabajo se enfoca en el diseño e implementación de un módulo educativo aplicado a la enseñanza y aprendizaje de la electrónica y robótica. El diseño abarca la selección de un lenguaje de programación adecuado a los requerimientos propuestos, un microcontrolador compatible con el lenguaje de programación y los respectivos sensores y actuadores necesarios para las aplicaciones asignadas. Posteriormente se procedió a diseñar la placa de circuito impreso en función a los parámetros establecidos anteriormente, realizando el cálculo teórico del consumo total del circuito, permitiendo la selección de una fuente de alimentación adecuada, en este caso una batería de polímero de litio.

Definidos todos los parámetros electrónicos, se procedió a diseñar el chasis del módulo, tomando en cuenta parámetros como resistencia a golpes o caídas, facilidad de ensamblaje y la implementación de un conector universal que permita añadir cualquier tipo de sensor, actuador o extensión al módulo. Respetando en todo momento las dimensiones propuestas de 10 x 10 centímetros, las cuales habilitan al módulo para la competencia en distintas categorías de robótica tanto profesional como escolar.

Finalmente se elaboraron “sketchs” o códigos de prueba, para brindar al alumno un punto de partida al momento de realizar la programación del módulo, sin interferir con su creatividad, permitiéndole plasmar sus ideas en instrucciones y mejorar el código según la aplicación o tarea asignada.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

CAPÍTULO I: PLANTEAMIENTO METODOLÓGICO

1.1. Identificación del problema

Los niños tienen curiosidad por entender las cosas, algunos desarman juguetes en busca de comprender su funcionamiento, sin embargo, conforme pasan los años esta atracción va desapareciendo y solo permanece en aquellos que aspiran a ser inventores.

La robótica educativa ahora se ha convertido en una herramienta de enseñanza a fin de que se potencien las inteligencias múltiples y permita a los alumnos indagar, analizar, aplicar y relacionar la ingeniería en cualquier situación de la vida.

En respuesta a esto las instituciones educativas privadas implementaron cursos de robótica en su currícula, por otro lado, instituciones educativas públicas recibieron módulos de robótica de la marca Lego WeDo brindados por el gobierno.

Incluso en las mejores condiciones las instituciones públicas y privadas no cuentan con una solución uniforme para esta necesidad, siendo casi imposible promover un enfoque de enseñanza que permita un mejor uso de los recursos tecnológicos.

1.2. Descripción del problema

Entre los alumnos existe un prejuicio acerca de los conceptos de ingeniería y la complejidad que implica diseñar y programar un prototipo robótico, esto provoca en los estudiantes desinterés hacia ese tipo de temas, por ello algunas instituciones educativas optaron por añadir a su currícula cursos de programación y de robótica para promover así el interés hacia temas de ingeniería.

Un gran número de instituciones educativas publicas cuentan con módulos de la marca Lego WeDo y laptops modelo XO-1 de la marca OLPC, una solución implementada por el gobierno en el año 2007 por el “Programa Una Laptop Por Niño”, herramienta pedagógica que introduce al alumno en conceptos relacionados a la robótica, física y lenguaje de programación, en la actualidad se han distribuido 83971 kits de robótica, 797352 laptops XO-1 y se han capacitado a 5144 docentes a nivel nacional.

En la mayoría de los casos, las instituciones no llegan a utilizar de forma correcta los módulos, además de las pocas horas asignadas a este tipo de actividades y sin la posibilidad de llevar el módulo a su hogar para continuar potenciando su creatividad.

Instituciones educativas privadas se están preocupando más por el desarrollo del tema de la robótica en sus aulas, han comenzado en los últimos años a impartir programas de aprendizaje haciendo uso de la robótica educativa, incluso en la educación primaria. Además, muchas compañías de juguetes y robótica se han sumado a la tendencia y han diseñado sus productos para hacer que los cursos relacionados con la robótica educativa sean más simples, interactivos, atractivos y emocionantes a nivel escolar.

En el mercado existen módulos de introducción a la programación y la robótica de marcas conocidas como Lego, Makeblock o Complubot, pudiendo ser una solución a implantar en la educación peruana pero son de difícil acceso, existen distribuidores de las marcas anteriormente señaladas pero no son oficiales y en la mayoría de los casos es necesario importar los módulos, adicionalmente el material didáctico que ofrecen también es limitado, no cuentan con un servicio post venta capaz de solucionar algún problema y se trabaja con pocos sensores y actuadores, en ocasiones es necesario comprar paquetes adicionales para poder realizar este tipo de mejoras.

Si la situación actual continua, el desarrollo de la robótica educativa en el Perú se verá estancado a diferencia de otros países que si ponen énfasis en fomentarla de una forma más dinámica.

1.3. Alcances de la investigación

Al finalizar este proyecto se busca implementar un módulo que permita desarrollar habilidades de electrónica y robótica en los estudiantes tales como:

- Permitir que los estudiantes se involucren en sus propios procesos de aprendizaje
- Fomentar el desarrollo del pensamiento lógico, intuición científica y creatividad

- Desarrollar habilidades para la resolución de problemas y para la investigación
- Alimentar su evolución como autodidactas
- Fomentar y estimular habilidades que serán de gran importancia en su futuro profesional como son el razonamiento analítico, el razonamiento lógico o el pensamiento crítico.
- Estimular el interés por las ciencias tecnológicas, uno de los campos de mayor futuro profesional.

Permitiendo así un desarrollo equitativo de la robótica en la educación peruana, que permita potenciar habilidades de los estudiantes, así como despertar aptitudes para la ingeniería.

1.4. Objetivos

1.4.1. Objetivo principal

Diseñar e implementar un sistema electrónico didáctico para promover el desarrollo de la electrónica, robótica y programación en estudiantes de educación básica regular utilizando una interfaz amigable, flexible y programable.

1.4.2. Objetivos específicos

- Analizar comparativamente acerca de las alternativas de módulos electrónicos educativos presentes en el mercado, teniendo como base soluciones ya implementadas en nuestro país.
- Diseñar las partes y etapas del módulo, tomando en consideración el tipo de sensores, actuadores y microcontroladores necesarios para disminuir el costo del producto final.
- Implementar hardware y software en función al módulo que se planea poner en funcionamiento, tomando en cuenta factores como el consumo de energía, el tipo de sensores, la compatibilidad del lenguaje de programación y la escalabilidad y modularidad del mismo.

- Elaborar sesiones o guías de aprendizaje progresivas que permitan al estudiante utilizar correctamente los módulos a nivel de soporte físico y lógico desde conceptos básicos hasta temas más avanzados.

1.5. Estado del arte

En los últimos años, la robótica educativa ha tomado fuerza en el Perú, poco a poco se van creando más academias de enseñanza que usan módulos fáciles de ensamblar y programar para jóvenes.

En el año 2016 el Ministerio de Educación (MINEDU) ha puesto en marcha el programa de Robótica Educativa a través de PerúEduca, la cual es una plataforma digital del MINEDU que brinda servicios a la comunidad educativa [1]. Cerca de 43000 kits de robótica educativa se entregarán en beneficio de 20000 instituciones educativas en zonas rurales y urbanas del país, más de 2.5 millones de estudiantes de primaria mejorarán sus aprendizajes en ciencia, matemática y comunicación para aplicarlos a situaciones reales. Estos kits permitirán a los niños construir modelos robóticos conectados a sus computadoras capaces de realizar las tareas que ellos programen de manera fácil y entretenida [2]. La Robótica Educativa de PerúEduca es un medio de aprendizaje multidisciplinario que se basa en la creación de modelos robóticos que permiten desarrollar habilidades en las otras áreas de aprendizaje, fortificando el pensamiento creativo y la resolución de los posibles problemas que puedan surgir [3].

El Diseño Curricular Nacional, propone el desarrollo de Talleres dentro de las horas de libre disponibilidad contempladas en el Plan de estudios. Dichos talleres contribuyen al logro de determinados aprendizajes considerados prioritarios, o de especial importancia para las necesidades específicas de los estudiantes. En los talleres se desarrolla principalmente la competencia de un área curricular. Además, se moviliza competencias y capacidades correspondientes a otras áreas curriculares [4].

El programa de robótica educativa de PerúEduca estaba pensado originalmente para realizarse con los kits WeDo de la marca LEGO. Actualmente ya no se encuentran en el mercado, en su reemplazo están los kits LEGO WeDo 2.0. Este kit se basa en los últimos estándares científicos y se creó para mejorar la curiosidad y las habilidades científicas de los estudiantes. Viene en un contenedor

de almacenamiento junto con bandejas de clasificación, etiquetas, un Smarthub, un motor mediano, un sensor de movimiento, un sensor de inclinación y suficientes elementos de construcción para dos estudiantes. El software compatible con computadoras de escritorio y tabletas proporciona un entorno de programación fácil de usar e incluye el paquete curricular WeDo 2.0, que cubre las ciencias de la vida, físicas, terrestres y espaciales, así como la ingeniería [5].

Otra marca que ofrece módulos similares es la marca Makeblock. El mBot es un robot educativo STEAM para principiantes, que hace que la programación de robots de enseñanza y aprendizaje sea simple y divertida. Con solo un destornillador, las instrucciones paso a paso y un programa de estudio, los niños pueden construir un robot desde cero y experimentar los placeres de la creación práctica. mBot tiene tres modos de control integrados, incluido el modo de evitación de obstáculos, el modo de seguimiento de línea y el modo de control manual. Tiene 4 puertos de expansión y puede conectarse a más de 100 tipos de módulos electrónicos. Junto con la aplicación mBlock Blockly, programe su mBot para desbloquear la habilidad de codificación simplemente arrastrando y soltando los bloques de comando [6]. De la misma marca se pueden encontrar kits más avanzados, pero usando de base el mBot. El kit STEAM Education -Science Robot es un kit que incluye un mBot, una variedad de sensores, módulos electrónicos y componentes mecánicos, y viene con 16 lecciones complementarias [7].

1.6. Antecedentes

A continuación, presentare investigaciones realizadas sobre módulos o kits de robótica educativa, estas servirán como antecedentes para la elaboración de la presente tesis, cada una de las investigaciones fue revisada y evaluada para el desarrollo del trabajo.

Título: Desarrollo de un robot móvil y una plataforma de interacción tangible para la inducción a la programación en ciencias de la computación de niños en edad preescolar.
Autor: Cárdenas Cáceres, Pablo
Año: 2018.
Tipo: Tesis para grado de magister.

Correlación:

Esta tesis realiza una comparación teórica de las investigaciones realizadas en el campo de programación tangible y propone la elaboración de una plataforma mecatrónica económica en comparación a alternativas ya existentes. La plataforma está conformada por un tablero de programación (que emplea la electrónica más básica), y un robot móvil (que utiliza algoritmos de control avanzado para garantizar la exactitud de sus movimientos). Finalmente se comprueba el funcionamiento del prototipo mecatrónico mediante la comparación de datos obtenidos en el modelo simulado y el real.

Título: Desarrollo de un módulo electrónico para la enseñanza del área de ciencia, tecnología y ambiente en la educación secundaria peruana.
Autor: SUAZO, José Antonio
Año: 2015.
Tipo: Tesis para título Ingeniero Electrónico.
Correlación:

Para la elaboración del módulo electrónico de esta tesis se consideraron tres etapas: primero, una introducción a la electrónica; segundo, la elaboración de un módulo electrónico general; y por último la implementación de tres proyectos aplicativos utilizando el módulo. Los resultados obtenidos se basaron en el desarrollo de un taller de electrónica a 12 alumnos de educación secundaria del IEP San Martín de Porres, y se comprobó que el rendimiento educativo en CTA de los alumnos dependía de su desempeño en el taller de electrónica, donde diseñaron e implementaron un módulo de electrónica.

Título: Implementación de kits de robótica para mejorar la comprensión científica y tecnológica de los alumnos de educación primaria en la provincia de Coronel Portillo – Ucayali.
Autor: Herrera Salazar Jose Luis, Espinoza Duran Fernando, Vallejo Aguilar Luz Maribel
Año: 2018.
Tipo: Tesis para título Ingeniero Sistemas.

Correlación:

Esta tesis plantea implementar un kit de robótica para mejorar las capacidades de comprensión científica y tecnológica de los alumnos de educación primaria en la provincia de Coronel Portillo – Ucayali. Se encarga de determinar en qué medida la implementación de kits de robótica mejoraría los contenidos de la capacidad conceptual, capacidad procedimental y capacidad actitudinal de la comprensión científica y tecnológica en los estudiantes de educación primaria.

Título: Desarrollo e implementación de un módulo robótico de bajo coste para acercar la robótica educativa y la investigación multiagente a usuarios con pocos recursos económicos.
Autor: Simó Villanueva, Marcos
Año: 2020.
Tipo: Tesis de máster
Correlación:

Esta tesis desarrolla una unidad robótica móvil con amplia conectividad y de bajo costo. El módulo robótico emplea impresión 3D y utiliza componentes electrónicos de bajo costo como un microcontrolador ESP32. Este tipo de microcontrolador cuenta con conectividad WIFI y permite transmitir video en tiempo real. El robot permite ser controlado y programado mediante una conexión de red local con un servidor, también cuenta con algoritmos básicos de navegación autónoma. El propósito principal de esta tesis es que el costo sea lo más reducido y accesible posible para poder acercar un robot con una alta conectividad y manejabilidad a la docencia e investigación en la robótica si se disponen de pocos recursos económicos.

Título: Diseño e Implementación de un Módulo Didáctico para el Aprendizaje en la Construcción, Implementación y Manipulación de Robots.
Autor: Lancheros, Diana
Año: 2010.
Tipo: Artículo publicado en la revista Formación Universitaria Vol. 3 N° 5.
Correlación:

Este artículo consta de un proyecto desarrollado como herramienta pedagógica para permitir afianzar el aprendizaje en la construcción,

implementación y manipulación de robots utilizando piezas Lego®. Fue necesario realizar una investigación preliminar de la literatura para determinar la situación actual a nivel nacional e internacional, luego se procedió al diseño del módulo, comenzando por el software en conjunto con las partes electrónicas y el diseño. En la fase final se evalúa el nivel de interpretación y capacidad de razonamiento espacial obtenido en los estudiantes que utilizaron el módulo.

Como se puede apreciar todos los trabajos de investigación se enfocan en utilizar la robótica educativa como una herramienta para potenciar alguna habilidad en los estudiantes, otro punto en común es que buscan que el precio del módulo no sea elevado utilizando componentes de bajo costo e impresión 3D y algunos emplean comunicaciones inalámbricas para controlar el módulo.

Todas las características anteriormente descritas serán tomadas en cuenta para la elaboración del módulo de robótica educativa y adicionalmente se añadirán características extra como modularidad, resistencia y algo que ninguno de los trabajos anteriores tomó en consideración, que son las competencias de robótica tanto nacionales como internacionales, la mayoría de estas ya cuentan con estándares para poder participar, el más importante quizás relacionado a las dimensiones del robot para algunas categorías (10x10x10cm), es por eso que para la elaboración de nuestro modulo se consideraran esas dimensiones como una regla importante del diseño.

CAPÍTULO II: MARCO TEÓRICO

2.1. Robótica

La terminología “Robot” se aprecia la primera vez en 1921, en la obra teatral R.U.R. (Rossum´s Universal Robots) del novelista y autor dramático checo Karel Capek en cuyo idioma la palabra “robota” tiene como significado fuerza de trabajo o servidumbre. En el término robot confluyen las imágenes de máquinas para la realización de trabajos productivos y de copia de movimientos y comportamientos de seres vivos [8].

La robótica es una ciencia o rama de la tecnología, la cual se encarga de estudiar el diseño y construcción de máquinas capaces de desempeñar tareas realizadas por el ser humano o que requieran del uso de inteligencia. De forma general, la robótica se define como: El conjunto de conocimientos teóricos y prácticos que permiten concebir, realizar y automatizar sistemas basados en estructuras mecánicas poli articuladas, dotados de un determinado grado de “inteligencia” y destinados a la producción o sustitución de la mano del hombre en diversas tareas [9].

2.2. Robótica educativa

La robótica educativa también conocida como robótica pedagógica es una disciplina que tiene por objetivo la concepción, creación y puesta en funcionamiento de prototipos robóticos y programas especializados con fines pedagógicos. La robótica educacional crea las mejoras en las condiciones de apropiación de sabiduría que permite a los estudiantes fabricar sus propias representaciones de los fenómenos del mundo que los rodea, facilitando la adquisición de conocimientos en el tema acerca de estos fenómenos y su transferencia a diferentes áreas del conocimiento [10].

La robótica educativa busca iniciar el interés de los estudiantes transformando las asignaturas tradicionales (Matemáticas, Física, Informática) en más atractivas e integradoras, al crear entornos de aprendizaje propicios que recreen los problemas del lugar que los rodea [11].

Unas de las características especiales que tiene la robótica educativa es la capacidad de mantener la atención del estudiante. El hecho de que el estudiante pueda usar y experimentar con estas herramientas de aprendizaje basadas en robótica hace que el estudiante pueda centrar sus percepciones y observaciones en la actividad que está haciendo [10].

2.3. Beneficios de la robótica educativa

La robótica educativa es significativa para ayudar en las habilidades productivas, creativas, digitales y comunicativas; y se convierte en un motor para el descubrimiento cuando produce cambios en las personas, en las ideas y actitudes, en las relaciones, modos de actuar y pensar de los estudiantes y profesores. [11].

Al encontrarse la robótica educativa inmersa en la idea constructivista la lección, recurre a la significancia de los aprendizajes. En este argumento, constituye una lección globalizada, de carácter que la nueva lección está relacionada de modo sustantivo con aprendizajes previos y, además, puede ser relacionado con el provecho de aprendizajes posteriores. Cuando los conocimientos de un método son utilizados para solucionar problemas de una disciplina distinta, se habla de transferencia [12].

Los ambientes de amaestramiento permiten impulsar procesos cognitivos y sociales que propician un aprendizaje característico en el estudiante y las destrezas necesarias para desempeñarse adecuadamente en el argumento diverso y difícil que requiere la humanidad. Estos espacios son generados gracias a las relaciones e interacciones que ocurren en la sala de clase entre los estudiantes y docentes, y entre ellos con los medios con los que se tiene [10].

Uno de los objetivos de hacerse servir la robótica en las aulas es afianzar a los estudiantes en las ciencias y la tecnología. Siguiendo el paradigma constructivista/construccionista y el amaestramiento a través del juego se puede asistir a la edificación de nuevos conocimientos. Por otra parte, las competiciones con robots son muy populares, ya que un desafío ofrece motivación extrínseca agregada para los estudiantes, aumenta sus habilidades de trabajo en equipo y anima al estudiante a nivelar y evaluar una diversidad de opiniones [11].

Gracias a que la robótica educativa se caracteriza por ser configurable y multidisciplinaria, otorga a los estudiantes la oportunidad de elaborar, amplificar y experimentar con diferentes robots que permitirán solucionar algunos problemas de la existencia diaria y al igual tiempo facilitara el amaestramiento. En otras palabras, se trata de plasmar las condiciones para la edificación de conocimientos y acceder así la transmisión de estos en diferentes campos.

2.4. Robots móviles

La robótica móvil es considerada actualmente como un “área de la tecnología avanzada manejadora de problemas de alta complejidad” [13]. Los robots móviles son dispositivos de transportación automáticos, esto quiere decir que son capaces de desplazarse a través de cierto ambiente de trabajo con un nivel de autonomía, portando cargas en una plataforma mecánica acompañada de un sistema de locomoción. Sus aplicaciones pueden ser muy variadas y siempre están relacionadas con tareas que normalmente son riesgosas o nocivas para la salud humana [14], en las áreas de inspección, programación, inteligencia artificial, percepción e instrumentación, y sirven de cimiento para el impulso en diversos campos de la actividad, aportando soluciones tecnológicas innovadoras orientadas al progreso de mejores robots y al incremento del abanico de aplicaciones disponibles [13].

Son los mismos robots móviles los que determinan el trabajo que realizaran en base a algunas características particulares. el trabajo determinará tanto particularidades estructurales del robot (tipo de rueda, forma física del robot sistema de tracción y dirección, etc.), como características sensoriales. En general los robots móviles distribuyen sus sistemas de tracción y dirección sobre los ejes de sus ruedas de acuerdo con las exigencias de la agilidad, maniobrabilidad y características del terreno. La exactitud y prisa con que el robot móvil debe lograr su ocupación implica tener un sistema de tracción confiable y un sistema de orientación que dé maniobrabilidad al robot. Esta confiabilidad y maniobrabilidad que debe tener el robot móvil, determinan las características del sistema de tracción y orientación, no únicamente en lo que respecta a la técnica, sino asimismo a la cantidad de ruedas necesarias y al tipo y habilidad de éstas para lograr una estructura mecánica estable [14].

2.5. Cinemática y dinámica del robot

Con la finalidad de lograr ejecutar estudios teóricos referente a un robot móvil, se requiere disponer de un modelo matemático total y absoluto [15]. Un único robot móvil puede poseer varios modelos matemáticos que lo representan de distintas maneras. Cada una de estas ecuaciones es utilizada con distintos propósitos, dependiendo de qué propiedades del robot se desean analizar o que comportamientos alcanzar.

Los robots móviles se pueden montar en base a distintos diseños de plataformas basadas en los diversos sistemas de tracción que pueden utilizar. Las plataformas más comunes utilizan el sistema de tracción diferencial (differential steering), este sistema utiliza motores independientes para cada una de las ruedas, pero situados sobre el mismo eje, además utiliza ruedas locas o puntos de sustento para suministrar estabilidad a la plataforma.

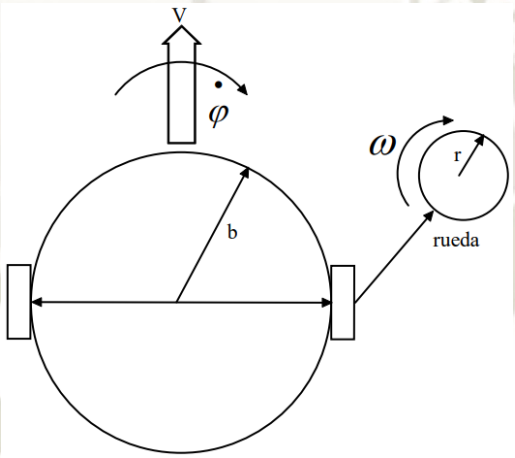


Figura 1. Parámetros físicos y variables dinámicas de un robot móvil de dos ruedas [15]

En La Figura 1 se puede ver tanto los parámetros físicos del robot, como las variables dinámicas. Los parámetros físicos del robot son: la longitud del eje (o distancia entre las ruedas) denominada 2b; el radio de cada rueda r1 y r2; la masa del cuerpo del robot, M; y la masa de cada una de las ruedas, m. Las variables relacionadas con el movimiento del robot son: la velocidad lineal que posee el cuerpo del robot, V; la velocidad angular que posee el cuerpo, ϕ; y las velocidades angulares de las ruedas, ω1 =θ1 para la rueda derecha y ω2 =θ2 para la rueda izquierda. Además, entre las variables dinámicas se encuentra asimismo la posición absoluta del robot en el espacio. Esta posición queda definida por las

coordenadas bidimensionales del centro de masa (x, y) y el ángulo entre la dirección de movimiento del robot y el eje x, denominado ϕ, tal como se observa en la Figura 2.

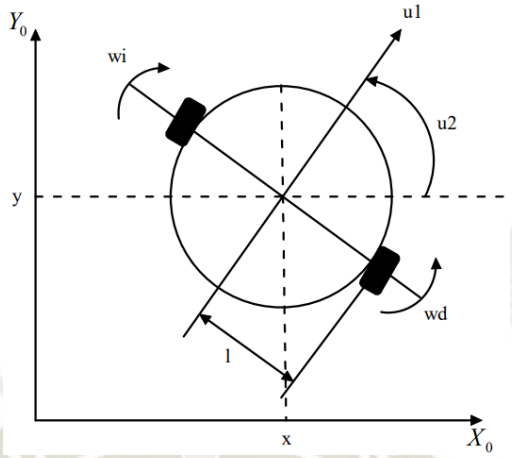


Figura 2. Localización en el plano cartesiano del robot [15].

El modelo también consta de parámetros eléctricos, asociados a los motores DC que se utilizan para entregar la tracción a las ruedas del robot [15].

2.6. Microcontrolador

Un microcontrolador es un instrumento electrónico competente de llevar a cabo procesos lógicos. Estos procesos o acciones son programados en lenguaje ensamblador por el usuario, y son introducidos en este a través de un programador [16]. Los microcontroladores son básicamente microprocesadores completos rodeados de ciertos bloques periféricos básicos [17]. La palabra microcontrolador proviene de la unión de las palabras *micro* porque es pequeño y *controladores* porque controla máquinas o incluso otros controladores [18].

2.6.1. Microcontrolador PIC

Los PIC (*Peripheral Interface Controller*) son una familia de microcontroladores de bajo precio, reducido consumo, tamaño pequeño, gran calidad, fiabilidad y abundancia de información [19].

Son una familia de microcontroladores tipo RISC (*Reduce Instruction Set Computing*) fabricados por Microchip Technology Inc. y derivados del PIC1650,

originalmente desarrollado por la división de microelectrónica de General Instrument.

Hoy en día gran cantidad de PICs vienen con varios periféricos incluidos (módulos de comunicación serie, UARTs, núcleos de control de motores, etc.) y con memoria de programa desde 512 a 32.000 palabras (una palabra corresponde a una instrucción en lenguaje ensamblador, y puede ser de 12, 14, 16 ó 32 bits, dependiendo de la familia específica de PICmicro)[20].

Cuentan con infinidad de herramientas para programarlos, desde el ensamblador en un archivo txt, pasando por BASIC y hasta lenguajes de alto nivel como C [21].

2.6.2. Microcontrolador Arduino

Arduino es una plataforma de creación de código abierto, basada en una sencilla placa de circuito impreso que contiene un microcontrolador de la marca “ATMEL”, con entradas y salidas analógicas y digitales [22]. Esta plataforma está basada en hardware y software libre, flexible y fácil de utilizar para los creadores y desarrolladores.

Primeramente, debemos distinguir los conceptos de hardware libre y el software libre. El hardware libre son los dispositivos cuyas especificaciones y diagramas son de acceso público, de modo que cualquiera puede replicarlos. Arduino ofrece las bases para que cualquier otra persona o entidad pueda elaborar sus propias placas, pudiendo ser diferentes entre ellas, pero equivalentemente funcionales a partir de la misma base. El software libre son los programas informáticos cuyo código es accesible por cualquiera que desee utilizarlo y modificarlo. Arduino ofrece la plataforma Arduino IDE (Entorno de Desarrollo Integrado), que es un entorno de programación con el que cualquiera puede crear aplicaciones para las placas Arduino, de manera que se les puede dar todo tipo de utilidades [23].

2.6.3. Microcontrolador Raspberry Pi

Raspberry PI es una computadora de tamaño muy pequeño, desarrollado por la Fundación Raspberry PI en el Reino Unido en 2011, con un precio muy bajo y con el objetivo de promover la enseñanza de la Informática [24].

Este computador puede usarse en proyectos de electrónica y para tareas básicas que haría cualquier computador de sobremesa como navegar por internet, hojas de cálculo, procesador de textos, ver vídeo en alta definición e inclusive jugar a ciertos juegos.

La Raspberry Pi es la placa de una computadora ordinaria compuesta por un SoC, CPU, memoria RAM, puertos de entrada y salida de audio y vídeo, conectividad de red, ranura SD para almacenamiento, reloj, una toma para la alimentación, conexiones para periféricos de bajo nivel, etc. Prácticamente igual que encontramos en la parte de atrás de una computadora, porque la Raspberry es una computadora [25].

2.7. Lenguaje de programación

Un lenguaje de programación es un acumulado de símbolos y reglas sintácticas y semánticas que definen su organización y el sentido de sus elementos y expresiones, y es utilizado para controlar la conducta física y lógica de una máquina [26].

Mediante este lenguaje se comunican el programador y la máquina, permitiendo especificar, de una forma muy precisa, aspectos como:

- Cuáles datos debe operar un software específico;
- Cómo deben ser almacenados o transferidos esos datos;
- Las acciones que debe tomar el software dependiendo de los problemas variables [27].

2.7.1. Lenguaje ensamblador

El lenguaje ensamblador o assembler es un lenguaje de programación de nivel bajo simbólico que se ha determinado para que se puedan escribir programas con una sintaxis próxima al lenguaje de máquina, pero sin tener que escribir el

código en binario, sino utilizando una serie de mnemónicos más fáciles de hallar para el programador [28]. El lenguaje ensamblador mantiene una correlación 1 a 1 respecto al lenguaje máquina, es decir que una instrucción en ensamblador representa una instrucción en código máquina [29].

Normalmente, los programas escritos en ensamblador requieren muy poco espacio de memoria y se ejecutan mucho más rápido que si se hubiesen elaborado en un lenguaje de alto nivel, puesto que están optimizados para una construcción específica [30] Sin embargo, esto último es un problema, pues hace que los programas no sean portables de un microprocesador a otro con un microprocesador distinto. Escribir un programa en lenguaje ensamblador requiere de conocimientos acerca del hardware de la computadora, su agregado de instrucciones y sus reglas de uso [31].

2.7.2. Lenguaje C++

C++ fue desarrollado a comienzos de la década de los años ochenta (80), cuando Bjarne Stroustrup utilizó sus conocimientos en lenguaje de simulación para elaborar un lenguaje de programación dirigido a objetos [32].

Este lenguaje abarca tres paradigmas de la programación:

1. Programación Estructurada
2. Programación Genérica
3. Programación Orientada a Objetos [33]

Al día de hoy el C++ es un lenguaje versátil, eficaz y general. El C++ mantiene las ventajas del C en cuanto a cantidad de operadores y expresiones, variabilidad, brevedad y eficacia. Conjuntamente, ha eliminado algunas de las dificultades y limitaciones del C clásico. La evolución de C++ ha continuado con la aparición de Java, un lenguaje elaborado simplificando algunas cosas de C++ y añadiendo otras, que se utiliza para ejecutar aplicaciones en Internet [34].

C++ puede utilizarse a partir de programas interactivos simples e inclusive programas de ingeniería y científicos sofisticados y complejos, adentro del contexto de una estructura en verdad orientada a objetos [32].

2.7.3. Lenguaje Python

Python es un lenguaje de programación de alto nivel que se caracteriza por el hecho de ser un lenguaje sencillo, cómodo de leer, escribir y depurar, y también es portable [35]. Python es un lenguaje multiparadigma en el que conviven de modo nativo aspectos imperativos, funcionales y orientados a objetos. Estos paradigmas están muy bien desacoplados, lo que permite que la entrada al lenguaje se pueda hacer de manera progresiva [36].

Una de las características importantes de Python, y posiblemente la mayor, es la librería estándar con que cuenta. Con decenas de módulos cubre la mayoría de las necesidades básicas de un programador y mucho más [37].

Python usa tipado dinámico y conteo de referencias para la gestión de memoria.

Una característica significativa de Python es la resolución dinámica de nombres; es decir, lo que enlaza un método y un nombre de variable mientras la ejecución del programa (asimismo llamado enlace dinámico de métodos).

Otro objetivo del diseño del lenguaje es la desenvoltura de extensión. Se pueden escribir nuevos módulos cómodamente en C o C++. Python puede incluirse en aplicaciones que necesitan una interfaz programable [38].

A Python se lo considera un lenguaje interpretado, ya que sus programas son ejecutados por un intérprete. Existen dos maneras de usar el intérprete: forma de comando y forma de guion. En forma de comando se escriben sentencias en el lenguaje Python y el intérprete enseña el resultado.

Python proporciona, incluso, varias características de los lenguajes de programación funcionales y puede usarse para integrar conceptos que se pueden examinar con más determinación en cursos con Scheme y Lisp [39].

2.8. Sensores

Un sensor es un dispositivo electrónico que, a partir de la energía del medio donde se mide, brinda una señal de salida transducible que es función de la variable medida [40]. El sensor es un dispositivo de entrada, debido a que será un mediador entre la variable física y el sistema de medida [41].

Los sensores imitan la capacidad de captación de los seres humanos, por ello es cada vez más habitual encontrarlos incorporados a cualquier área tecnológica. Debido a esta característica de emular la impresión humana, podemos hallar sensores conectados con los diferentes sentidos: vision, oído, tacto, es decir, que reaccionan a la luz, el sonido, contacto, etc. De igual modo que nuestro cerebro reacciona a la información que recibe de nuestros sentidos, los dispositivos que cuentan con sensores reaccionarán a la información que reciben de ellos. Los sensores nos permiten interactuar con el ambiente de forma que nos brindan información de algunas variables que nos rodean para lograr procesarlas y así crear órdenes o activar procesos.

El realizar una mirada de nuestro entorno nos revelara que se han transformado en algo diario y que los encontramos en innumerables aparatos domésticos: mandos a distancia, sistemas de alarma y protección, electrodomésticos, domótica, etc. De igual modo están presentes en vehículos, telefonía celular, el campo de la medicina y por supuesto en la robótica.

La incorporación de la sensorica a los sistemas electrónicos los ha dotado de cierta “inteligencia” artificial, ya que a través de los datos que recopilan, y una vez procesados convenientemente, permiten tomar con exactitud y rapidez las decisiones más adecuadas dentro del cometido para el que están diseñados dichos sistemas electrónicos [42].

2.8.1. Clasificación según el tipo de variable medida

Esta distribución suele ser la más frecuente; sin embargo, tiene el inconveniente de incitar cierto desorden, ya que un sensor igual puede ser utilizado para la medición de distintas variables físicas; por ejemplo, un sensor ultrasónico resulta muy útil si se desea medir cercanía, el nivel de un líquido, la presencia de un objeto, la velocidad de un fluido, etcétera. No obstante, su principio de funcionamiento siempre será el mismo, y solo depende del tipo de configuración en que se coloque y cómo se interprete la señal de salida del mismo [41].

Tabla 1. Clasificación de los sensores según la variable física a medir [41].

Clasificación de los sensores según la variable física a medir	De posición, velocidad y aceleración
	De nivel y proximidad
	De humedad y temperatura
	De fuerza y deformación
	De flujo y presión
	De color, luz y visión
	De gas y pH
	Biométricos
	De corriente

Adicionalmente es necesario señalar las características que diferencian un sensor de otro, como son los tipos de señales que nos proporcionan:

- **Analógicos:** Determina la información mediante una señal analógica (tensión, corriente), es decir, que pueden adquirir inmensidad de valores entre un mínimo y un máximo [43]. La salida oscila, a nivel macroscópico, de forma continua
- **Digitales:** Determina la información mediante una señal digital que puede ser un “0” o un “1” lógico, bien una secuencia de bits [42].

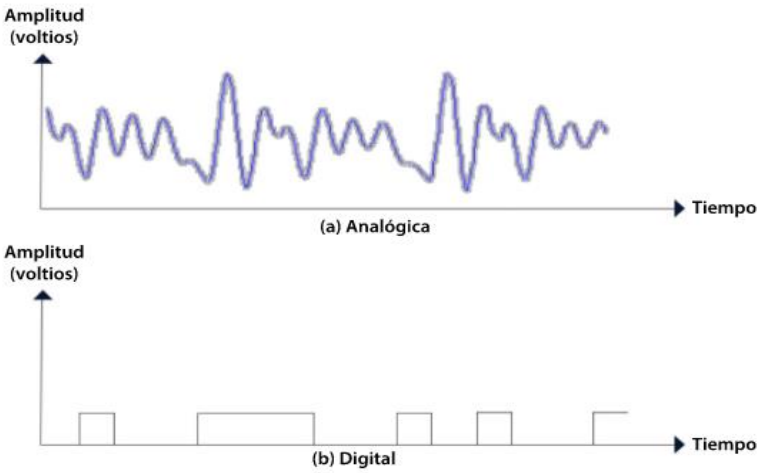


Figura 3. Tipos de señales de salida de los sensores [42].

2.8.2. Sensor de nivel y proximidad

Los sensores de cercanía o nivel son muy usados en aplicaciones como embotellado, sistemas de inspección para monitoreo de envasado, localización de obstáculos en sistemas inteligentes. Estos sensores muchas veces son confundidos con los sensores de presencia. No obstante, este tipo de sensores se limita a medir la cercanía de un objeto con respecto al sensor, sin interesar su colocación o establecer si el objeto está cerca del sensor para ser detectado, además de establecer el nivel de un contenedor en determinado porcentaje. Al igual que muchos sensores, estos pueden clasificarse de acuerdo al modo de funcionamiento[41].

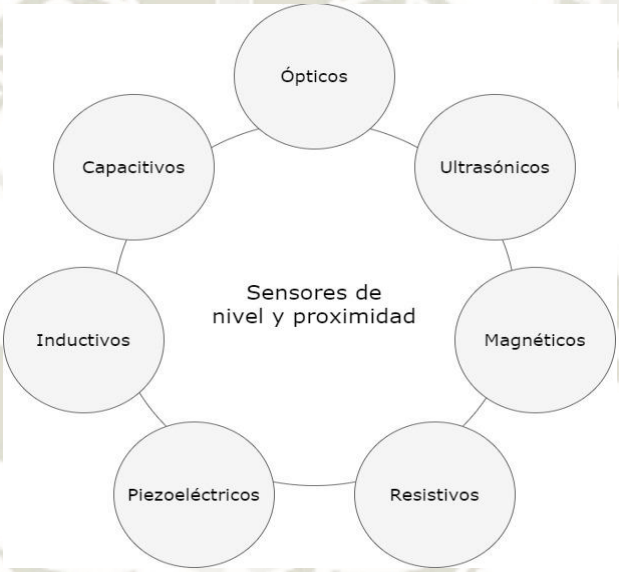


Figura 4. Clasificación de los sensores de nivel y proximidad [41].

2.8.3. Sensor ultrasónico

Los sensores de ultrasonido utilizan ondas ultrasónicas que tienen la habilidad que cuando viajan por un medio cualquiera son reflejadas si encuentran en su camino una interrupción o algún elemento extraño [44].

Las ventajas más importantes de este tipo de sensor son que, al ejecutar una lectura no invasiva, es decir, que no requiere contacto alguno para ejecutar la medida, la diversidad de objetos capaces de medir es muy amplia. Debido a la naturaleza de la señal ultrasónica, es viable ejecutar mediciones en superficies variadas, líquidos y en ambientes hostiles. La señal ultrasónica se puede crear

mediante diferentes técnicas, como electromagnéticas, ópticas, capacitivas y piezoeléctricas; de todas, esta última es una de las más utilizadas debido a su elevada certeza en comparación con las anteriores [41].

Este tipo de sensor está constituido por un emisor y un receptor. El emisor es el encargado de expresar una onda de ultrasonido (normalmente de frecuencias mayores de 20 KHz que se encuentra por encima del umbral auditivo del oído humano, 16KHz), cuya señal se suele representar de forma cónica [44].

Un sensor ultrasónico se ayuda del efecto Doppler, ya que actualmente el emisor genera una onda ultrasónica, la cual es absorbida en porción y reflejada en porción por el objeto a medir; así, a través del cálculo de la atenuación de la onda captada por el receptor, el tiempo que le toma a esta ser captada por el receptor, o por la existencia o inexistencia de dicha frecuencia en el emisor, es viable lograr características de la variable física que se desea establecer. El efecto Doppler se basa en un cambio aparente de frecuencia de la onda sonora respecto al emisor cuando esta es reflejada en un objeto móvil (o partículas inmersas en un fluido). Este cambio de frecuencia resulta proporcional a la velocidad relativa del emisor reflector. El cambio de frecuencia se puede computar mediante la ecuación:

$$f - 2f_e \frac{V}{V_S} cos\theta = f_e - f_r$$

donde:

f_e: frecuencia emitida

f_r : frecuencia recibida

θ: ángulo entre la velocidad y la dirección de propagación

V_S: velocidad del sonido

V: velocidad del objeto o fluido

f: diferencia de frecuencias

Si lo que se quiere es conocer el tiempo que demora la onda desde que es emitida hasta que es recepcionada, se debe usar la siguiente ecuación:

$$t = \frac{(d/sen\theta)}{V_s + V\ cos\theta}$$

donde:

t: tiempo [s]

d: distancia [m]

El núcleo de un sensor ultrasónico es un material piezoeléctrico; recordar que la piezoelectricidad es la propiedad que presenta un material de crear un voltaje debido a una fuerza aplicada. Los materiales piezoeléctricos son considerados un subconjunto de los materiales ferroeléctricos, cuya primordial característica es que presentan una polarización eléctrica finita, incluso sin ninguna fuerza aplicada. La onda ultrasónica que emite el material piezoeléctrico se genera por medio de una excitación eléctrica al material (el efecto piezoeléctrico asimismo se presenta de modo inverso; es decir, al emplear un voltaje el material experimenta una deformación, y como consecuencia este emite una onda mecánica). Esta onda es emitida por todo el material, lo que significa que es impuntual. Por consiguiente, los sensores ultrasónicos cuyo elemento piezoeléctrico es de forma redonda se conocen como transductores de fuente pistón, debido a la forma que presenta el campo sonoro que genera, entre más claro sea el color del campo sonoro, este es más potente. El principio de funcionamiento de estos sensores consiste en la generación de una onda de manera cíclica, la cual es de alta frecuencia y corta permanencia, conjuntamente de que se propaga en el medio. Al encontrar un objeto a su paso, esta es reflejada y vuelve en forma de eco al receptor. El circuito de acondicionamiento tiene la tarea de determinar el periodo transcurrido entre la emisión de la señal acústica y la recepción del eco. Gracias a la siguiente ecuación es posible determinar el tiempo que tarda el receptor en captar la señal reflejada en el objeto a medir [44].

$$d = \frac{1}{2} V_s t$$

donde:

d: distancia del emisor-receptor al objeto [m]

V_s: velocidad del sonido

t: tiempo transcurrido [s]

2.8.3.1. Devantec SRF04

El sensor SRF04 funciona generando impulsos de ultrasonidos inaudibles para el oído humano. Los impulsos generados viajan a la velocidad del sonido hasta chocar contra un objeto, entonces el sonido es reflejado y captado de nuevo por el receptor de ultrasonidos. Lo que hace el controlador incorporado es generar una

ráfaga de impulsos y seguido empieza a calcular el tiempo que tarda en alcanzar el eco. Este tiempo se refleja en un pulso de eco de amplitud igual a la distancia a la que se encuentra el objeto.

Visto desde un punto practico, lo que se debe hacer es enviar una señal de arranque en el pin 3 del SRF04 y posteriormente leer la amplitud del impulso que nos brinda en el pin 2. El pulso de disparo debe contar con una amplitud mínima de 10uS. Posteriormente leemos el pulso de salida de Eco y medimos su distancia que es igual al eco recibido. En el caso de que no se produzca ningún eco, ya que no se encuentra algún objeto, el pulso de eco tiene una longitud aproximada de 36 ms. Hay que generar un retardo de 10 ms desde que se hace una lectura hasta que se realiza la siguiente, con el fin de que el circuito se estabilice [45].

Tabla 2. Características técnicas del sensor de ultrasonido SRF04 [45]

Voltaje de operación	5V DC
Corriente de trabajo	30 mA típica. 50mA Max
Frecuencia	40Khz
Distancia de alcance	3 cm - 300 cm
Resolución	3 cm
Ángulo de medición	30 grados
Ancho de pulso de entrada Trigger	10 uS
Dimensión	43 mm x 20 mm x 17 mm

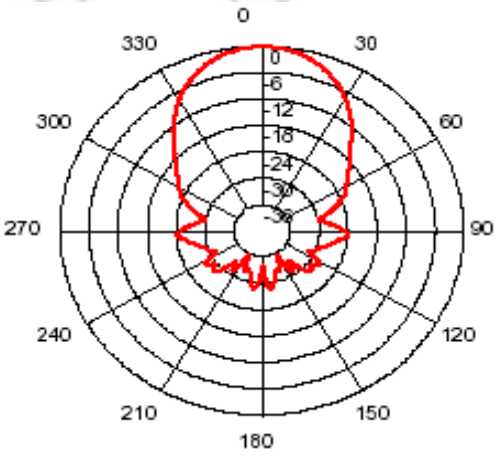


Figura 5. Patrón de haz emitido por el sensor SRF04 [46]

2.8.3.2. HC-SR04

El sensor ultrasónico HC-SR04 emplea un sonar para establecer la distancia a un objeto como lo hacen los delfines o murciélagos. Brinda una excelente detección de rango sin contacto con alta exactitud y lecturas estables en un paquete sencillo de usar. De 2cm a 400 cm o 1" a 13 pies. Su funcionamiento no es afectado por la luz solar o materiales de color negro como con los telémetros Sharp (no obstante, acústicamente materiales suaves como la tela pueden ser difíciles de descubrir). Conformado por un transmisor ultrasónico y un módulo receptor [47].

Tabla 3. Características técnicas del sensor de ultrasonido HC-SR04 [47].

Voltaje de operación	5V DC
Corriente de trabajo	15 mA
Frecuencia	40Khz
Distancia de alcance	2 cm - 400 cm
Resolución	0.3 cm
Ángulo de medición	30 grados
Ancho de pulso de entrada Trigger	10 uS
Dimensión	45 mm x 20 mm x 15 mm

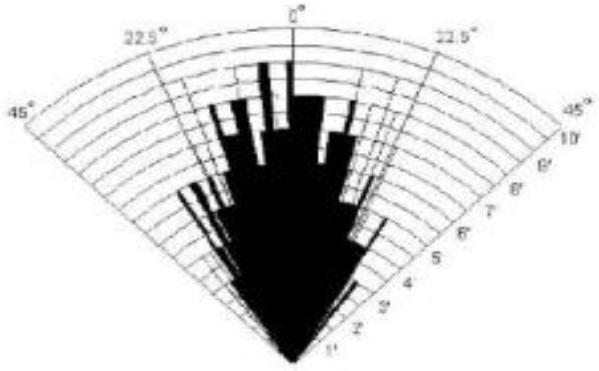


Figura 6. Ángulo efectivo de operación del sensor HC-SR04 *[48]*

2.8.3.3. GY-US42 I2C

El sensor ultrasónico GY-US42 es un módulo de gran exactitud para calcular distancias mayores a 20 cm hasta los 720 cm. El sensor emite mediante su sonda ondas ultrasónicas y al ser irradiadas con el objeto medido, la sonda recibe y devuelve la onda acústica para así utilizar esa variación de tiempo y determinar la distancia existente. El módulo tiene tres formas de leer los datos, por serie UART (nivel TTL); por puerto I2C, valido para cualquier controladora de vuelo APM 2.6/2.8 o Pixhawk para medidas de elevación en los drones. Este sensor es muy utilizado en robots inteligentes, enseñanza de equipos de laboratorio, pruebas de elaboración, laboratorios antropométricos, coches inteligentes o cuadricópteros [49].

Tabla 4. Características técnicas del sensor de ultrasonido GY-US42 I2C [49]

Voltaje de operación	5V DC
Corriente de trabajo	9 mA
Frecuencia	42KHz
Distancia de alcance	20 cm - 720 cm
Resolución	1 cm
Ángulo de medición	30 grados
Dimensión	21.5 mm x 21 mm x 24.5 mm

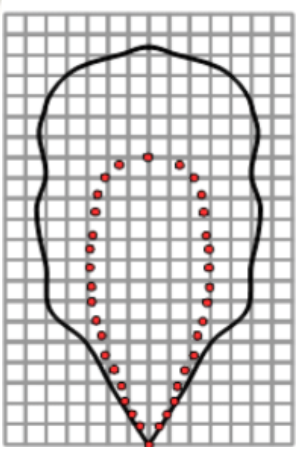


Figura 7. Patrón de haz emitido por el sensor GY-US42 [49]

2.8.4. Sensor de color, luz y visión

Cuando se habla de sistemas de medición de variables físicas, se considera a la luz como un haz, y se omiten todas sus características de onda y sus características determinadas por la mecánica cuántica, ello para permitir al estudiante familiarizarse con los principios de ejercicio de los dispositivos optoelectrónicos. Para cada dispositivo optoelectrónico en especial se debe evaluar si la fuente de luz es externa o independiente del sistema, o si por el contrario la fuente de luz está integrada en el sistema de medición. En la Tabla 5 se muestra una rápida clasificación de sensores de color, luz y visión [41].

Tabla 5. Clasificación de sensores de color, luz y visión [41].

Sensores de luz	Fotorresistencias
	Fototransistores
	Fotodiodos
Sensores de color	Basados en filtros
	Basados en fuentes fijas de luz
Sensores de visión	CCD
	CMOS

2.8.5. Sensor infrarrojo

Los sensores infrarrojos son una variedad de sensor de luz que emplea la parte del espectro de luz denominado infrarrojo. La utilización de esta parte del espectro, no visible para el ser humano, es debido a que presentan un número menor de interferencias que la mayoría de los sensores de luz [44]. Este sistema de cálculo es bastante utilizado con los llamados encoders, en los que el emisor de luz infrarroja y el dispositivo fotosensible (ya sea fotodiodo o fototransistor) se encargan de determinar la variación de posición de un disco ranurado en secciones opacas y transparentes; también, son muy utilizados en sensores de presencia, como complemento de contadores en líneas de producción, cronómetros, etcétera.

El LED infrarrojo (IRLED, diodo emisor de luz infrarroja) es el elemento emisor de luz en el sistema (espectro infrarrojo); la variedad de luz que emite este

mecanismo se encuentra afuera del espectro perceptible para el ojo humano, por lo que para ver si el LED está o no encendido es obligatorio auxiliarse de un aparato electrónico, como una cámara (véase Figura 8).

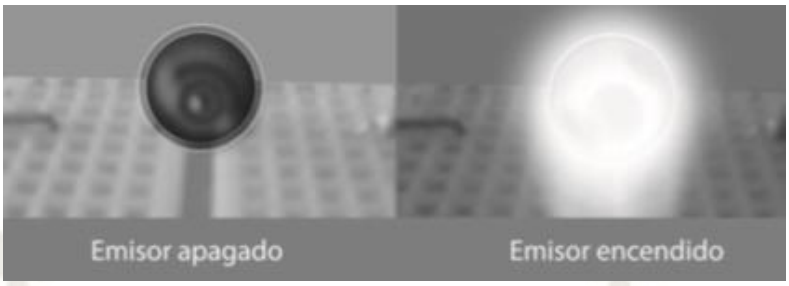


Figura 8. Led infrarrojo visto con la ayuda de una cámara [41].

Como cualquier diodo emisor de luz (LED), el LED infrarrojo también tiene un ánodo y un cátodo. Cuando el diodo se polariza en directa, esto es, cuando el voltaje positivo se encuentra aplicado en el ánodo y la referencia a tierra en el cátodo, este dispositivo emite la luz infrarroja, que es el origen intrínseco del sistema de medida. Hay disponibles distintas configuraciones para aprovechar esta luz emitida por el IRLED; así, este puede aprovecharse para determinar la presencia de un objeto (un objeto reflectante, ranuras opacas o transparentes en encoders ópticos), la distancia de un objeto reflectante, la intensidad de color en ciertas configuraciones (para esto es obligatorio calibrar el sistema), etcétera [41].

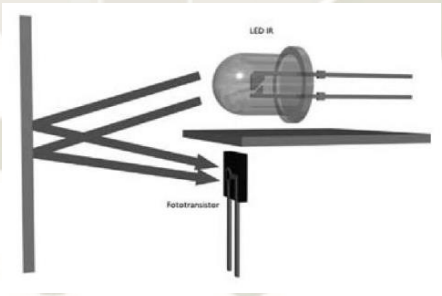


Figura 9. IRLED como detector de presencia o distancia [41].

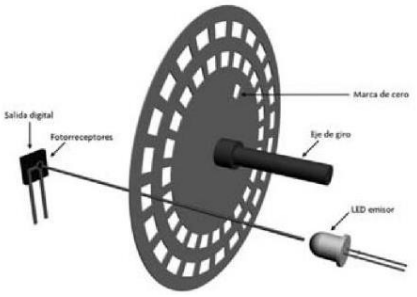


Figura 10. IRLED como parte de un encoder [41].

Las diferentes configuraciones del IRLED se ilustran en la Figura 11.

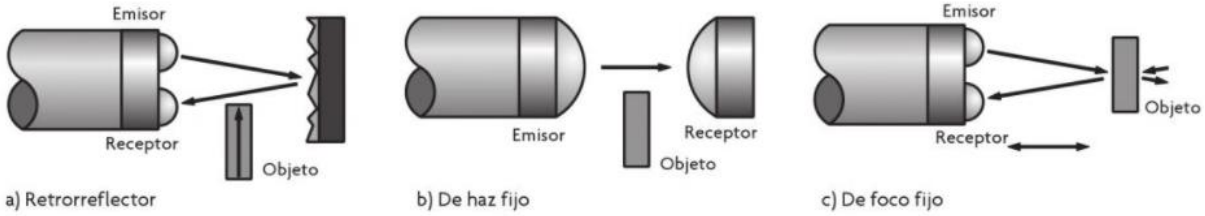


Figura 11. Configuraciones del IRLED [41].

a) Retrorreflector: El objeto refleja el haz de la fuente de luz generando un cambio de intensidad en la señal que produce el receptor; por este caso se detecta la presencia de un objeto. Este arreglo se caracteriza por contar con un objeto extra donde se refleja el origen de luz.

b) De haz fijo: En corriente utilizada en encoders y detectores de presencia.

c) De foco fijo: La intensidad de la señal registrada en el receptor depende de la proximidad del objeto al emisor; esta distribución se utiliza para medidores de distancia [41].

2.8.5.1. QTR-8A

El arreglo de sensores de reflectancia QTR-8A está diseñado como un sensor de línea, pero se puede emplear como un sensor de cercanía o reflectancia de uso corriente. El módulo es un sujetador excelente para ocho pares de emisor y receptor de infrarrojos (fototransistor) situados uniformemente a una distancia de 9.525 mm. Cada fototransistor está conectado a una resistencia pull-up para formar un divisor de voltaje que produce una salida de voltaje analógica entre V y VIN (que suele ser 5 V) en función del Ir reflejado. Un voltaje de salida más bajo es una indicación de una mayor reflexión.

Todas las salidas son independientes, pero los LED están dispuestos en parejas para reducir a la mitad el consumo de corriente. Los LED están controlados por un MOSFET con una compuerta normalmente elevada, lo que hace permitir que los LED se apaguen configurando la puerta MOSFET a un voltaje bajo. Apagar los LED puede dar resultados ventajosos para frenar el consumo de energía cuando los sensores no están en uso o para variar el brillo efectivo de los LED a través del control PWM [50].

Tabla 6. Características técnicas del sensor QTR-8A [50].

Voltaje de operación	5V DC
Formato de salida	Señal analógica
Consumo de corriente	100mA
Señales de salida	8 señales analógicas
Distancia optima de detección	3mm
Distancia máxima de detección	6mm
Dimensión	75mm × 12mm × 3mm (sin clavijas de cabezal instaladas)

2.8.5.2. QTR-3A

El conjunto de sensores de reflectancia QTR-3A está dispuesto como un sensor de línea, pero se puede manipular como un sensor de cercanía o reflectancia de uso general. El módulo es un sujetador excelente para tres pares de emisores y receptores de infrarrojos (fototransistor). Con sensores situados a intervalos de 9.525 mm a lo extenso del eje más largo de la placa, esta matriz funciona bien como un detector simple para robots de rastreo de línea, ya que los recorridos de seguimiento de línea se hacen usualmente usando 3/4" (19 mm) cinta aislante negra. El sensor del medio está levemente desplazado a lo extenso del eje corto de la placa.

Cada fototransistor está acoplado a una resistencia pull-up para crear un divisor de voltaje que produce una salida de voltaje analógica entre V y VCC (que normalmente es de 5 V) en función del Ir reflejado. Un voltaje de salida más bajo es una señal de una mayor reflexión[51].

Tabla 7. Características técnicas del sensor QTR-3A [51].

Voltaje de operación	5V DC
Formato de salida	Señal analógica
Consumo de corriente	50mA
Señales de salida	3 señales analógicas
Distancia optima de detección	3mm
Distancia máxima de detección	6mm
Dimensión	32mm × 8mm × 3mm (sin clavijas de cabezal instaladas)

2.8.5.3. QTR-1A

El sensor de reflectancia Pololu QTR-1A tiene un solo par de fototransistor y LED infrarrojo. El fototransistor está acoplado a una resistencia pull-up para formar un divisor de voltaje que produce una salida de voltaje analógica entre V y VIN (que es típicamente 5 V) en función del Ir reflejado. Un voltaje de salida más bajo es una indicación de una mayor reflexión.

La resistencia limitadora de corriente LED está configurada para entregar cerca de 17 mA al LED cuando el VIN es de 5 V. Algunas líneas de E/S del microcontrolador pueden afrontar el consumo de corriente, lo que permite que el sensor se encienda y apague a través de una E/S línea para ahorrar energía.

El sensor fue creado para usarse con la placa paralela a la superficie que se detecta. por su pequeño tamaño, se pueden constituir corridamente varias unidades para adaptarse a diversas aplicaciones, como la localización de líneas y la detección de proximidad / bordes [52].

Tabla 8. Características técnicas del sensor QTR-3A [52].

Voltaje de operación	5V DC
Formato de salida	Señal analógica
Consumo de corriente	17mA
Señales de salida	1 señal analógica
Distancia optima de detección	3mm
Distancia máxima de detección	6mm
Dimensión	8mm × 13mm × 3mm (sin clavijas de cabezal instaladas)

2.9. Actuadores

Un actuador es un terminal con la capacidad de crear una fuerza que ejerce un cambio de posición, velocidad o estado de algún tipo sobre un dispositivo mecánico, a partir de la conversión de la energía[41]. Los actuadores tienen por principal misión crear el movimiento de los elementos del robot según las órdenes dadas por la unidad de control [53].

Normalmente, los actuadores se clasifican en dos grandes grupos:

- Por el tipo de energía usada: Actuador neumático, hidráulico y eléctrico
- Por el tipo de movimiento que crean: Actuador lineal y rotatorio

Las magnitudes físicas se pueden transformar en otro tipo de magnitudes con características que logran interactuar con el medio; al final, dichas propiedades se presentan como un cambio en el estado de un sistema. Las diferencias involucradas con estas magnitudes suelen ser de fuerza, posición, velocidad y aceleración [41].

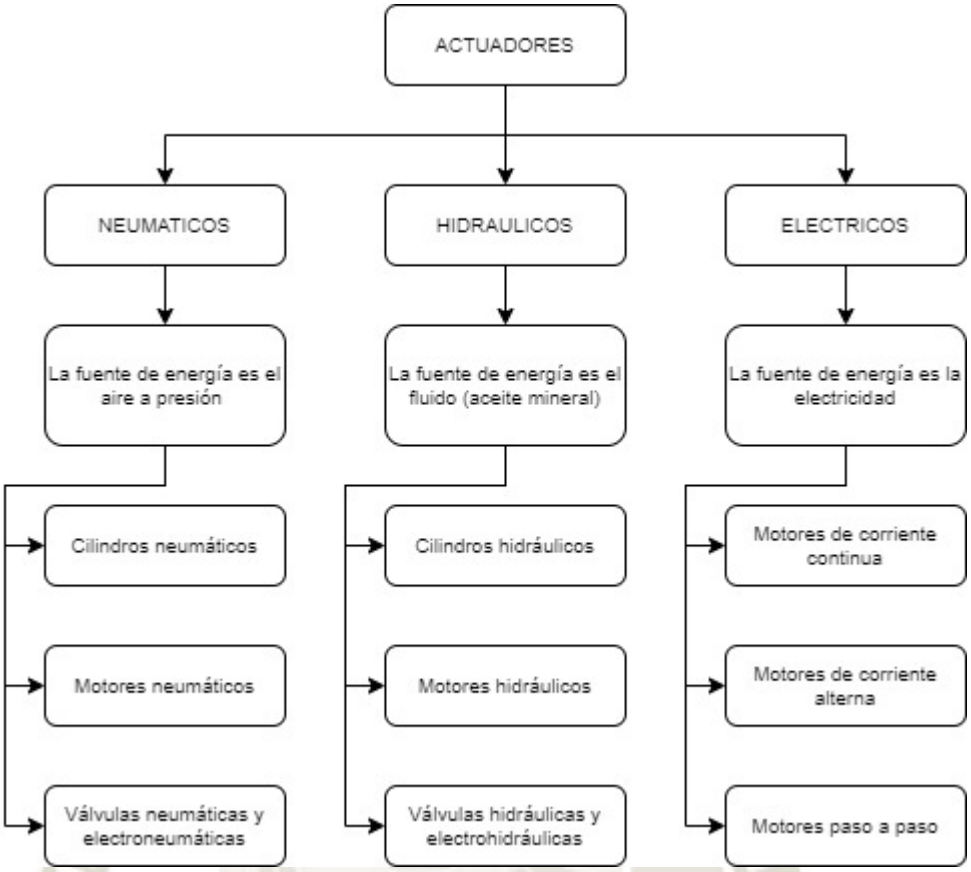


Figura 12. Clasificación de actuadores según el tipo de energía utilizada [41].

Los actuadores crean las fuerzas o pares que se necesitan para animar el esqueleto mecánico. Se utilizan tecnologías hidráulicas, para generar potencias importantes, y neumáticas, pero en hoy en día se ha extendido al empleo de motores eléctricos, y particularmente motores de corriente continua servocontrolados, utilizando en algunos casos motores paso a paso y otros actuadores electromecánicos sin escobillas. Existen también robots industriales de accionamiento directo que permiten excluir los problemas mecánicos relacionados al empleo de engranajes y otras transmisiones. Se investiga en nuevos actuadores que disminuyan la inercia, suministren un par elevado, aumenten la exactitud, produzcan menos ruido magnético y sean de bajos peso y consumo [54].

2.9.1. Actuadores eléctricos

Los actuadores eléctricos convierten la energía eléctrica en energía mecánica, ya sea rotacional o lineal. De los actuadores disponibles en el mercado, estos son los que se usan con mayor demanda, ya que su operan con energía

eléctrica, que es el tipo de energía que se encuentra disponible en la red de repartición eléctrica, por otro lado, los actuadores que son alimentados con energía neumática o hidráulica requieren compresores para producir la energía [41].

2.9.2. Principio de funcionamiento de los actuadores eléctricos

En el entorno de la física es muy reconocido que una partícula cargada eléctricamente ubicada dentro del espacio de ejercicio de un campo magnético está expuesta a una fuerza electromagnética (Fe), la cual se representa mediante la siguiente expresión:

$$F_E = e(B + vu\vec{B})$$

donde:

e: carga del electrón [c]

v: vector de velocidad [m/s]

B: intensidad del campo magnético [T/m]

$\vec{B}$: vector de inducción magnética [Wbm-2]

Además, esta peculiaridad de la partícula puede extenderse a un filamento conductor por el cual circula una corriente eléctrica, de acuerdo con la expresión que se muestra a continuación:

$$F_E = \int_0^L Idl \times \vec{B}$$

donde:

L: longitud del conductor [m]

I: corriente eléctrica [A]

$\vec{B}$: vector de inducción magnética [Wbm-2]

Anteriormente se dijo que los actuadores eléctricos se rigen en el principio de funcionamiento explicado con anticipación, en el cual se determina que si en un filamento conductor por el cual circula una corriente eléctrica se posiciona dentro del ejercicio de un campo magnético, dicho filamento experimenta una fuerza electromagnética que induce un desplazamiento perpendicular a las líneas de acción del campo magnético.

Con el fin de ampliar la magnitud de la fuerza de desplazamiento, un actuador eléctrico está conformado por un gran número de filamentos conductores,

conocidos como espiras. Debido a que la corriente eléctrica que circula a través de un conjunto de espiras suele obtener propiedades magnéticas, esto ocasiona el movimiento circular en el eje (rotor) del actuador, debido a la interacción con los polos (imanes o electroimanes), con lo que se crea u origina la energía mecánica [41].

2.9.3. Clasificación de actuadores eléctricos

Normalmente los actuadores eléctricos se etiquetan de acuerdo al tipo de energía eléctrica con la que se alimentan, por el tipo de movimiento que crean y por la forma de excitación, entre otros aspectos [41].

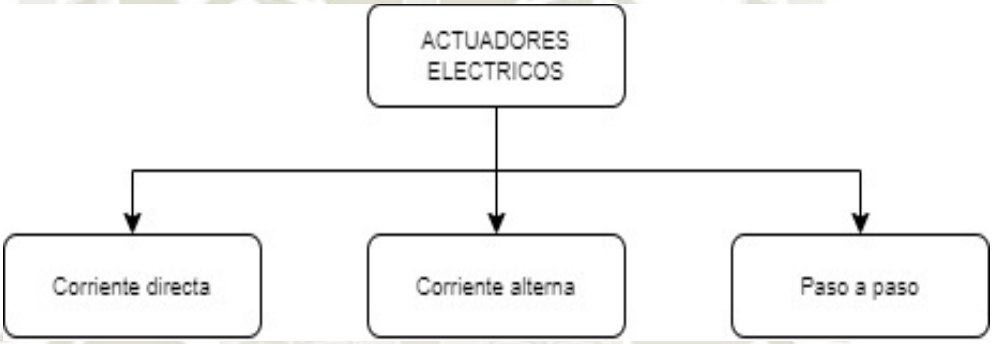


Figura 13. Clasificación de los actuadores eléctricos por el tipo de energía de alimentación [41].

2.9.3.1. Actuadores de corriente directa (DC)

Para su debido trabajo, los actuadores de corriente directa exigen un flujo eléctrico de corriente que fluya en un solo sentido. Este tipo de actuadores se conforman de dos partes fundamentales, conocidas usualmente como rotor y estator. El rotor conforma la parte móvil del actuador, conjuntamente de que es la parte que proporciona la fuerza que actúa sobre el elemento mecánico. Por su parte, el estator conforma la parte fija del actuador y es aquella que provee el magnetismo requerido para incitar la fuerza electromotriz. Una de las principales características de los actuadores de corriente directa radica en que al modificar el voltaje de alimentación se puede variar la velocidad del eje del actuador, ya que la velocidad de rotación en un motor DC depende directamente del voltaje, además de que el par es proporciona la corriente que circula por su devanado [41].

2.9.3.2. Actuadores de corriente alterna (AC)

Los actuadores de corriente alterna, por su parte, se sirven de un flujo eléctrico en el que la intensidad continuamente cambia de dirección, esto como resultado del cambio habitual de polaridad de la tensión aplicado en las borneras de alimentación del motor.

En aplicaciones de velocidad cambiante, los motores de corriente alterna requieren necesariamente de la frecuencia de operación del voltaje aplicado para cambiar los rangos de velocidad [41].

2.9.3.3. Motores paso a paso

El motor paso a paso funciona con el mismo principio físico propio de los actuadores de DC y AC, solo que esta variedad de actuador electromagnético transforma una serie de impulsos eléctricos en desplazamientos angulares discretos, determinando la capacidad de desplazarse un determinado valor en grados (pasos) del eje motriz dependiendo de las entradas de control. Actualmente, en el mercado existen tres tipos de motores paso a paso:

- De imanes permanentes
- De reluctancia variable
- Híbridos [41].

2.10. Comunicación bluetooth

El protocolo bluetooth va encaminado a las comunicaciones inalámbricas que requieren de un reducido precio y de poco consumo de energía a distancias pequeñas entre dispositivos móviles y redes de área local (del tipo alámbricas o inalámbricas), permitiendo la comunicación de voz y de datos en transmisiones punto a punto o de punto a multipunto[55]. Opera en la banda ISM de los 2.4 GHz y los dispositivos que cuentan con este protocolo son capaces de comunicarse entre sí cuando se encuentran dentro del rango. Las comunicaciones se realizan por radiofrecuencia de forma que los dispositivos no tienen que estar alineados y

pueden incluso estar en diferentes habitaciones si la potencia de transmisión es eficaz [56].

Tabla 9. Clasificación de dispositivos Bluetooth según su potencia de transmisión [56].

Clase	Potencia máxima permitida		Alcance
	mW	dB	
Clase 1	100	20	100 m
Clase 2	2.5	4	10 m
Clase 3	1	0	1 m
Clase 4	0.5	0	0.5 m

Tabla 10. Clasificación de dispositivos Bluetooth según su capacidad de canal [56].

Versión	Ancho de Banda
Versión 1.2	1 Mbit/s
Versión 2.0 + EDR	3 Mbit/s
Versión 3.0 + HS	24 Mbit/s
Versión 4.0	32 Mbit/s
Versión 5.0	50 Mbit/s

2.10.1. Bluetooth HC-05

El módulo bluetooth utilizado en el proyecto es el HC-05, el mismo utiliza el protocolo UART RS232 Serial, y es excelente para la implementación de aplicaciones inalámbricas, debido a su costo reducido, conjuntamente se puede encontrar el chip para que uno mismo pueda soldarlo, ya soldado sobre un módulo (breakout), con los pines para ejecutar la comunicación y un led indicador, siendo esta ultima la opción seleccionada para nuestro modulo[57] [58].

Tabla 11. Características técnicas del módulo bluetooth HC-05 [57] [59].

Versión	Bluetooth V2.0 + EDR
Frecuencia de operación	2.4GHz Banda ISM
Voltaje de operación	3.6 VDC – 6VDC
Corriente de operación	< 40 mA
Corriente modo sleep	< 1mA
Baud Rate ajustable	1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 baudios
Dimensiones	4.4 cm x 1.6 cm x 0.7 cm
Modulación	GFSK (Gaussian Frequency Shift Keying)

Es importante decir que el HC-05 tiene un similar, el módulo HC-06, aparentemente son similares. Una notable diferencia es que el módulo HC-06 funciona como Slave y el HC-05 como Master y Slave. Físicamente de pueden diferenciar por el número de pines. En el HC-06 tiene un conector de 4 pines mientras que el HC-05 trae uno de 6 pines [60].

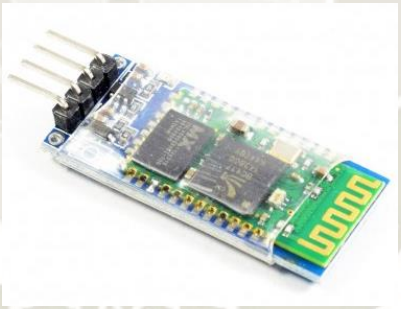


Figura 14. Módulo bluetooth HC-06 [60].

Figura 15. Módulo bluetooth HC-05 [60].

2.10.2. MIT app inventor

MIT App Inventor es un ambiente de programación visual e intuitivo que permite a todos, inclusive a los niños, elaborar aplicaciones completamente funcionales para tabletas y teléfonos inteligentes. Aquellos que son nuevos en MIT App Inventor pueden elaborar una primera aplicación fácil en funcionamiento en menos de 30 minutos. Gracias a su herramienta basada en bloques facilita la elaboración de aplicaciones complejas y de alto impacto en menor tiempo que los entornos de programación tradicionales. El proyecto MIT App Inventor busca democratizar la creación de software empoderando a todas las personas, fundamentalmente a los jóvenes, a pasar del consumo de tecnología a la creación de tecnología.

Un equipo pequeño de personas y estudiantes de CSAIL, encaminado por el catedrático Hal Abelson, forma el núcleo de un movimiento internacional de inventores. Conjuntamente de liderar la eficacia educativa en torno a MIT App Inventor y realizar investigaciones sobre sus impactos, este equipo central mantiene el entorno de desarrollo de aplicaciones en línea gratuita que sirve a más de 6 millones de usuarios.

Los programas de codificación basados en bloques inspiran el empoderamiento científico y creativo. MIT App Inventor va más allá para suministrar un empoderamiento existente a los niños para que marquen la diferencia, una forma de conseguir un impacto social de valor inmenso en sus comunidades [61].

2.11. Baterías LiPo

La batería de polímero de litio reconocida como LiPo, es distinta del resto de las baterías por el electrolito usado. El diseño principal data de los años 70 usando un polímero sólido como electrolito. Este electrolito se ensamblaba en un recipiente plástico que no conducía la electricidad, y que impedía el paso de electrones [62].

Estas baterías son ligeras, flexibles y resistentes a fugas, con un electrolito de gran conductividad que tiene el potencial de conceder una gran potencia. Las baterías LiPo son de tipo recargable y entre del mundo de los sistemas de almacenamiento son de última generación, teniendo un gran desempeño [63].

Las celdas de polímero de litio utilizan una bolsa flexible de aluminio en lugar de fundas rígidas, esto conlleva un ahorro enorme en el tamaño y el peso que demanda cada celda, un 20% más ligero que el equivalente en pilas cilíndricas [62].

Dependiendo de la conexión interna de las celdas de una batería se obtiene menor o mayor nivel de voltaje, de modo que éstas pueden conectarse en serie o paralelo, considerando que cada celda tiene un voltaje de 3.7 voltios de valor nominal [63].

Tabla 12. Ventajas y desventajas de las baterías LiPo [62].

Ventajas	Desventajas
Pueden reducirse hasta grosores de 1 milímetro	Requiere un circuito de seguridad para mantener los límites de voltaje
Pueden empaquetarse de múltiples formas	Almacenar en lugar frío al 40% de su carga
Alta densidad de energía	Limitaciones en su transporte (compañías aéreas)
Poco peso	Tecnología en desarrollo
No necesitan mantenimiento	Pueden explotar si se perfora
Sin efecto memoria	
Bajo porcentaje de autodescarga	

2.12. Impresión 3D

Fabricación aditiva es el seudónimo técnico que abarca todas las tecnologías de impresión 3D, se trata de la elaboración de objetos tridimensionales por aportación de material en vez de sustracción. En impresión 3D, partiendo de un archivo digital (modelo 3D), se utilizan en variados procesos aditivos en los que se aplican capas sucesivas de materia para establecer un objeto tangible. Existen varias formas de abordar el problema y cada tecnología tiene sus virtudes e inconvenientes.

La fabricación aditiva en un contexto social o como herramienta en si es una tecnología salvadora ya que destruye casi todas las limitaciones que presentan las tecnologías de fabricación tradicional.

Se quiere decir que una impresora 3D no entiende diferencias en cuanto a la complicación de una forma, le cuesta el mismo esfuerzo fabricar un simple prisma

de 6 caras que algo que desde un punto de vista humano se considera mucho más complejo como la réplica de un adorno barroco de la fachada de una iglesia. Esta libertad de creación no tiene precedentes en la historia ya que hasta ahora para elaborar objetos hemos dependido siempre de las limitaciones de forma que imponen el uso de moldes o herramientas de corte y desbaste como una fresadora.

Una impresora 3D no solo es capaz de elaborar formas complejas, sino que asimismo es competente de elaborar un objeto adentro de otro, como por ejemplo elaborar un silbato con la bola adentro. Esto significa que una impresora 3D es capaz de producir objetos complejos que no requieren de pasos posteriores de ensamblaje. Es decir, hacer una llave inglesa ya terminada y ensamblada: el cuerpo de la llave, el diente y la rosca todos impresos a la vez, completamente ensamblados y funcionales.

Las libertades de las que disfruta esta tecnología unidas a la democratización y acceso universal que se está produciendo tanto con los programas de modelado como con la impresión 3D a través de servicios online, fablabs o makerspaces indican que la fabricación aditiva supondrá un antes y un después para el mundo de la elaboración. Fielmente estamos hablando de acercar y exponer el mundo del diseño y la fabricación a un amplio espectro de público.

Esto quizá no tiene mucho interés para aquellos que ya tienen conocimientos o práctica en el sector, pero sí que supone un cambio revelador para el público no especializado. Esto tiene como significado que millones de personas que precedentemente nunca habrían pensado en experimentar con la creación de algún objeto ahora tienen la oportunidad de probar, experimentar y por tanto aportar alguna innovación [64].

CAPITULO III: DESARROLLO DE INGENIERIA / INGENIERIA DE DETALLE

3.1.Metodología de diseño

La metodología de diseño que se aplica para el desarrollo del módulo es el diseño Top Down, consiste en comenzar con una idea con alto nivel de abstracción e ir incrementando el nivel de detalle a medida que se vaya avanzando en el diseño. La idea principal del módulo se dividirá en subideas y cada subidea se subdividirá en más subideas y así sucesivamente hasta llegar a los componentes primarios de diseño, como se muestra en la Figura 15.

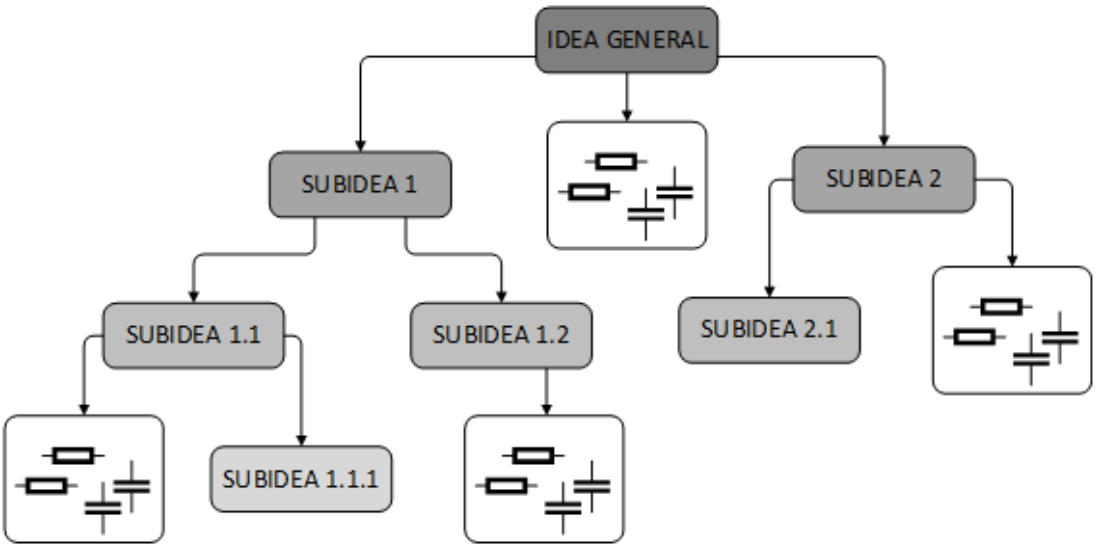


Figura 16. Ejemplo de diseño Top Down.
Fuente: Elaboración propia.

3.2.Descripción comportamental del circuito

La función principal del circuito es controlar un robot móvil al que se le añadirá una gran variedad de sensores y actuadores, contando con conectores normalizados y señalados, permitiendo al estudiante la fácil conexión y enfocando su atención en la programación del mismo sin dejar de lado conceptos de electrónica.

El circuito también contara con un botón integrado para iniciar o detener la rutina del robot y también contara con indicadores visuales y auditivos.

El circuito debe ser capaz de realizar las siguientes funciones:

- Comunicarse con una PC a través de un puerto USB
- Acondicionar las señales de los sensores (análoga o digital)
- Controlar 2 motores con corriente de arranque de 1.6A máximo
- Contar con E/S digitales y análogas para conectar sensores y actuadores
- Brindar un voltaje de 5V para conectar distintos tipos de sensores
- Permitir la comunicación vía bluetooth
- Contar con un sistema de protección contra inversión de polaridad
- Funcionar a un voltaje de alimentación máximo de 12V
- Permitir el reemplazo de componentes dañados de forma rápida

3.3.Diagrama de conexiones del circuito

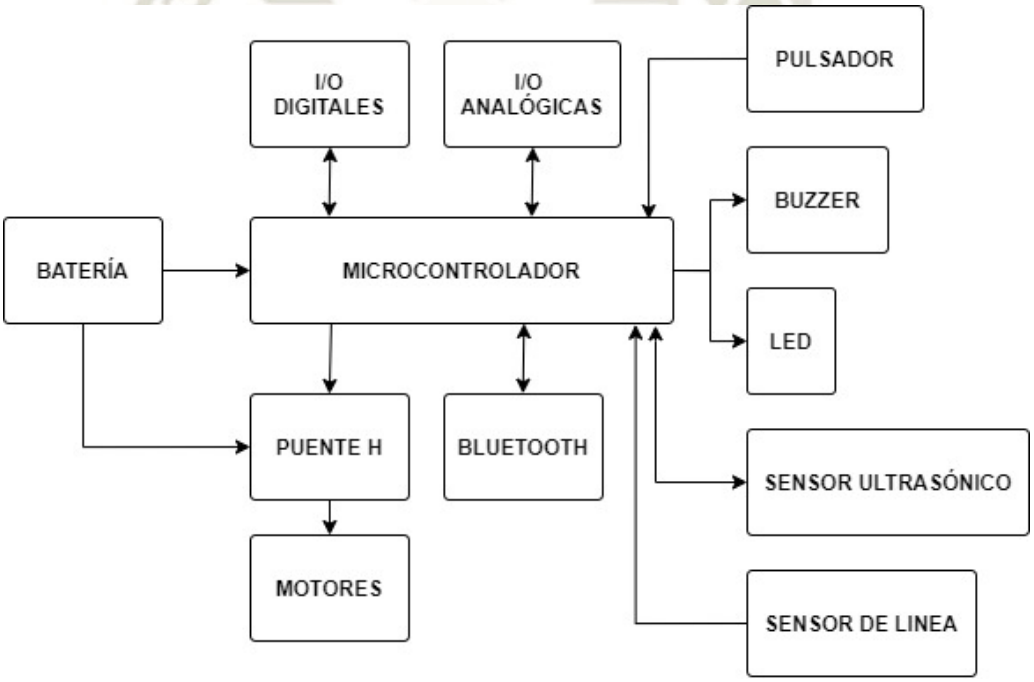


Figura 17. Diagrama de bloques del circuito

Fuente: Elaboración propia.

3.4.Especificaciones técnicas del robot

Tabla 13. Ficha técnica

FICHA TECNICA			
MECANICA	Tracción	Diferencial	
	Peso	150 grm	
	Dimensiones	10 x 10 x 10 cm	
ELECTRONICA	Microcontrolador	Arduino nano	Múltiples I/O
	Sensores	Línea	QTR-3A
		Ultrasónico	HC-SR04
		Pulsador	12mm
		Entradas extra	Pines normalizados
	Actuadores	Micromotores	6v 800 RPM
		Servomotor	SG90
		Buzzer	5v
		Led	5mm
		Salidas extra	Pines normalizados
	Alimentación	Batería lipo	7.4v 850mah
ADICIONALES	Resistente a golpes y caídas		
	Fácil de cablear, ensamblar y programar		
	Capaz de desplazarse sobre distintas superficies		
	Piezas personalizables (colores y formas)		

3.5.Comparación y selección del lenguaje de programación

A continuación, se presenta una tabla comparativa entre los lenguajes de programación seleccionados detallando las ventajas y desventajas de cada uno, enfocándonos en resaltar algunas características que son deseables en un lenguaje usado en un primer curso de programación.

Tabla 14. Comparación de lenguajes de programación

LENGUAJE DE PROGRAMACION	PARADIGMA	VENTAJAS	DESVENTAJAS
Ensamblador	Imperativo	• Precisión • Código breve • Entendimiento del microcontrolador	• No es portable • No posee estructura • Código difícil de iterar
C++	Imperativo Estructurado	• Portable • Veloz • Fácil de iterar • Variedad de librerías	• Menor velocidad que ensamblador • Código ocupa más memoria
Python	Imperativo Orientado a objetos Funcional	• Portable • Simplificado • Flexible • Variedad de librerías	• Lenguaje lento • Falta de documentación

Por todo lo expuesto anteriormente en la tabla, se decidió trabajar con el lenguaje C++ en este módulo, por todas las ventajas que brinda y sin preocuparnos por la eficiencia o el espacio de memoria que ocupa el código, debido al tipo de aplicación pedagógica que estamos realizando.

3.6.Comparación y selección del microcontrolador

A continuación, se presenta una tabla comparativa de los microcontroladores seleccionados detallando las ventajas y desventajas de cada uno, enfocándonos en resaltar algunas características que son deseables en un microcontrolador capaz de controlar un robot aplicado al ámbito educativo.

Tabla 15. Comparación de microcontroladores

MICROCONTROLADOR	LENGUAJE	VENTAJAS	DESVENTAJAS
PIC	Ensamblador C++	• Tamaño compacto • Múltiples entradas y salidas • Bajo costo • Variedad de modelos	• Necesita componentes extra para programar y funcionar • Software especializado
Arduino	C++	• Placa de desarrollo • Múltiples entradas y salidas • Bajo costo • Open source	• Poca memoria de almacenamiento • Procesamiento limitado (16 bits)
Raspberry Pi	Python	• Potencia de cálculo • Cuenta con sistema operativo • Gran capacidad almacenamiento	• Costo elevado • Complicada para proyectos de electrónica

Los microcontroladores anteriormente expuestos son ideales para nuestros proyectos, son asequibles y cada uno tiene sus puntos fuertes y sus puntos débiles. Un Arduino puede encender un led con cuatro líneas de código, para aprender a hacerlo con PIC se requiere mucho más y en lenguajes como Ensamblador es una tarea un poco más fuerte.

Arduino por lo general serán más fáciles de utilizar en pequeños proyectos, mientras que la Raspberry Pi gracias a su poder de procesamiento es más adecuada para proyectos más complejos o que requieran de un sistema operativo completo.

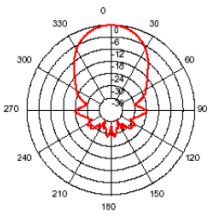
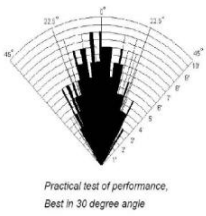
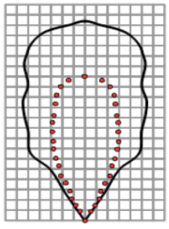
Todo lo anteriormente mencionado demuestra que es mucho más fácil aprender a trabajar con Arduino, siendo el microcontrolador asignado para este proyecto, debido a que es la alternativa más amigable para acercar a los alumnos al entendimiento de la electrónica, robótica y programación.

3.7.Sensores

3.7.1. Comparación y selección del sensor ultrasónico

Dentro de la amplia gama que existe de sensores, nos interesan aquellos que sirven para localizar un obstáculo que se encuentre en la ruta de nuestro modulo ya sea estático o en movimiento, además necesitamos que el sensor pueda detectar diferentes materiales sin importar el ambiente o condición, es por eso que se decidió utilizar un sensor del tipo ultrasónico.

Tabla 16. Comparación de 3 modelos de sensores de ultrasonido

	DEVANTECH SRF04	HC-SR04	GY-US42 I2C
Presentación			
Voltaje de operación	5V DC	5V DC	5V DC
Corriente de trabajo	30 – 50 mA	15 mA	9 mA
Comunicación	TTL	TTL	TTL – I2C
Rango de lectura	3 – 300 cm	2 – 400 cm	20 – 720 cm
Dimensión	43 x 20 x 17 mm	45 x 20 x 15 mm	21.5 x 21 x 24.5 mm
Ángulo de medición			

El modelo elegido para utilizar en el módulo es un HC-SR04. Debido al rango de medición de 2 a 400 cm ideal para nuestra aplicación. Algo muy importante es que no se requiere de un interfaz de comunicación I2C, facilitando la implementación ya que solo requiere de un pulso de disparo TTL para funcionar y solo será necesario destinar 2 pines del microcontrolador para trabajar con este sensor.

Este sensor es un modelo muy comercial, de costo económico y cuenta con una gran variedad de soportes disponibles en internet, los cuales serán utilizados como referencia para el diseño del soporte específico para nuestro modulo.

3.7.1.1. Cálculo de distancia

En el diagrama de tiempo mostrado en la Figura 18, podemos observar que se necesita un pulso de disparo de 10us para activar el sensor, luego el sensor envía un ultrasonido de 8 ciclos a 40 Khz y pone la línea de eco en estado alto. A continuación, se espera a escuchar el eco, una vez detectado, la línea de eco se pone en estado bajo, formando así un ancho de pulso que es directamente proporcional a la distancia del objeto.

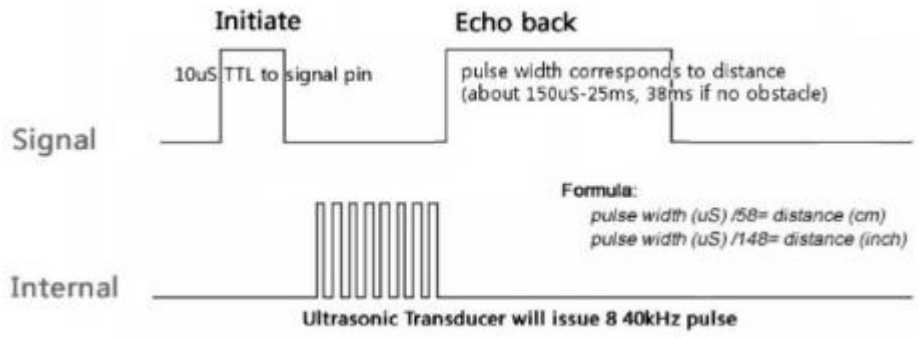


Figura 18. Pulsos emitidos por el sensor ultrasónico [65]

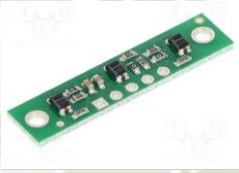
Midiendo el tiempo que dura el ancho de pulso, es posible calcular el rango en pulgadas o centímetros. Si el sensor no detecta nada, después de 30ms la línea de eco será puesta en bajo de cualquier forma. Si se mide el ancho de pulso en us, entonces será necesario dividir entre 58 para obtener la distancia en centímetros y entre 148 para pulgadas.

Podemos disparar el sensor HC-SR04 cada 50ms, es decir 20 veces cada segundo y después de cada disparo debemos esperar al menos 20ms antes de enviar el siguiente disparo, aun cuando el ancho de pulso del eco anterior fue pequeño, esto para no causar un falso eco.

3.7.2. Comparación y selección del sensor de línea

Este sensor como su nombre indica, será el encargado de detectar una línea negra sobre un fondo blanco o viceversa. Generalmente los arreglos de sensores se pueden elaborar utilizando diodos led infrarrojos, pero como se busca un tamaño y consumo reducido, evaluaremos los módulos disponibles en el mercado.

Tabla 17. Comparación de 3 modelos de sensores de línea

	QTR-1A	QTR-3A	QTR-8A
Presentación			
Voltaje de operación	5V DC	5V DC	5V DC
Corriente de trabajo	17 mA	50 mA	100 mA
Formato de salida	señal analógica	señal analógica	señal analógica
Numero de salidas	1	3	8
Distancia optima de detección	3 mm	3 mm	3 mm
Dimensión	8 x 13 x 3 mm	32 x 8 x 3 mm	75 x 12 x 3 mm

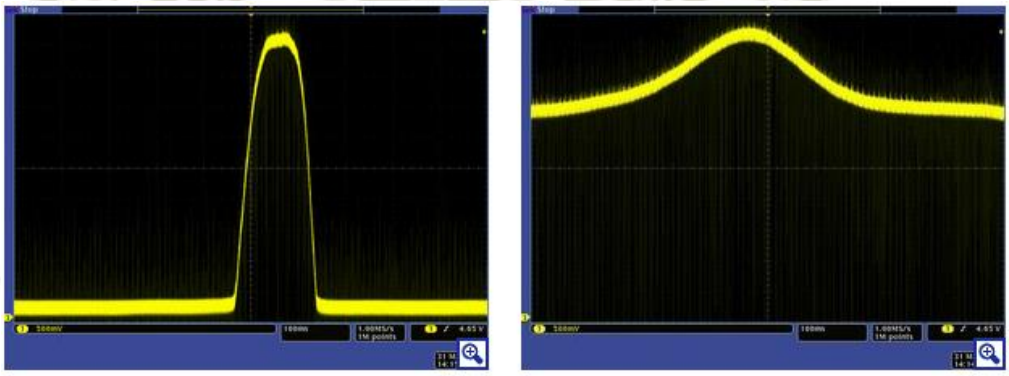
Después de evaluar los sensores anteriormente mencionados, se eligió el sensor de línea QTR-3A, ya que como el fabricante menciona, este funciona como un sistema de detección mínimo para robots seguidores de línea y gracias a la disposición de los sensores se podrán implementar distintas condiciones que permitirán detectar curvas muy cerradas o intermitencias en la línea. Otra ventaja del modelo QTR-3A es que está diseñado para detectar una línea entre 19 a 20mm, el grosor estándar para las competencias de robots seguidores de línea, además el diseño es compacto y nos facilita crear un soporte impreso en 3D para poder protegerlo.

3.7.2.1. Detección de línea

Hay varias formas de interactuar con las salidas QTR-3A, pero para esta experiencia nos vamos a centrar en 2:

- Utilizar un convertidor de analógico a digital (ADC) del microcontrolador para medir los voltajes.
- Conectar cada salida directamente a una línea de E/S digital del microcontrolador y ayudarnos de su comparador interno.

Este último método funcionará si puede obtener una alta reflectancia de su superficie blanca como se muestra en la imagen de la izquierda de la Figura 19, pero probablemente fallará si tiene un perfil de señal de menor reflectancia como el de la derecha.



Salida QTR-1A a 1/8 "de distancia de un disco blanco giratorio con una línea negra.

Salida QTR-1A a 3/8 "de distancia de un disco blanco giratorio con una línea negra.

Figura 19. Representación del método de detección de la línea [51]

Existen bibliotecas para trabajar este tipo de sensores, inclusive el fabricante nos brinda una, pero para este proyecto solo nos guiaremos de la señal brindada por el sensor, de esta forma los estudiantes podrán entender de mejor manera los conceptos de sensores y tendrán más libertad al momento de programar el módulo.

3.8. Actuadores

3.8.1. Servomotor SG90

Este tipo de servo es ideal para las primeras experiencias de aprendizaje y prácticas con servos, ya que sus requerimientos de energía son bastante bajos y

permite alimentarlo con la misma fuente de alimentación que el circuito de control. Muy usado en aeromodelismo, pequeños brazos robóticos y mini artrópodos.

Puede rotar aproximadamente 180 grados (90° en cada dirección). Tiene la facilidad de poder trabajar con diversidad de plataformas de desarrollo como Arduino, PICs, Raspberry Pi, o en general a cualquier microcontrolador.

Características:

- Voltaje de operación: 3.0 - 7.2V
- Consumo de corriente: 250mA (±50)
- Velocidad: 0.1seg / 60 grados
- Torque reposo: 1.3Kg x cm (4.8V), 1.6Kg (6.0V)
- Ancho de pulso: 4useg (Dead band)
- Peso: 9g
- Engranajes: Nylon
- Dimensiones: 22*11.5*27 mm
- Longitud del conductor: 150mm

3.8.2. Micromotor con caja reductora

Con una sección transversal que mide solo 10 × 12mm, estos pequeños motorreductores DC con escobillas están disponibles en una amplia gama de relaciones de engranajes, desde 5: 1 hasta 1000: 1. Con la excepción de las versiones de relación de transmisión de 1000: 1, todos los motorreductores de micro-metal tienen las mismas dimensiones físicas, por lo que generalmente es fácil cambiar una versión por otra, si cambian los requisitos de diseño.

Características:

- Voltaje de operación: 6V DC
- Corriente de arranque: 1.5 A
- Corriente sin carga: 100 mA
- Relación de transmisión: 50:1
- RPM (velocidad sin carga): 650 RPM
- Torque: 0.74 Kg/cm
- Potencia máxima: 1.2W

3.8.3. Driver TB6612FNG

El controlador de motor TB6612FNG puede controlar hasta dos motores de DC a una corriente constante de 1,2 A (pico de 3,2 A). Se pueden usar dos señales de entrada (IN1 e IN2) para controlar el motor en uno de los cuatro modos de función: CW, CCW, freno corto y parada. Las dos salidas del motor (A y B) se pueden controlar por separado, y la velocidad de cada motor se controla mediante una señal de entrada PWM con una frecuencia de hasta 100 kHz. El pin STBY se debe poner en estado alto para sacar el motor del modo de espera.

La tensión de alimentación lógica (VCC) puede estar en el rango de 2,7 a 5,5V, mientras que la alimentación del motor (VM) está limitada a una tensión máxima de 15V. La corriente de salida tiene una potencia de hasta 1,2 A por canal (o hasta 3,2 A para un pulso único y corto).

3.8.4. Cálculo de la velocidad del modulo

Con las características brindadas anteriormente se puede utilizar una fórmula para poder tener un estimado de la velocidad de desplazamiento del módulo, utilizando como variables el diámetro de la rueda (3.5cm) y las RPM (600) del motor

Cálculo de la circunferencia:

$$circunferencia = diametro \times \pi$$

$$circunferencia = 3.5 \times 3.14$$

$$\boldsymbol{circunferencia = 10.99\ cm\ o\ 4.32\ pulgadas}$$

Cálculo de la velocidad:

$$velocidad = circunferencia \times RPM$$

$$velocidad = 4.32 \times 600$$

$$\boldsymbol{velocidad = 2592\ pulgadas/\min\ o\ \ 1.09\ m/s}$$

3.9.Diseño y desarrollo de la placa de circuito impreso

3.9.1. Cálculo de consumo del circuito

Para calcular el consumo de nuestro modulo necesitamos identificar los voltajes de operación de cada uno de los componentes electrónicos que vamos a utilizar y también debemos revisar el consumo de corriente, eso nos permitirá definir el grosor adecuado de pistas y también poder elegir la batería adecuada.

Tabla 18. Consumo de corriente y voltaje de los elementos del circuito

	VOLTAJE DE OPERACIÓN (V)	CONSUMO DE CORRIENTE (mA)
Microcontrolador	6 a 12 V	15 mA
Sensor ultrasónico HC-SR04	5 V	15 mA
Sensor de línea QTR-3A	5 V	50 mA
Bluetooth HC-05	5 V	40 mA
Driver controlador de motores	5 V	5 mA
Micromotor	6 a 8 V	200 mA
Servomotor SG-90	5 V	220 mA
Buzzer	5 V	10 mA
Led	5 V	15 mA

Estos valores son teóricos, pero nos servirán para ir desarrollando la placa de circuito impreso PCB. Una vez implementada la placa, se procederá a realizar la medición real y contrastar los valores teóricos con los medidos.

3.9.2. Dimensión de resistencias

Debido al bajo consumo de todos los componentes del módulo, a excepción de los motores que se alimentan directamente de la batería, es que se está utilizando el regulador de voltaje interno integrado en el Arduino Nano que es el modelo AMS1117.

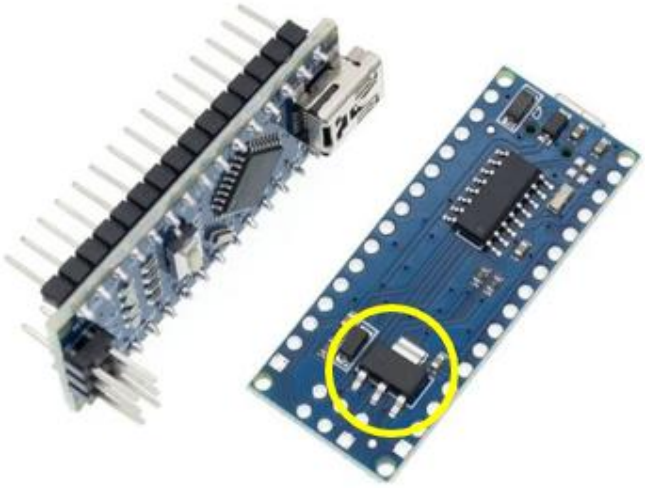


Figura 20. Regulador de voltaje integrado en el Arduino Nano [66]

La serie AMS1117 de reguladores de voltaje fijos y ajustables está diseñada para proporcionar una corriente de salida de hasta 1 A y funcionar hasta 1V diferencial de entrada a salida. El voltaje de caída del dispositivo está garantizado como máximo 1.3V, disminuyendo al mínimo las corrientes de carga.

Sabiendo esos parámetros podemos aplicar la Ley de Ohm para calcular la dimensión de las resistencias, este proyecto utilizara 3 tipos de resistencias:

- Resistencia de 330 ohm para limitar la corriente que circula por el led
- Resistencia de 1Kohm en la etapa de amplificación del buzzer
- Resistencia de 10Kohm que son las recomendadas para utilizar como pull-down y permiten al microcontrolador distinguir el voltaje de entrada del pulsador

$$I = \frac{V}{R}$$

$$I = \frac{5V}{330\ ohms} \quad I = \frac{5V}{10000\ ohms} \quad I = \frac{5V}{1000\ ohms}$$

$$\boldsymbol{I = 0.015\ A} \quad \boldsymbol{I = 0.0005\ A} \quad \boldsymbol{I = 0.005\ A}$$

$$P_R = V_R \times I_R$$

$$P_R = 5V \times 0.015A \quad P_R = 5V \times 0.0005A \quad P_R = 5V \times 0.005A$$

$$\boldsymbol{P_R = 0.075\ watts} \quad \boldsymbol{P_R = 0.0025\ watts} \quad \boldsymbol{P_R = 0.025\ watts}$$

Es decir, muy por debajo de la potencia de 1/4W. Por lo tanto, la potencia de nuestras resistencias será 1/4W con lo cual nos sobra y podríamos inclusive reemplazarlas por las de 1/8W que son mucho más pequeñas.

3.9.3. Cálculo del ancho de pistas

El estándar general para el diseño de circuitos impresos, ANSI-IPC 2221, recomienda el cálculo del ancho de pista mediante la siguiente ecuación:

$$\text{Ancho} = \frac{\left[\frac{\text{I}}{0.0647 \times \Delta\text{T}^{0.4281}}\right]^{1.485442662}}{\text{Alto} \times 1.378}$$

donde:

ΔT: diferencia de temperatura con el ambiente

I: corriente máxima que circula por la pista

Alto: espesor en onzas del cobre

Dados los parámetros anteriores se considera como corriente máxima 1.5A que es el valor que nos brindan los micromotores bajo carga, cabe aclarar que el resto de los componentes presentan un consumo mínimo de corriente, se aplicara el ancho calculado a todas las pistas para facilitar la posterior implementación. La diferencia de temperatura con el ambiente será de 15° y el espesor de la PCB será de 1 onza.

$$\text{Ancho} = \frac{\left[\frac{1.5}{0.0647 \times 15^{0.4281}}\right]^{1.485442662}}{1 \times 1.378}$$

$$\text{Ancho} = \frac{\left[\frac{1.5}{0.2062}\right]^{1.485442662}}{1.378}$$

$$\textbf{Ancho} = \mathbf{13.83\ Milesimas}$$

Se determinó que el ancho de las pistas aplicado en la placa de circuito impreso seria de 22 milésimas, permitiéndonos la posibilidad de prototipar mediante el método de planchado o serigrafía, para más adelante mandar a elaborar la placa terminada con un fabricante especializado del exterior.

También hay que tener en cuenta la separación mínima entre las zonas conductoras (las pistas) en función de la tensión de trabajo. A continuación, se muestra una tabla en la que se puede apreciar la separación mínima en milímetros en función de la tensión.

Tabla 19. Separación mínima entra las zonas conductoras en función al voltaje

VOLTAJE (V)	SEPARACION MINIMA (mm)
0 a 30	0.317
31 a 50	0.444
51 a 150	0.571
151 a 300	0.825
301 a 500	1.587
>500	0.003 mm/V

Por ejemplo, consultando la tabla anterior, la separación mínima será de 0,317 milímetros. Aunque como norma general de diseño se recomienda una distancia mínima de 0.4 milímetros

3.9.4. Placa de circuito impreso

El software utilizado fue EagleCAD de Autodesk tanto para el diseño del circuito esquemático como para la placa de circuito impreso, se utilizaron componentes thru-hole debido a que son de fácil acceso y no requieren de mayor cuidado al momento de montarlos en la PCB y ofrecen una ventaja modular al momento de reemplazar algún componente dañado.

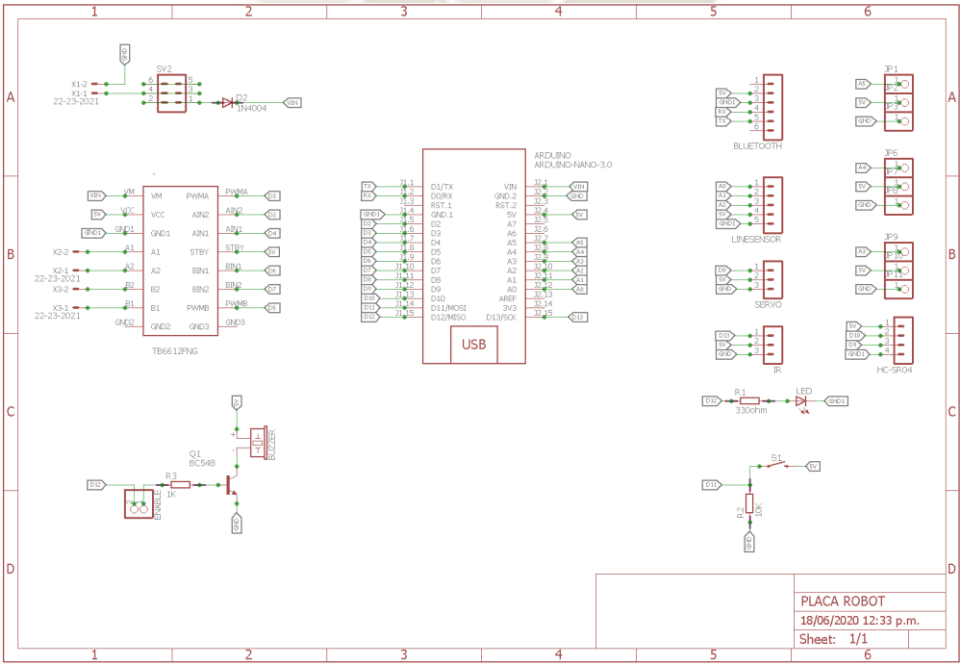


Figura 21. Diseño del circuito esquemático

Fuente: Elaboración propia.

Para la selección del Arduino nano y el driver TB6612FNG en el programa EagleCAD utilizamos una librería disponible llamada DIY MODULES.

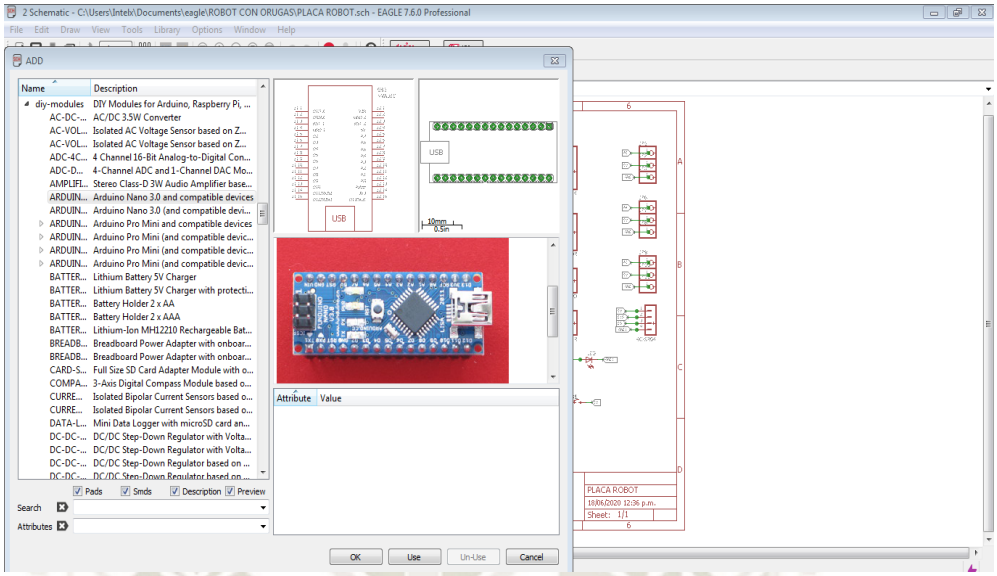


Figura 22. Librería DIY MODULES en EagleCAD

Fuente: Elaboración propia.

Para el diseño de la placa de circuito impresa PCB se tomó en consideración que la medida máxima debía ser de 8 x 8 cm, ya que el tamaño total de nuestro modulo educativo no debía exceder los 10 x 10 cm incluyendo accesorios, para reducir el espacio se trabajó una PCB de 2 caras con vías metalizadas, adicionalmente se añadieron agujeros de 3mm para fijar la placa al chasis del robot utilizando espaciadores de bronce.

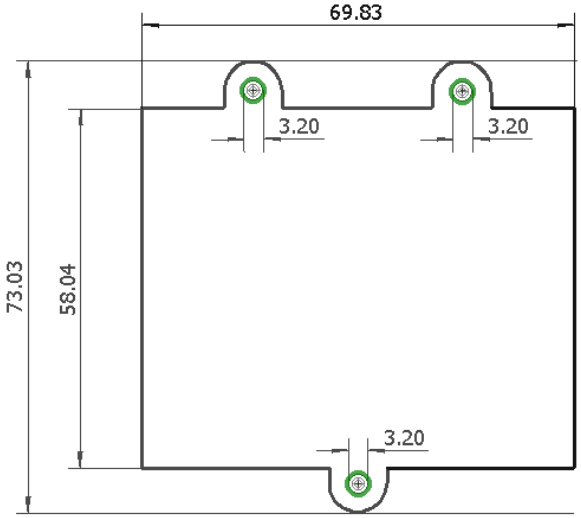


Figura 23. Dimensiones de la placa de circuito impreso

Fuente: Elaboración propia.

Se procedió a ubicar todos los componentes, distribuyéndolos de la mejor forma posible, considerando que los sensores con conexión normalizada (línea y ultrasónico) estarían situados en la parte delantera del robot, también se ubicaron los conectores para los motores y la alimentación en los bordes para evitar que el cableado interfiera con el indicador led o con el pulsador.

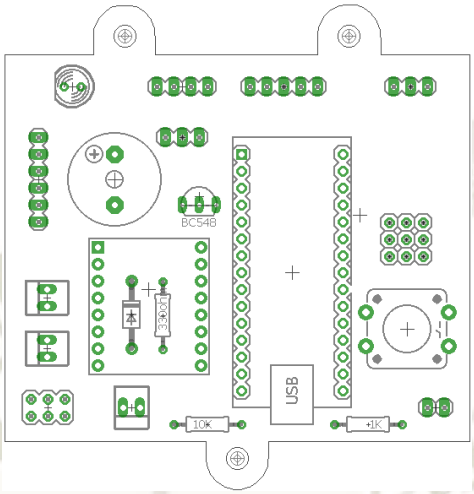


Figura 24. Distribución de los componentes en la placa de circuito impreso

Fuente: Elaboración propia.

Una vez situados los componentes procedemos con el ruteo de los mismos, respetando la medida para el ancho de vías determinada anteriormente de 22 milésimas para todo el circuito a excepción de las líneas de alimentación Vin, GND y salida de motores que fueron reforzadas ya que el regulador ASM1117 con el que cuenta Arduino permite voltajes de hasta 16 voltios, de igual forma el driver de motores.

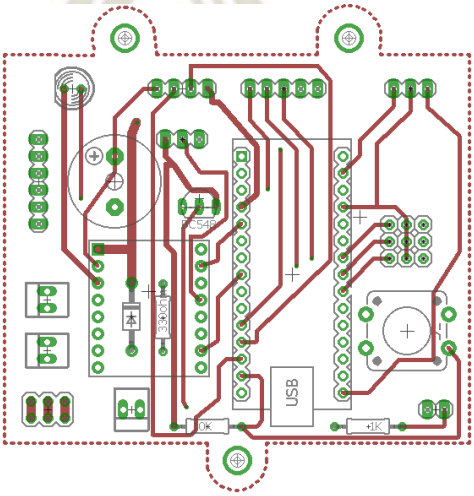


Figura 25. Ruteo de la capa superior (Top Layer)

Fuente: Elaboración propia.

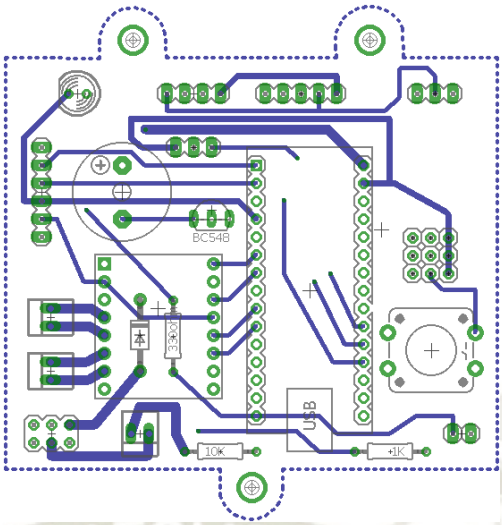


Figura 26. Ruteo de la capa inferior (Bottom Layer)

Fuente: Elaboración propia.

El diseño final nos queda de la siguiente manera, situando todos los componentes dentro del contorno, cumpliendo con los requerimientos anteriormente planteados.

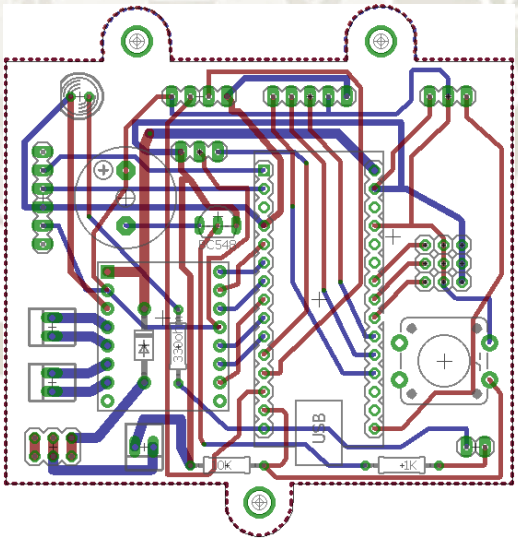


Figura 27. Placa de circuito impreso terminada

Fuente: Elaboración propia.

3.10. Alimentación

3.10.1. Bateria de polímero de litio(LiPo)

La batería que mejor se adecua a nuestros requerimientos es una batería de 2 celdas (2S) debido a que nos brinda un voltaje entre 8.4V (carga completa) y 7.4V (carga baja), cercano al voltaje de funcionamiento de los micromotores y en el rango regulable del AMS1117. Para el módulo se utilizó una batería de la marca ProFuse de 2 celdas que otorga ventajas de tamaño, peso y carga, como se detalla a continuación:

- Capacidad: 850mah
- Voltaje: 7.4V (8.4 carga completa)
- Tasa de descarga: 25C
- Peso: 46 gramos
- Dimensiones: 42x31x18 mm
- Conector: JST

3.10.2. Cálculo de la duración de la batería

Primero debemos asumir que la batería se utilizara para motores (voltaje no regulado) y para alimentar el resto de los componentes electrónicos (regulado a 5V), a continuación, se muestra la Tabla 10 que detalla el consumo de corriente teórico y medido en el módulo.

Tabla 20. Consumo de corriente teórico y medido del módulo

Módulo	Valor Teórico (mA)	Valor Medido (mA)	Error de medición (%)
Micromotores 6V	200	1000	400.0%
Sensor ultrasónico HC-SR04	15	3.2	78.7%
Sensor de línea QTR-3A	50	50.1	0.2%
Bluetooth HC-05	40	43.5	8.8%
Microcontrolador	15	15	0.0%
Buzzer 5v	10	150	1400.0%
Led 5mm	15	14	6.7%
Servomotor SG90	220	150	31.8%
TOTAL	565	1425.8	240.8%

El cálculo para la duración de la batería del módulo se realiza en base a los valores obtenidos de las mediciones recopiladas en la tabla anterior, debido a la gran diferencia que existe al momento de utilizar los micromotores, dando como resultado un valor más aproximado al resultado real.

$$Tiempo\ de\ uso = \frac{Corriente\ de\ bateria}{\frac{Corriente\ de\ carga\ x\ 1000}{60}}$$

$$Tiempo\ de\ uso = \frac{850mA}{\frac{1.425\ x\ 1000}{60}}$$

$$Tiempo\ de\ uso = 36\ minutos$$

El tiempo de uso estimado es de 36 minutos, tomando en cuenta que se destinara parte del tiempo a la programación del módulo y que el valor anterior obtenido es en el caso de utilizar todas las funciones al mismo tiempo, mediante pruebas realizadas anteriormente podemos afirmar que se puede trabajar sin ningún problema con el módulo durante una sesión de 3 horas académicas (2 horas y 15 minutos).

3.11. Diseño y desarrollo del chasis

El software elegido para diseñar las piezas fue Tinkercad de Autodesk, un programa de modelado 3D en línea gratuito que se ejecuta en un navegador web, conocido por su simplicidad y facilidad de uso.

Algunas de las funciones por las cuales se utiliza este software son:

- Diseño sencillo de piezas 3D.
- Ensamblaje sencillo de piezas en el diseño 3D.
- Se ejecuta en un navegador web.
- Licencia gratuita para estudiantes.

El módulo utilizara orugas como sistema de locomoción debido a su precisión y a las ventajas para desplazarse sobre distintas superficies, la empresa Tamiya ofrece un set de orugas que se ajusta a nuestras necesidades, con la ayuda de la impresión 3D se crearon unos adaptadores compatibles con el eje del micromor tipo D de 3mm para poder sujetar las piezas a los motores.

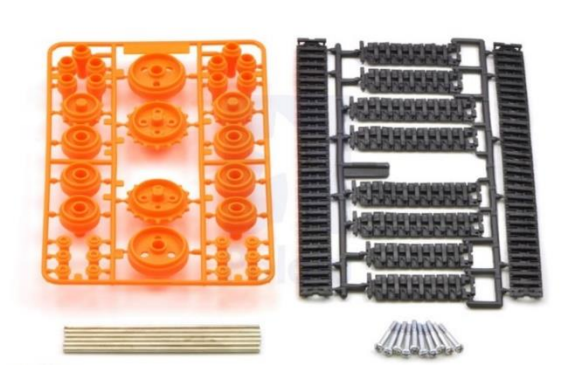


Figura 28. Kit de orugas y ruedas Tamiya 70100 [67]

Se estableció que las dimensiones del módulo debían ser de 10cm de largo por 10cm de ancho como máximo, esto debido a las normas impuestas en algunas competencias de robótica, cumpliendo con esa condición se diseñó una base encargada de contener el sistema de locomoción, la batería y la tarjeta de control.

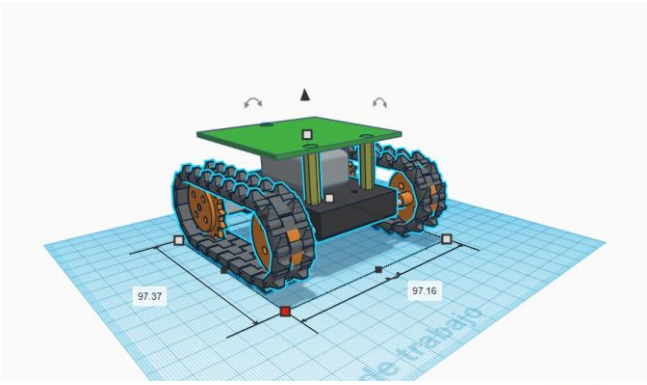


Figura 29. Dimensiones del módulo
Fuente: Elaboración propia.

Para cada aplicación contaremos con soportes diseñados a medida que serán atornillados a la estructura principal, permitiendo cambiar de forma rápida los sensores y la aplicación. A continuación, se detallan algunos de los soportes diseñados hasta el momento:

- Soporte para la aplicación esquiva obstáculos o minisumo, este soporte se monta en la parte delantera del robot y nos permite sujetar un sensor ultrasónico HC-SR04 para detectar objetos delante del módulo.

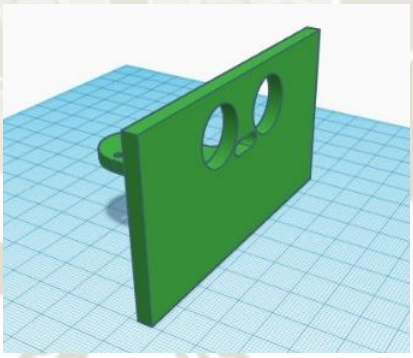


Figura 30. Soporte para el sensor ultrasónico HC-SR04
Fuente: Elaboración propia.

- Soporte para la aplicación seguidor de línea, situado en la parte delantera del robot con orificios para sujetar un sensor QTR-3A y mantenerlo de forma paralela al suelo a una distancia de 3mm (distancia óptima de detección).

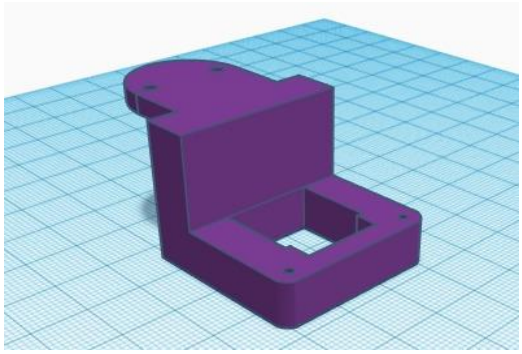


Figura 31. Soporte para el sensor de línea QTR-3A

Fuente: Elaboración propia.

- Soporte para la aplicación soccer o MiroSot, la finalidad de este soporte es poder guiar una pelota de 50mm de diámetro y evitar que las orugas interfieran con el manejo de la misma.

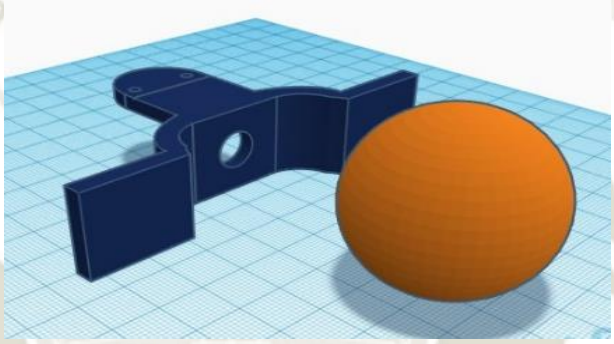


Figura 32. Soporte para la aplicación soccer o MiroSot

Fuente: Elaboración propia.

El software de modelado nos brinda una idea de cómo quedaran los prototipos ya terminados con sus respectivos soportes y permite realizar modificaciones antes de imprimir las piezas con la ayuda de la impresora 3D, a continuación, se presentan las 3 aplicaciones anteriormente mencionadas.

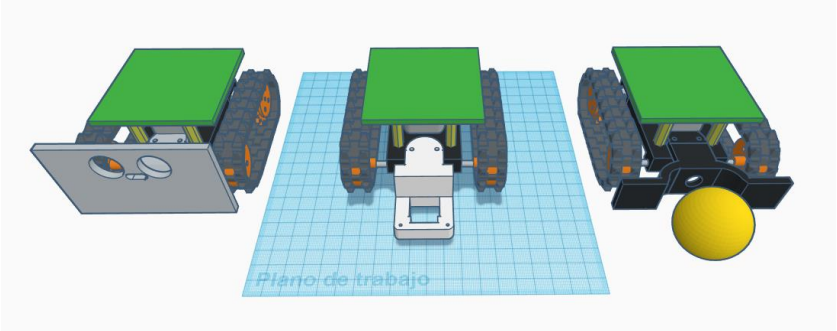


Figura 33. Módulo configurado en sus 3 aplicaciones

Fuente: Elaboración propia.

En total se van a imprimir 6 piezas por modulo en plastico PLA, el cual brindara resistencia a las piezas sin comprometer su peso, adicionalmente se utlizaran tornillos M3 para sujetar los soportes y espaciadores de bronce de 25mm para sujetar la placa de circuito impreso al chasis.

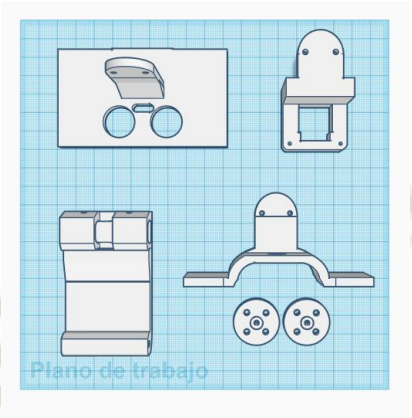


Figura 34. Piezas empleadas en un módulo

Fuente: Elaboración propia.

Luego de imprimir las piezas, realizar el montaje y conectar los sensores mediante cables jumper, los módulos quedaron terminados y listos para ser utilizados. A continuación, se muestran las fotos del ensamblaje final de los módulos.

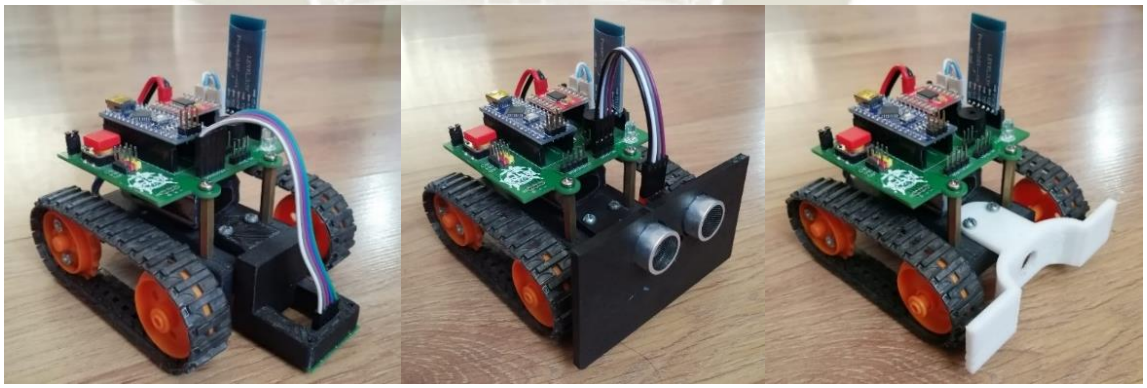


Figura 35. Ensamblaje final de los módulos

Fuente: Elaboración propia.

3.12. Software

3.12.1. Configuración bluetooth

Antes de utilizar el módulo bluetooth, debemos configurarlo adecuadamente utilizando los comandos AT, los cuales permiten configurar algunas funciones por ejemplo, cambiar el nombre del módulo, el código PIN, el Baud Rate, etc.

- AT: Comprueba la conexión, responde con OK.
- AT+VERSION: devuelve la versión del firmware del dispositivo.
- AT+NAME: Cambia el nombre del dispositivo, responde con un OKsetname, está limitado a 20 caracteres.
- AT+PINXXXX: Cambia el PIN de la conexión, responde con OKserPIN, por defecto será 1234.
- AT+BAUDX: Modifica el baud rate del dispositivo, X puede estar entre los siguientes valores:

Tabla 21. Valores de X para modificar el baud rate del módulo bluetooth HC-05

Valor de x	Baud Rate
1	1200
2	2400
3	4800
4	9600 (Defecto)
5	19200
6	38400
7	57600
8	115200
9	230400
A	460800
B	921600
C	1382400

Entendido lo anterior vamos realizamos las conexiones para configurar el HC-05. Para configurar el módulo necesitamos enviar los comandos AT desde una computadora, esto lo podemos hacer entablando la comunicación entre la PC y el

módulo HC-05 de forma indirecta a través de un Arduino, realizando la conexión de la siguiente forma:

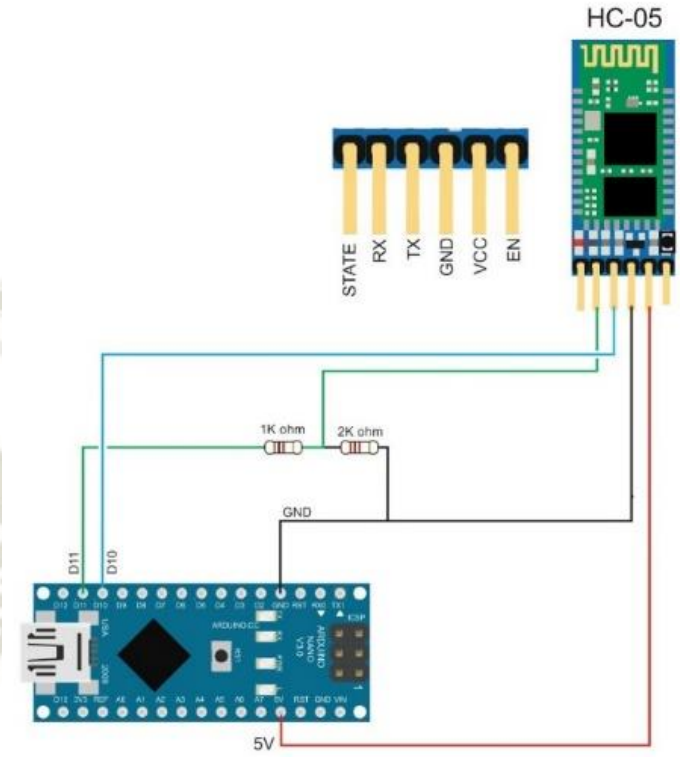


Figura 36. Conexiones del módulo bluetooth con la placa Arduino para su configuración [68]

Luego se procede a compilar y cargar el siguiente sketch, que se encarga de leer los datos enviados de la PC a través de nuestro IDE y se lo envía utilizando la comunicación serial hacia los pines RXD y TXD de nuestro módulo HC-05.

```
#include <SoftwareSerial.h>   // Incluimos la librería  SoftwareSerial
SoftwareSerial BT(10,11);    // Definimos los pines RX y TX del Arduino conectados al Bluetooth

void setup()
{
  BT.begin(9600);       // Inicializamos el puerto serie BT (Para Modo AT 2)
  Serial.begin(9600);   // Inicializamos  el puerto serie
}

void loop()
{
```

```
  if(BT.available())    // Si llega un dato por el puerto BT se envía al monitor serial
  {
    Serial.write(BT.read());
  }

  if(Serial.available())  // Si llega un dato por el monitor serial se envía al puerto BT
  {
     BT.write(Serial.read());
  }
}
```

Para entrar al modo AT 2, antes de alimentar o encender el módulo es necesario mantener presionado el botón con el que cuenta el módulo y alimentar, después que enciende podemos soltar el botón. Si el LED parpadea lentamente podemos saber que ingresamos al Modo AT 2. El siguiente diagrama ilustra los pasos que debemos seguir para poder configurar de forma correcta el módulo bluetooth.

Figura 37. Diagrama de flujo para la configuración del módulo bluetooth

Fuente: Elaboración propia.

Ahora abrimos nuestro Monitor serial IDE de Arduino. En la parte inferior debemos escoger “Ambos NL & CR” y la velocidad “38400 baud” (la velocidad para comunicarse en el MODO AT 2)

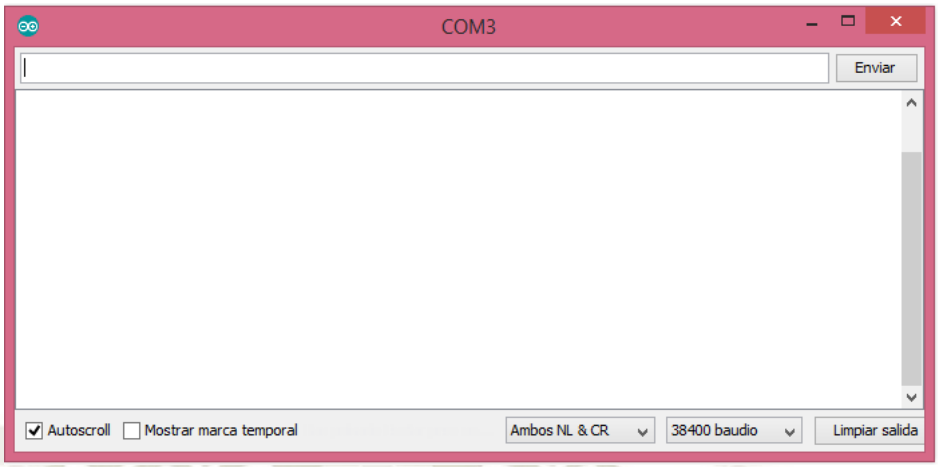


Figura 38. Monitor serial IDE de Arduino

Fuente: Elaboración propia.

Lo primero es comprobar si nuestro bluetooth responde a los comandos AT, para eso realizamos un test de comunicación enviando el comando AT:

Enviar: AT

Recibe: OK

Si recibimos como respuesta un OK significa que ingresamos al modo de configuración AT 2, en caso no aparezca ese mensaje debemos verificar las conexiones o los pasos anteriores.

Por defecto nuestro bluetooth se llama “HC-05”, vamos a nombrar cada módulo como “BOT-(numero)” esto se puede cambiar con el siguiente comando AT:

Enviar: AT+NAME=BOT-1

Respuesta: OK

Por defecto viene con el código de vinculación (Pin) “1234”, vamos a cambiar la contraseña por “0000” utilizando el siguiente comando AT:

Enviar: AT+PSWD="0000"

Respuesta: OK

La velocidad por defecto es de 9600 baudios, con Stop bit =0 (1 bit de parada) y sin paridad, esos parámetros se ajustan a nuestros requerimientos, en caso se desee cambiarlos, se puede utilizar el siguiente comando AT:

Sintaxis: AT+UART=<Baud>,< StopBit>,< Parity>

Enviar: AT+UART=9600,0,0

Respuesta: OK

Donde:

- < Baud > equivale a una velocidad, los valores pueden ser: 4800, 9600, 19200, 38400, 57600, 115200, 23400, 460800, 921600 o 1382400.
- < StopBit> es el Bit de parada, puede ser 0 o 1, para 1 bit o 2 bits de parada respectivamente, Para aplicaciones comunes se trabaja con 1 bit por lo que este parámetro normalmente se lo deja en 0.
- < Parity> Es la paridad, puede ser 0 (Sin Paridad), 1 (Paridad impar) o 2 (Paridad par). Para aplicaciones comunes no se usa paridad, por lo que se recomienda dejar este parámetro en 0.

3.12.2. Código base

El módulo cuenta con distintas aplicaciones, para ello debemos generar funciones compatibles con los sensores y actuadores que vamos a utilizar, que servirán a los alumnos como código base, permitiéndoles luego modificar la programación de acuerdo a la tarea requerida.

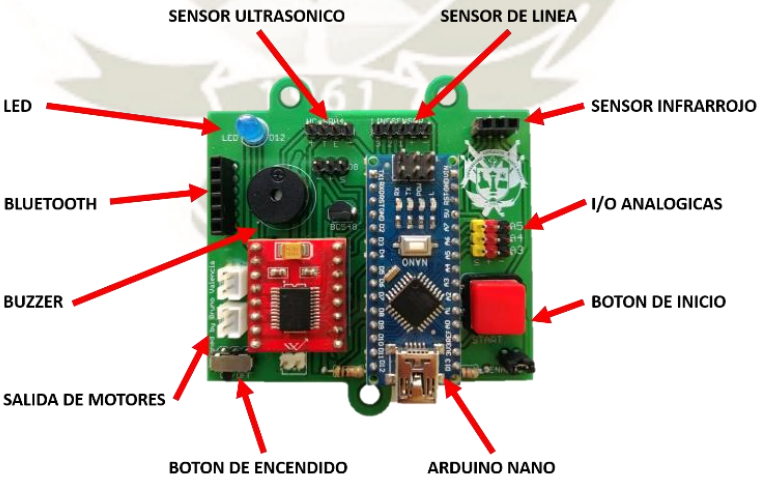


Figura 39. Componentes de la placa del módulo

Fuente: Elaboración propia.

Primero debemos declarar las variables que vamos a utilizar, muy independientemente de la función que cumplirá el módulo, se declararan todas las variables necesarias para cada una de las funciones definidas, esto permitirá trabajar más rápido al momento de realizar el código, los tipos de variables serán:

- Variables para el control de motores
- Variables para lectura de sensores
- Variables para utilizar los actuadores integrados en la placa
- Variables globales para almacenar valores

Con la ayuda de los comentarios en el código "//" vamos a diferenciar cada una de ellas.

```
//SERVOMOTOR
#include <Servo.h>
Servo servo;

//MOTOR DERECHA
int PWMA = 3; //Control de velocidad
int AIN1 = 4; //Dirección
int AIN2 = 2; //Dirección

//MOTOR IZQUIERDA
int PWMB = 5; //Control de velocidad
int BIN1 = 6; //Dirección
int BIN2 = 7; //Dirección

//SENSOR ULTRASONICO
int echo = 9;
int trig = 10;
int tiempo, distancia, lectura;  //Variables para el cálculo de distancia

//SENSOR DE LINEA
int SensorIzq  = A0;
int SensorCen = A1;
int SensorDer = A2;
int Sizq = 0;
int Scen = 0;
```

```
int Sder = 0;
int linea = 400;  //Variable para definir el tono de la linea

//INTEGRADOS EN LA PLACA
int pulsador=11;
int estado=0;
int led=12;
int ldr=13;
```

La siguiente parte del programa se denomina void setup() y se encarga de especificar al microcontrolador los comandos que ejecutará en el momento del arranque, estos comandos sólo se ejecutarán una vez al inicio del sistema y pueden realizar las siguientes funciones:

- pinMode con el número y tipo de pin. Esta línea define el modo de operación de los pines de Arduino.
- Serial.begin con indicación de velocidad (la mayoría de las veces 9600). Esta instrucción inicializa la operación del puerto serie a la velocidad especificada.
- Instrucciones sobre cómo conectar e inicializar varios objetos de la librería arduino. Por ejemplo, servo.atach(8) indicará a la biblioteca que hemos conectado el servo drive al pin 8, y todas las acciones posteriores del código de la biblioteca se realizarán con este puerto.
- Inicialización de variables globales si por alguna razón no podemos hacerlo al definir las variables en el campo de visión global.

```
void setup() {
  Serial.begin(9600);
  pinMode(PWMA, OUTPUT);
  pinMode(AIN1, OUTPUT);
  pinMode(AIN2, OUTPUT);
  pinMode(PWMB, OUTPUT);
  pinMode(BIN1, OUTPUT);
  pinMode(BIN2, OUTPUT);
  pinMode(pulsador, INPUT);
  pinMode(ldr, INPUT);
  pinMode(led, OUTPUT);
```

```
  pinMode(echo, INPUT);
  pinMode(trig,OUTPUT);
  servo.attach(8);
}
```

La función de bucle void loop() es la función principal, ahí colocaremos los comandos que se ejecutarán mientras la placa Arduino esté habilitada. Comenzando con el primer comando, el microcontrolador irá hasta el final y saltará inmediatamente al principio para repetir la misma secuencia, ejecutando un bucle infinito.

Esta parte del programa se encargará de definir la aplicación de nuestro modulo, ya sea seguidor de líneas, esquiva obstáculos o permitirnos controlar el led integrado para que se encienda cada vez que presionamos el pulsador como el siguiente ejemplo:

```
void loop()
{
  lectura = digitalRead(pulsador);     //Lectura del estado del BOTON
  if (lectura == HIGH)                    //Si el BOTON es presionado
  {digitalWrite(led, HIGH);}              //Encendemos el LED
  else                                    //Si el BOTON no está siendo presionado
  {digitalWrite(led, LOW);}               //Apagamos el LED
 }
```

Las funciones de usuario pueden ser escritas para realizar tareas repetitivas y para reducir el tamaño de un programa. Segmentar el código en funciones permite crear piezas de código que hacen una determinada tarea y volver al área del código desde la que han sido llamadas.

Vamos a generar funciones que permitan a los alumnos utilizar de una manera más amigable los sensores y actuadores asignados anteriormente, sin dejar de lado la posibilidad del alumno para realizar optimizaciones de las mismas según su criterio.

Generalmente las funciones se almacenan en un fichero separado, en este caso vamos a almacenarlas en el mismo código, eso nos va a permitir poder realizar modificaciones de forma más práctica, además la mayoría de funciones ocupan pocas líneas de código, como se detalla más adelante.

Las primeras funciones que vamos a declarar son las que van a permitir al robot desplazarse en distintas direcciones utilizando los micromotores, para eso es necesario elaborar una tabla que nos permite entender el funcionamiento de nuestro driver de motores, el cual nos brinda 2 pines de dirección (señal digital) y 1 pin de velocidad (señal pwm) por motor.

Tabla 22. Configuraciones de los motores

	MOTOR DERECHA			MOTOR IZQUIERDA		
	AIN1	**AIN2**	**PWMA**	**BIN1**	**BIN2**	**PWMB**
AVANZAR	HIGH	LOW	1-254	HIGH	LOW	1-254
RETROCEDER	LOW	HIGH	1-254	LOW	HIGH	1-254
DERECHA	HIGH	LOW	1-254	LOW	HIGH	1-254
IZQUIERDA	LOW	HIGH	1-254	HIGH	LOW	1-254
FRENAR	LOW	LOW	254	LOW	LOW	254

Sabiendo esos parámetros, procederemos a desarrollar la función AVANZAR con la sintaxis adecuada para cargarla en el código, se trabajará de la misma forma con el resto de funciones.

```
void avanzar()
{
   digitalWrite(AIN1, HIGH);
   digitalWrite(AIN2, LOW);
   analogWrite(PWMA, 150);
   digitalWrite(BIN1, HIGH);
   digitalWrite(BIN2, LOW);
   analogWrite(PWMB, 150);
}
```

Por último, controlaremos un servomotor conectado al pin 8 de nuestro modulo, el cual puede girar 180 grados y servirá para ampliar el campo de vista del sensor ultrasónico o acoplar una pinza para sujetar objetos, para controlarlo utilizaremos pulsos generados con la ayuda de una biblioteca llamada Servo.h instalada por defecto en el IDE de Arduino, la cual nos permite crear un objeto llamado servo e ingresar los grados de desplazamiento del mismo, sin la necesidad de preocuparnos por los pulsos que debemos generar para realizar dicha tarea.

```
#include <Servo.h>
Servo servo;                //Creamos el objeto servo
void setup()
{
  servo.attach(8);          //Definimos el pin donde estará conectado
}
void loop()
{
servo.write(0);             //Desplazamos el servomotor a la posición 0°
delay(500);
servo.write(180);           //Desplazamos el servomotor a la posición 180°
delay(500);
}
```

3.12.3. Código seguidor de línea

La función de esta aplicación es la de seguir una línea negra sobre un fondo blanco, para esto utilizaremos el sensor de línea QTR-3A, procederemos a elaborar el diagrama de flujo en función a las señales que nos brindaran los 3 sensores de línea.

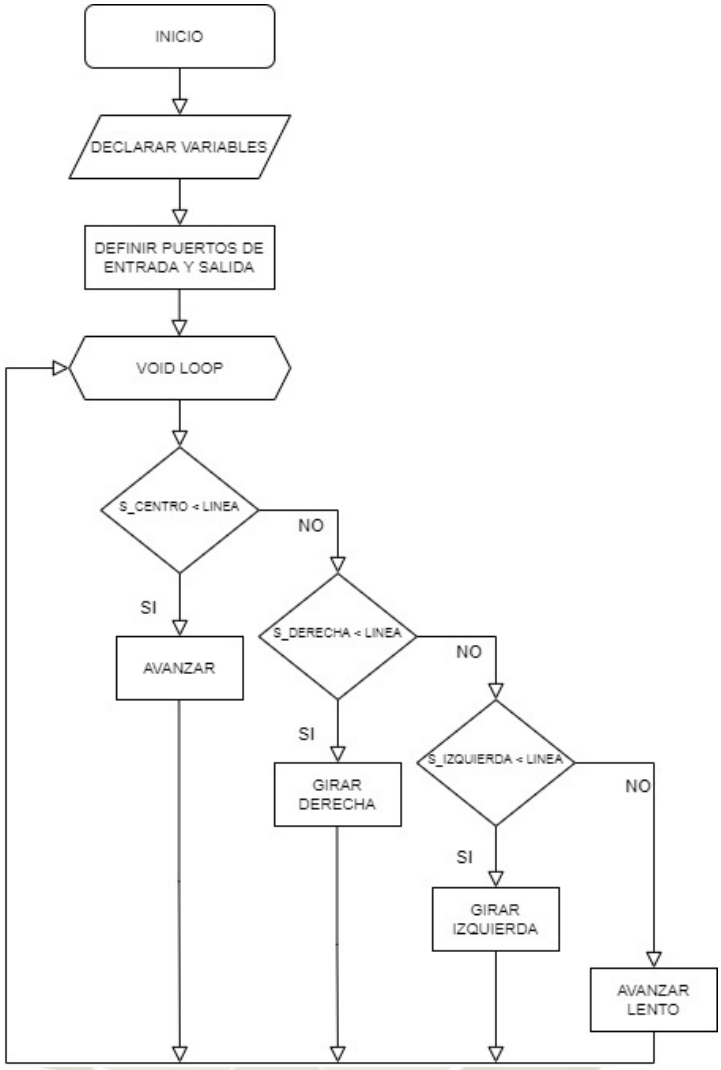


Figura 40. Diagrama de flujo del seguidor de línea

Fuente: Elaboración propia.

Sera necesario elaborar una función denominada void sensores(), la cual se encargará de acondicionar las señales analógicas brindadas por los sensores de línea, los sensores ópticos utilizados en el arreglo de sensores QTR-3A son el modelo GP2S60, dichos sensores cuentan con un tiempo de respuesta típico de 40µs y máximo de 200µs, según datasheet.

Utilizaremos la instrucción delayMicroseconds() para agregar un retardo de 250µs, con la finalidad de evitar errores en la detección de la línea y devolver 3 variables (Sizq, Sder y Scen) con las que los alumnos podrán trabajar.

```
void sensores()
{
 Sizq = analogRead(SensorIzq);
 delayMicroseconds(250);
 Sder = analogRead(SensorDer);
 delayMicroseconds(250);
 Scen = analogRead(SensorCen);
 delayMicroseconds(250);
}
```

Una vez definida la función void sensores() podemos invocarla en el void loop() y trabajar con las variables que devuelve haciendo uso de las condicionales “if” y “else”.

El código terminado para la aplicación seguidor de línea quedaría de la siguiente forma, es importante recordar que las declaraciones de variables, los datos del void setup() y las funciones para el desplazamiento del robot no fueron modificadas.

```
void loop()
{
   sensores();
   if(Sizq<=linea)    {izquierda();delay(1);}
   if(Scen<=linea)   {avanzar();delay (1);}
   if(Sder<=linea)    {derecha();delay (1);}
   else          {avanzar_lento();delay (1);}
}

void sensores()
{
 Sizq = analogRead(SensorIzq);
 delayMicroseconds(250);
 Sder = analogRead(SensorDer);
 delayMicroseconds(250);
 Scen = analogRead(SensorCen);
 delayMicroseconds(250);
}
```

3.12.4. Código esquiva obstáculos

La función de esta aplicación es la de detectar un objeto situado delante del robot, para esto utilizaremos el sensor ultrasónico HC-SR04, procederemos a elaborar el diagrama de flujo en función a la señal que nos brindara el sensor.

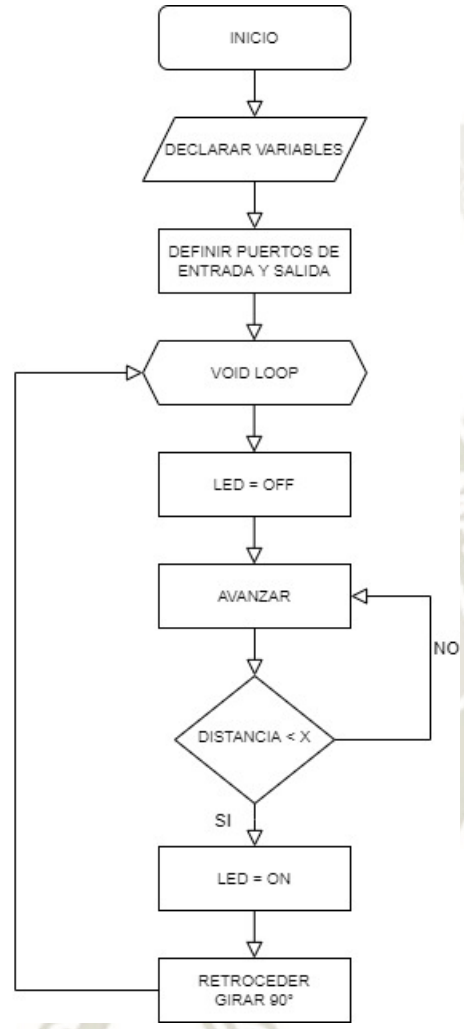


Figura 41. Diagrama de flujo del módulo esquiva obstáculos

Fuente: Elaboración propia.

Para la función void ultrasonico() empezamos enviando un pulso de 10us al trigger del sensor, seguidamente recibimos el pulso de respuesta del sensor por el pin echo, para medir el pulso usamos la función pulseIn(*pin*, *value*). La variable tiempo, almacena el tiempo que demora en llegar el eco del ultrasonido, el siguiente paso es calcular la distancia entre el sensor ultrasónico y el objeto, para eso nos guiamos de la siguiente formula:

$$\frac{340m}{2} \times \frac{1s}{1000000us} \times \frac{100cm}{1m} = \frac{2d}{t}$$

$$d(cm) = \frac{t(us)}{59}$$

La finalidad de la función void ultrasonico() es la de brindar la distancia en centímetros y almacenarla en un variable que luego la podamos utilizar en el código, a continuación se detalla la sintaxis.

```
void ultrasonico()
{
    digitalWrite(trig, LOW);              //Apaga trigger por 2us
    delayMicroseconds(2);
    digitalWrite(trig, HIGH);             //Genera el pulso de trigger por 10us
    delayMicroseconds(10);
    digitalWrite(trig, LOW);
    tiempo = pulseIn(echo, HIGH);         //Lee el tiempo del Echo
    distancia = tiempo/59;                //Calcula la distancia en centímetros
}
```

Una vez definida la función void ultrasonico() podemos invocarla en el void loop() y trabajar con las variables que devuelve haciendo uso de las condicionales “if” y “else”.

El código terminado para la aplicación esquiva obstáculos quedaría de la siguiente forma, es importante recordar que las declaraciones de variables, los datos del void setup() y las funciones para el desplazamiento del robot no fueron modificadas.

```
void loop()
{
   ultrasonico();
   if (distancia <=x)
       {digitalWrite(led, HIGH);retroceder();delay(80);derecha();delay(200);}
   else
       {avanzar();digitalWrite(led, LOW);delay(1);}
  }

void ultrasonico()
{
    digitalWrite(trig, LOW);              //Apaga trigger por 2us
```

```
    delayMicroseconds(2);
    digitalWrite(trig, HIGH);              //Genera el pulso de trigger por 10us
    delayMicroseconds(10);
    digitalWrite(trig, LOW);
    tiempo = pulseIn(echo, HIGH);          //Lee el tiempo del Echo
    distancia = tiempo/59;                 //Calcula la distancia en centímetros
}
```

3.12.5. Código control bluetooth

Esta función permite controlar el robot mediante una aplicación bluetooth instalada en un celular, para esto utilizaremos el módulo bluetooth HC-05, procederemos a elaborar el diagrama de flujo en función a la señal que nos brindara el sensor.

Figura 42. Diagrama de flujo control de bluetooth

Fuente: Elaboración propia.

El void loop() será el encargado de interpretar la información recibida y transformarla en acciones, para eso es necesario establecer el emparejamiento bluetooth, una vez realizado el emparejamiento se generaran instrucciones dependientes de los comandos que se envíen mediante la aplicación bluetooth, por ejemplo si se presiona la flecha para avanzar, se enviara un carácter que será interpretado por una función específica enviando una señal a los motores para que el módulo se desplace hacia adelante y así sucesivamente con el resto de acciones definidas por el alumno.

Es importante recordar que las declaraciones de variables, los datos del void setup() y las funciones para el desplazamiento del robot no fueron modificadas.

```
void loop()
{
  if(Serial.available()>0){       // lee el bluetooth y almacena en estado
     estado = Serial.read(); }

if(estado=='c'){parar();}                    // Parar
if(estado=='a'){derecha();}                  // Avanzar
if(estado=='e'){izquierda();}                // Reversa
if(estado=='b'){adelante();}                 // Izquierda
if(estado=='d'){atras();}                    // Derecha
if (estado =='j'){digitalWrite(LED, HIGH);}  // Activar bocina
}
```

Como podemos apreciar, solo será necesario modificar el void loop() para definir el comportamiento o aplicación del módulo, esto permitirá a los alumnos migrar de aplicación de forma fácil o inclusive utilizar distintas aplicaciones al mismo tiempo, por ejemplo migrar de soccer a seguidor de línea con solo enviar un comando mediante bluetooth.

3.12.6. Creación de la aplicación bluetooth

Para la elaboración de la aplicación bluetooth utilizaremos la herramienta online MIT app inventor, la cual nos permite realizar dos funciones muy importantes que son diseñar la interfaz de usuario y programar al comportamiento de la aplicación juntando bloques.

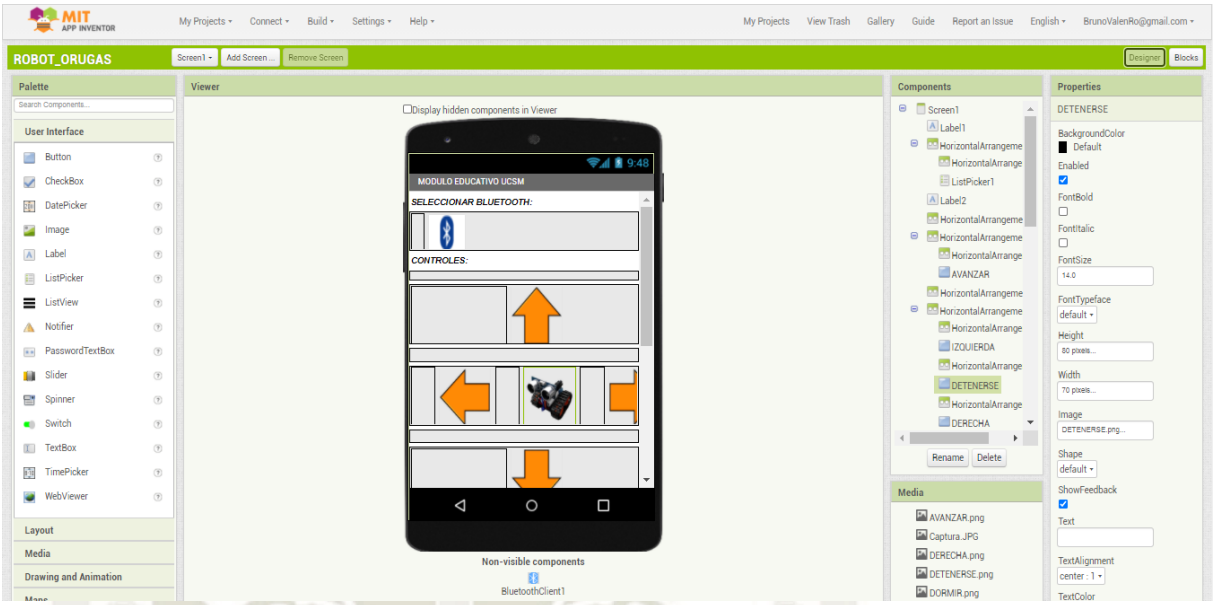


Figura 43. Entorno de diseño de MIT app inventor
Fuente: Elaboración propia.

Es necesario definir las instrucciones que serán enviadas al módulo, para que en función a ellas podamos diseñar la interfaz de usuario. Las siguientes instrucciones forman parte de una aplicación genérica que le permita al módulo ejecutar las funciones anteriormente desarrolladas:

- **Seleccionar bluetooth:** Se encargará de enlazar nuestra aplicación con el módulo bluetooth conectado en el robot.
- **Instrucciones de desplazamiento:** Permitirán al módulo avanzar, retroceder, girar a la izquierda, girar a la derecha y detenerse.
- **Llamado a funciones personalizadas:** Consta de dos botones sin una función específica asignada, los cuales permitirán al alumno ejecutar cualquier función o rutina elaborada por ellos, por ejemplo, reproducir una melodía con el buzzer o desplazar el módulo en una forma determinada.
- **Llamado a funciones preestablecidas:** Permitirán activar las funciones anteriormente desarrolladas como son seguidor de línea y esquivar obstáculos.

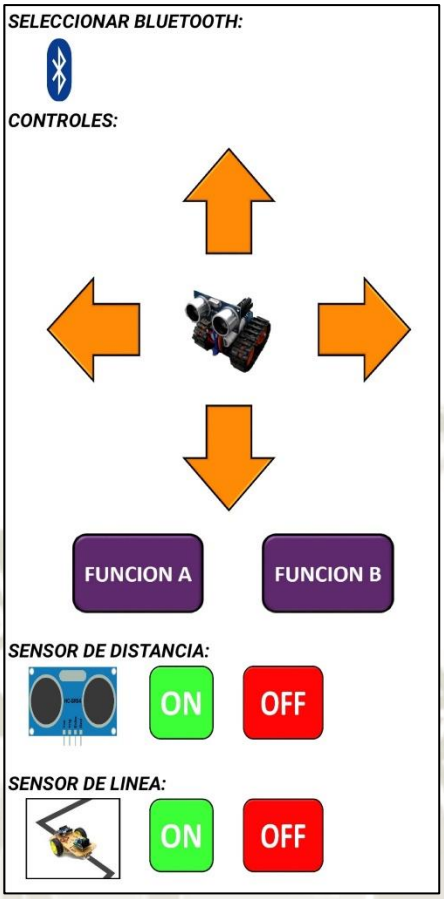


Figura 44. Interfaz de usuario de la aplicación bluetooth

Fuente: Elaboración propia.

Una vez definida la interfaz de usuario, seleccionaremos la pestaña de bloques para programar el comportamiento de los botones y definir los comandos que estos enviaran al módulo. Primero es necesario establecer la conexión con el cliente bluetooth para iniciar la comunicación, luego se crearán bloques para cada botón a los cuales se asignará un carácter del alfabeto, el cual al ser recibido por el módulo será interpretado y ejecutará una función determinada.

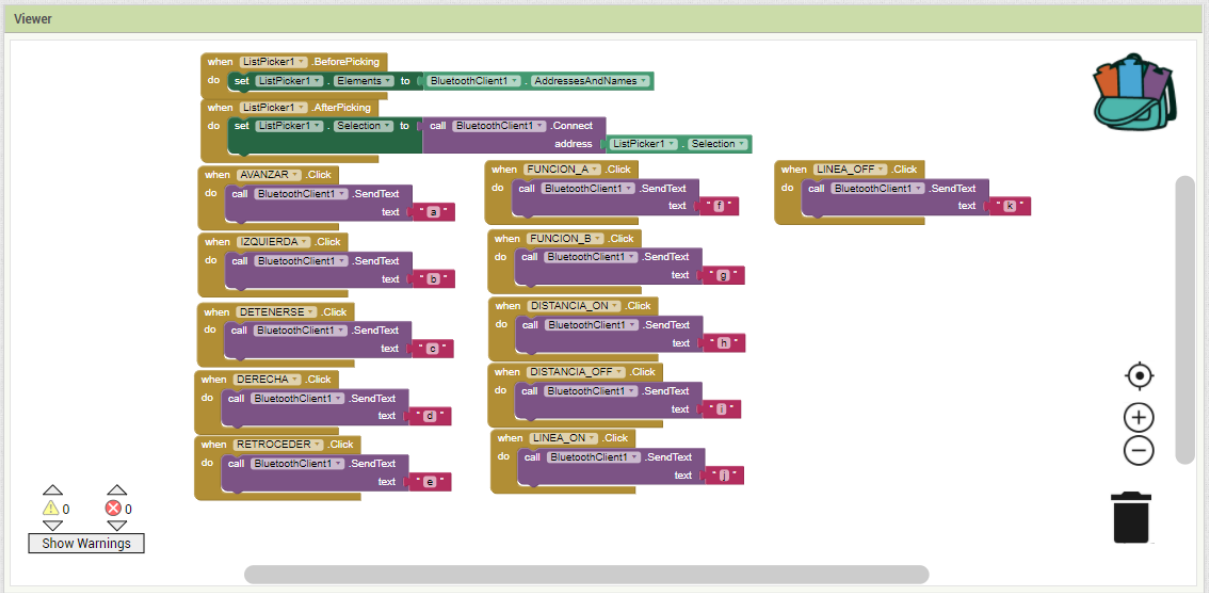


Figura 45. Interfaz de bloques de la aplicación bluetooth

Fuente: Elaboración propia.

Teniendo las dos partes de la aplicación bluetooth completadas, se procede a la construcción de la aplicación android con la extensión ".apk", para eso nos dirigimos a la pestaña "Build" en el entorno de diseño de MIT app inventor y contamos con dos opciones; la opción de generar un código QR para escanearlo con un celular o descargar el archivo e instalarlo de forma manual en el celular.

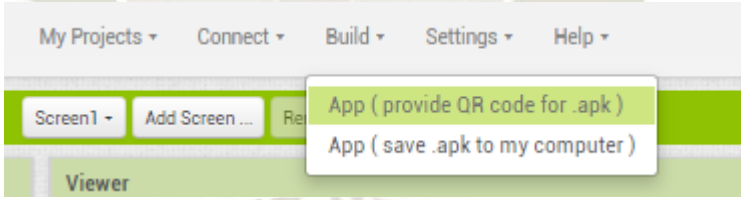


Figura 46. Pestaña de construcción de la aplicación bluetooth

Fuente: Elaboración propia.

Es importante recalcar que todo el progreso se almacena en la aplicación online, permitiendo el acceso desde cualquier lugar con acceso a internet y así poder seguir escalando la aplicación en función a las futuras aplicaciones que se deseen implementar.

CAPITULO IV: RESULTADOS OBTENIDOS, PRUEBAS Y OPTIMIZACION

4.1 Seguidor de líneas

Con la teoría de funcionamiento expuesta en el capítulo anterior y utilizando de referencia el código brindado, se procedió a probar el módulo en un entorno de pruebas real, en este caso definimos 2 escenarios concretos:

- Primer escenario: Detectar una línea negra elaborada con una cinta aislante situada sobre una mesa o carpeta, esta situación es muy común, ya que los alumnos al momento de programar el módulo se ven en la necesidad de realizar cambios a algunas instrucciones del código en función a la lógica que deseen trabajar y tener el módulo situado sobre su escritorio les permite un acceso rápido al ordenados y permite cargar el código de una manera más veloz.
- Segundo escenario: Detectar una línea negra impresa sobre una pieza de lona, esta situación se presenta en la mayoría de competencias de robótica y presenta un desafío para los alumnos debido a que no existe una normativa exacta para los parámetros de impresión de la línea, es por ello que los alumnos deben realizar una medición de los valores que recibe el sensor al ser situado sobre la línea y así poder establecer los valores más óptimos para trabajar con el módulo.

4.1.1 Lecturas obtenidas en el entorno

Para depurar el código fue necesario utilizar una función que nos brinda el IDE de Arduino, la cual nos permite visualizar datos por el puerto serial, en este caso necesitábamos ver los valores que brindaba el sensor de línea al momento de ser colocado sobre la línea negra y sobre la superficie de contraste.

Esto se resolvió utilizando un cable USB de 1.5 metros y una laptop con el IDE de Arduino, la siguiente sintaxis de código fue la empleada para visualizar la lectura de los 3 sensores con los que contaba el módulo:

```
void loop()
{
  sensores();
  Serial.print("IZQ: ");
  Serial.print(Sizq);
  Serial.print(" ");
  Serial.print("CEN: ");
  Serial.print(Scen);
  Serial.print(" ");
  Serial.print("DER: ");
  Serial.println(Sder);
}

void sensores ()
{
 Sizq = analogRead(SensorIzq);
 delayMicroseconds(250);
 Sder = analogRead(SensorDer);
 delayMicroseconds(250);
 Scen = analogRead(SensorCen);
 delayMicroseconds(250);
}
```

Una vez cargado el código en el módulo, se puede apreciar que solo se realiza la acción de lectura a través del sensor de línea y los valores son mostrados en el monitor serial, tanto de la línea negra como de la superficie de contraste (mesa o lona), la ventaja de este tipo de código es que solo es necesario identificar el valor de lectura de la línea negra para poder asignarlo a la variable “línea” y poder generar las instrucciones en función a ese valor.

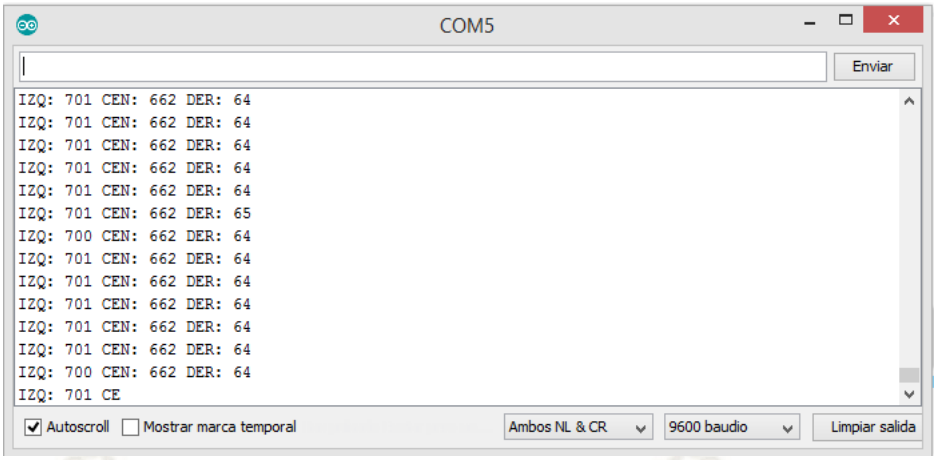


Figura 47. Lectura de línea elaborada con cinta aislante sobre una mesa

Fuente: Elaboración propia.

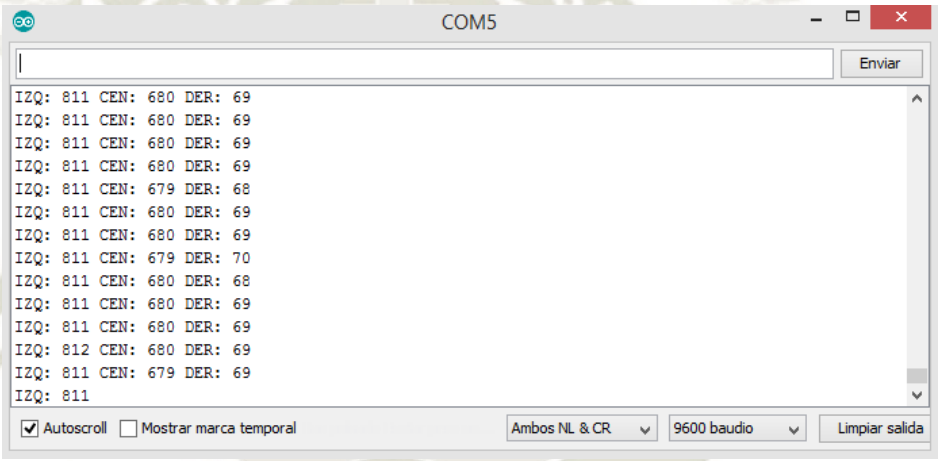


Figura 48. Lectura de línea impresa sobre lona blanca

Fuente: Elaboración propia.

4.1.2 Ajustes recomendados

Realizadas las pruebas del código básico y las mediciones de los valores brindados por el sensor, se determinaron 3 mejoras a realizar en el código:

- Añadir una instrucción de inicio vinculada al botón integrado en la placa, para permitir iniciar el módulo cuando el jurado o profesor indique, cumpliendo con las normas de algunos concursos y también brindando más control sobre el módulo.
- Emplear instrucciones más restrictivas con la ayuda del operador Booleano “AND” el cual solo es cierto cuando ambas sentencias cumplen la condición, permitiendo elaborar condiciones cuando 2 sensores

detectan la línea al mismo tiempo o inclusive los 3 sensores, resultando en un desempeño mayor del módulo.

- Modificar el tipo de giro del módulo, ya que implementa por defecto el autogiro, en el que ambas ruedas giran en sentido contrario permitiendo al módulo girar en lugares de espacio reducido, es recomendable que un robot seguidor de líneas siempre se desplace hacia adelante, es por eso que el tipo de giro aplicado será el de pivote, el cual mantiene una rueda detenida mientras que la otra describe el arco que define el giro.

4.2 Esquiva obstáculos

Esta aplicación está centrada en torno a la lectura que nos brinda el sensor de ultrasonido, la ventaja de este sensor es que no se ve afectado por ruido u ondas de luz, pero una desventaja es que no pueden diferenciar objetos pequeños y grandes, entonces será necesario comprobar los siguientes puntos.

- Precisión de la medición brindada por el sensor.
- Sensibilidad del sensor para encontrar un objeto.

4.2.1 Lecturas obtenidas en el entorno

Para realizar las pruebas fue necesario utilizar la función que nos brinda el IDE de Arduino, la cual nos permite visualizar datos por el puerto serial, en este caso necesitábamos ver los valores que brindaba el sensor ultrasónico al momento de detectar un objeto en distintas condiciones.

```
void loop()
{
   ultrasonico();
  Serial.print("Distancia: ");
  Serial.println(distancia);
}

void ultrasonico()
{
    digitalWrite(trig, LOW);
    delayMicroseconds(2);
```

```
    digitalWrite(trig, HIGH);              // genera el pulso de trigger por 10us
    delayMicroseconds(10);
    digitalWrite(trig, LOW);
    tiempo = pulseIn(echo, HIGH);          // Lee el tiempo del Echo
    distancia = tiempo/59;                 // calcula la distancia en centimetros
}
```

Para confirmar la precisión de la lectura brindada por el sensor, vamos a situar un objeto a una distancia de 45 cm delimitada por una cinta métrica y contrastaremos los valores obtenidos de la variable “distancia”.

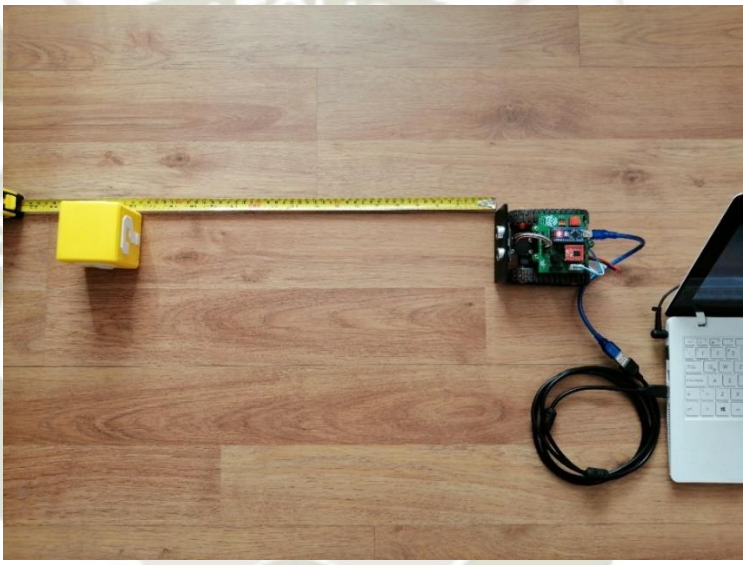


Figura 49. Prueba de detección de objeto a 45 cm

Fuente: Elaboración propia.

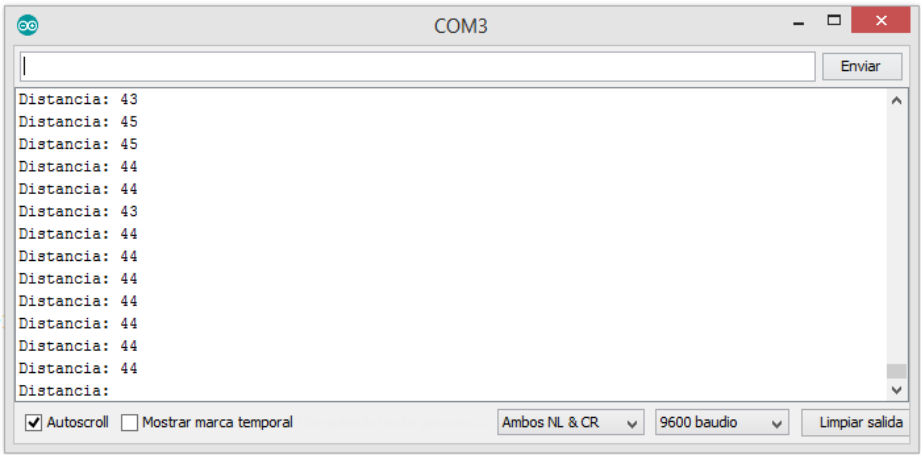


Figura 50. Lectura del sensor ultrasónico

Fuente: Elaboración propia.

Es importante determinar el tamaño mínimo de un objeto perceptible por el sensor para tomar en cuenta como limitación al momento de realizar una prueba o tarea asignada por el tutor.

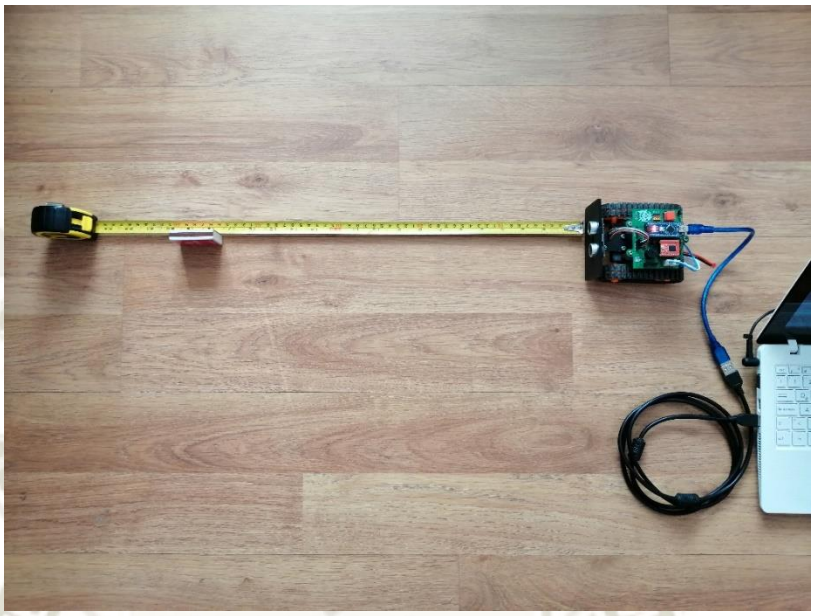


Figura 51. Prueba de sensibilidad de detección de objeto a 45 cm
Fuente: Elaboración propia.

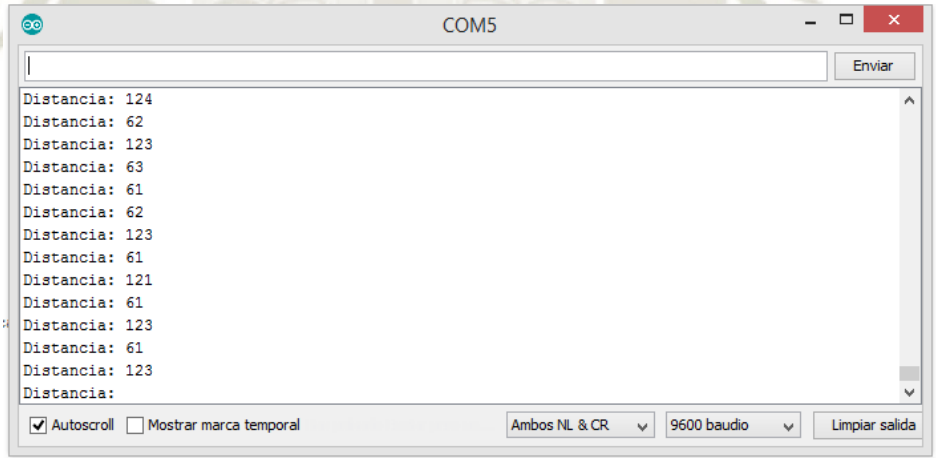


Figura 52. Lectura del sensor ultrasónico
Fuente: Elaboración propia.

Como se puede apreciar al momento de detectar una caja de fósforos situada de manera lateral, se obtiene una medida intermitente y con ruido, además este valor solo se obtiene cuando el objeto se sitúa al medio exacto del sensor, es por eso que se estableció un objeto de 5 x 5 cm, como el objeto más pequeño con

el que se puede trabajar, debido a que esa medida concuerda con las dimensiones del sensor ultrasónico.

4.2.2 Ajustes recomendados

Es importante tener en cuenta todos estos parámetros al momento de aplicar la detección de objetos del módulo:

- Los valores brindados por el sensor son confiables respecto a la medida real.
- La detección de objetos funciona de mejor manera al tratarse de un objeto plano y de dimensiones mayores a 5cm, permitiendo que la señal ultrasónica rebote de mejor forma.
- Se puede modificar la función para trabajar el sensor como un detector de presencia, permitiendo una lectura más veloz de lo valores, pero anulando la posibilidad de determinar la distancia a la que se encuentra el objeto.

4.3 Control bluetooth

Para esta aplicación utilizamos el módulo bluetooth HC-05 configurado como esclavo, el cual permite al alumno tener control sobre el módulo, para el correcto funcionamiento del mismo, será importante evaluar algunos aspectos como:

- Comprobar el rango efectivo de la comunicación bluetooth.
- Determinar la latencia al enviar una señal desde la aplicación hasta la ejecución en el módulo.

4.3.1 Lecturas obtenidas en el entorno

Ya que el módulo bluetooth está conectado directamente a los pines de transmisión y recepción serial nativos de Arduino, vamos a utilizar la librería “SoftwareSerial.h” para poder habilitar otros pines del microcontrolador para la comunicación bluetooth, dejando los pines nativos libres para la comunicación con la pc y permitiendo así visualizar mediante el monitor serie los comandos enviados por la aplicación celular.

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(9, 10); // RX, TX

void setup() {
  Serial.begin(57600);
  while (!Serial) { ; }
  mySerial.begin(9600);
}

void loop() {
  if (mySerial.available()) {
    Serial.println(mySerial.read());
  }
}
```

Se definieron los pines 9 y 10 (pines asignados al sensor ultrasónico) como recepción y transmisión respectivamente, se cargó el programa y se procedió a realizar la prueba de la distancia máxima de comunicación vía bluetooth, presionando el botón “avanzar” de la aplicación celular, el cual envía el carácter “a” cómo se puede apreciar en el monitor serie, el cual fue configurado a 57600 baudios para no interferir con la comunicación bluetooth.

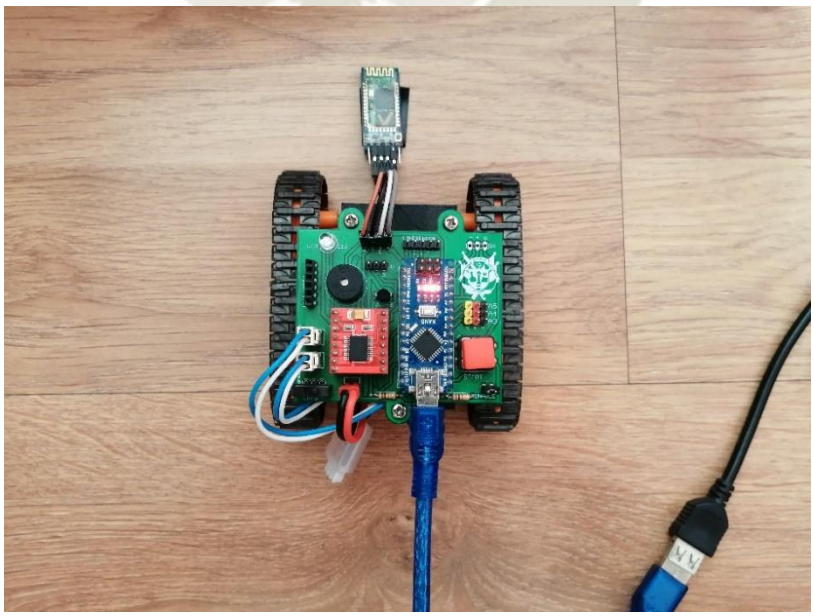


Figura 53. Conexión del módulo bluetooth al puerto ultrasónico

Fuente: Elaboración propia.

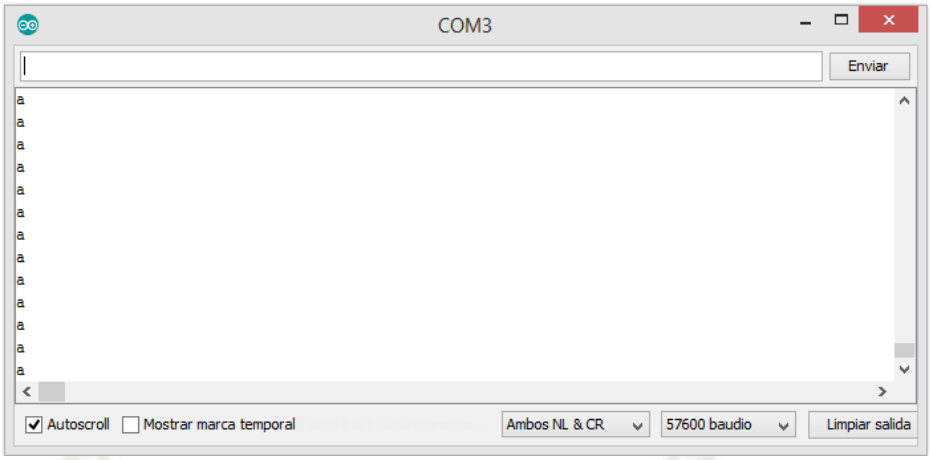


Figura 54. Lectura obtenida del módulo bluetooth

Fuente: Elaboración propia.

Realizadas las pruebas, se demostró que el rango efectivo en un ambiente cerrado (simulando un salón) y con una línea de vista directa sin obstáculos entre el módulo y el celular, es entre 7 y 9 metros, con algunas intermitencias en la señal, así que se definió 7 metros como la distancia máxima para trabajar con este módulo.

4.3.2 Ajustes recomendados

Realizadas las pruebas se determinó respetar estos parámetros al momento de controlar el módulo mediante bluetooth:

- Cuanto mayor es la velocidad en baudios, más rápido se envían y reciben los datos, pero existen límites en cuanto a la rapidez con la que se pueden transferir los datos. Se recomienda no exceder los 115200 baudios, de no ser así comenzará a ver errores en el extremo receptor, ya que los relojes y los períodos de muestreo simplemente no pueden mantenerse al día.
- Al momento de desarrollar la app bluetooth se recomienda utilizar botones “TouchUp” y “TouchDown”, estos simulan un botón real y solo funcionan al mantenerse presionados, cuando el módulo se aleje del rango de la señal bluetooth, este dejara de desplazarse y esperara a establecer la conexión, evitando así percances al momento que el alumno controle el módulo.

- Tomar en consideración el rango efectivo del módulo al momento de realizar una práctica, ya que, una vez perdida la conexión bluetooth, se debe volver a enlazar el dispositivo, afectando el desempeño o causando penalidades en algunas competencias.

CAPITULO V: PRESUPUESTO DEL MODULO

5.1 Presupuesto

A continuación, se presenta el presupuesto en Nuevos Soles (S/.) para implementar el módulo.

Tabla 23-A. Costo total de la implementación del módulo

No Ítem	Componente	Descripción	Cantidad	Precio Unitario	Sub-totales
Componentes Electrónicos					
1	Led	5mm	1	0.20	0.20
2	Buzzer	5v	1	0.50	0.50
3	Transistor	BC548	1	0.30	0.30
4	Pulsador	12mm	1	1.00	1.00
5	Switch deslizable	3 pines	1	0.50	0.50
6	Resistencias	1k,10k,330ohm	3	0.10	0.30
7	Diodo	1N4007	1	0.20	0.20
8	Arduino	Nano	1	15.00	15.00
9	Bluetooth	HC-05	1	18.00	18.00
10	Driver de motores	TB6612FNG	1	15.00	15.00
11	Sensor ultrasónico	HC-SR04	1	6.00	6.00
12	Sensor de línea	QTR-3A	1	17.80	17.80
13	Batería	Lipo 7.4v 850mah	1	50.00	50.00
Componentes Mecánicos					
14	Kit de orugas	Tamiya 70100	1	37.80	37.80
15	Micromotores	N20	2	25.00	50.00
16	Servomotor	SG90	1	8.00	8.00

Tabla 23-B. Costo total de la implementación del módulo

No Item	Componente	Descripción	Cantidad	Precio Unitario	Sub-totales
Otros					
17	Manufactura PCB	JLCPCB	1	5.80	5.80
18	Chasis y brackets	Impresión 3D	1	25.00	25.00
19	Conectores molex	2 pines	2	0.50	1.00
20	Cables jumper 1x40	Hembra - hembra	1	5.00	5.00
21	Espadines 1x40	Macho y hembra	1	2.00	2.00
22	Espaciadores	22mm tornillo M3	3	1.20	3.60
23	Tornillos y tuercas	1/8" 10mm	6	0.10	0.60
24	Cargador balanceado	Bateria 2s y 3s	1	35.00	35.00
25	Mano de obra	Costo por hora	1	25.00	25.00
				Total	323.60

El costo total del módulo educativo es de 323.60 soles, un precio competente en el mercado ya que generalmente el costo de este tipo de módulos o también llamados kits educativos de robótica oscila entre los 500 a 1000 soles, en algunos casos llegando a costar hasta 2000 soles, como son la serie EV3 de la marca LEGO EDUCATION.

Figura 55. Precio del módulo Mbot disponible en el mercado peruano [69]

Set De Robot Educativo
Transformable Mbot...

S/ 992

Envío gratis

Figura 56. Precio del módulo Mbot Ranger disponible en el mercado peruano [70]

Lego Mindstorms Education Ev3

S/ 1.975

Figura 57. Precio del módulo Mbot Ranger disponible en el mercado peruano [71]

Es importante resaltar que el presupuesto fue realizado en base a los costos que nos ofrecieron los distribuidores locales, pudiendo reducirse más si se importan los componentes de China, una opción que se está evaluando para la producción en lotes del módulo, de acuerdo con los requerimientos de las instituciones interesadas.

Adicionalmente el costo de los servicios de impresión 3D en el país está disminuyendo, permitiendo realizar mejoras o reparaciones al módulo sin afectar directamente el costo del módulo.

CONCLUSIONES

- Se logró diseñar e implementar un sistema electrónico didáctico para promover el desarrollo de la electrónica, robótica y programación en estudiantes de educación básica regular utilizando una interfaz amigable, flexible y programable.
- Después de un análisis de las diferentes alternativas de módulos electrónicos educativos existentes en el mercado, se identificaron las carencias y características más relevantes, resultando en el diseño e implementación de un prototipo acorde a nuestras necesidades como se describe en la tabla 13.
- Se realizo una correcta selección de los sensores, actuadores y microcontroladores necesarios para el diseño del módulo y enfocados a nuestros requerimientos, como se detalla en la tabla 14, 15, 16 y 17 respectivamente, manteniendo un costo bajo en el producto final.
- Se consiguió implementar el hardware necesario considerando los distintos factores propuestos, de igual forma se seleccionó el software adecuado para que el módulo funcione en conjunto de manera óptima.
- Se elaboraron guías de aprendizaje progresivas que permiten el uso correcto del módulo y brindan un mejor entendimiento del funcionamiento de este y cada una de sus partes.

RECOMENDACIONES

- Es necesario la búsqueda de nuevos métodos de enseñanza, mediante los cuales se pueda llegar de mejor manera a los estudiantes, logrando el entendimiento pleno de lo enseñado, por ello no debemos dejar de lado recursos tecnológicos con los cuales ellos están más familiarizados.
- La enseñanza de manera práctica se ha convertido en una manera más eficiente para que los alumnos pueden llegar al entendimiento pleno del tema, el aprendizaje basado en proyectos donde se emplee lo enseñado, está siendo usado y explorado cada vez más.
- Al comenzar en este campo de manera didáctica, es más probable que los conocimientos perduren más tiempo en los alumnos y sean de más fácil entendimiento.
- El uso de este módulo en la enseñanza permite estimular el interés de estudiantes en ciencias tecnológicas e ingeniera, impulsándolos a desarrollarse cada vez más en ese campo.

REFERENCIAS BIBLIOGRAFICAS

[1] MINEDU Perú, «PERÚEDUCA SISTEMA DIGITAL PARA EL APRENDIZAJE,» [En línea]. Disponible en: http://www.perueduca.pe/inicio.

[2] «¿En qué consiste el proyecto de robótica educativa?,» PerúEduca, 04 agosto 2016. [En línea]. Disponible en: https://www.youtube.com/watch?v=ODB2iRdy_dE&feature=emb_logo.

[3] PerúEduca, [En línea]. Disponible en: https://perueduca.info/docentes/proyecto-robotica-educativa/.

[4] MINEDU Perú, «PERÚEDUCA SISTEMA DIGITAL PARA EL APRENDIZAJE,» [En línea]. Disponible en: http://www.perueduca.pe/robotica/.

[5] LEGO®, [En línea]. Disponible en: https://www.lego.com/en-us/product/lego-education-wedo-2-0-core-set-45300.

[6] Makeblock, «mBot: Kit de robot educativo de nivel de entrada,» [En línea]. Disponible en: https://www.makeblock.com/mbot/.

[7] Makeblock, «mBot STEAM EDU Kit Robot Science,» [En línea]. Disponible en: https://education.makeblock.com/mbot-robot-science-kit/.

[8] A. Ollero, Robótica. Manipuladores y robots móviles. Barcelona: MARCOMBO, pp. 1-2..

[9] EcuRed, [En línea]. Disponible en: http://www.ecured.cu.

[10] F. Bravo y A. Forero, «La robótica como un recurso para facilitar el aprendizaje y desarrollo de competencias generales,,» *Teoría de la*

Educación. Educación y Cultura en la Sociedad de la Información, vol. 13, nº 2, pp. 120-136, 2012 [En línea]. Disponible en https://www.redalyc.org/pdf/2010/201024390007.pdf.

[11] I. Moreno, . L. Muñoz, J. Serracín, J. Quintero, K. Pittti y J. Quiel, «La robótica educativa, una herramienta para la enseñanza-aprendizaje de las ciencias y las tecnologías,» *Teoría de la Educación. Educación y Cultura en la Sociedad de la Información,* vol. 13, nº 2, 2012. [En línea]. Disponible en https://www.redalyc.org/pdf/2010/201024390005.pdf, pp. pp. 74-90.

[12] S. Monsalves, «Estudio sobre la utilidad de la robótica educativa desde la perspectiva del docente,» *Revista de pedagogía,* vol. 32, nº 90, pp. 81-117, 2011 [En línea]. Disponible en https://www.redalyc.org/pdf/659/65920055004.pdf.

[13] G. Bermudez, «Robots móviles. Teoría, aplicaciones y experiencias,» *Tecnura,* vol. 5, nº 10, pp. 6-17, 2001. [En línea]. Disponible en https://doi.org/10.14483/22487638.5882.

[14] I. Bambino, Una Introducción a los Robots Móviles, 2008. En línea]. Disponible en https://aadeca.org/pdf/CP_monografias/monografia_robot_movil.pdf.

[15] L. Ríos y M. Bueno, «Modelo matemático para un robot móvil,» *Scientia Et Technica,* vol. 14, nº 38, pp. 13-18, Jun., 2008. [En línea]. Disponible en https://www.redalyc.org/pdf/849/84903803.pdf.

[16] P. Aguayo, «Introducción al microcontrolador,» Nov. 2004. [En línea]. Disponible en: https://es.slideshare.net/josediarte71/microcontroladores-31794913.

[17] L. Parra, Microprocesadores, Estado de México: Red Tercer Milenio, 2012. p.24.

[18] A. Bizama, «Introducción a los microcontroladores,» 14 Nov. 2012. [En línea]. Disponible en: http://anibalbizama.blogspot.com/2012/11/introduccion-los-microcontroladores.html.

[19] E. Palacios, F. Remiro y L. López, , Microcontrolador PIC16F84. Desarrollo de proyectos, 1ª ed., México: Alfaomega, 2004. p. 1..

[20] MATPIC, «PICs MICROCHIP,» 2017. [En línea]. Disponible en: http://www.matpic.com/esp/microchip/microchip.html.

[21] J. Viera, Microchip PIC, Ene 2009. [En línea]. Disponible en: http://www.iuma.ulpgc.es/~nunez/clases-micros-para-com/mpc0809-trabajos/mpc0809JosueVierauC_PIC.pdf.

[22] C. Tapia y . H. Manzano, Evaluación de la plataforma Arduino e implementación de um sistema de control de posición horizontal Fac. De Ing., Univ. Politécnica Salesiana, Guayaquil, Ecuador: https://dspace.ups.edu., 2013.

[23] Y. Fernández, «Xataka Basics,» 3 Agosto 2020. [En línea]. Disponible en: https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno.

[24] R. Vaseux, «Desarrollo de un sistema operativo para Raspberry Pi con sus drivers básicos,» Trabajo fin de grado Univ. Politécnica de Madrid, Madrid, España, 2017. , [En línea]. Disponible en: http://oa.upm.es/48933/1/TFG_ROBERT_ALEXANDER_VAZEUX_BLANCO.pdf.

[25] E. Rodríguez, «Xataka,» 18 set. 2018. [En línea]. Disponible en: https://www.xataka.com/makers/cero-maker-todo-necesario-para-empezar-raspberry-pi.

[26] Güimi, «Lenguajes de programación,» Oct. 2008. [En línea]. Disponible en: https://guimi.net/descargas/Monograficos/G-Lenguajes_de_programacion.pdf.

[27] Rock Content, «¿Qué es un lenguaje de programación y qué tipos existen?,» 20 Abr 2019. [En línea]. Disponible en: https://rockcontent.com/es/blog/que-es-un-lenguaje-de-programacion/#:~:text=Es%20un%20lenguaje%20formal%20que,y%20l%C3%B3gico%20de%20.

[28] M. Orenga y G. Manonellas, Programación en ensamblador (x86-84)., Catalunya: UOC, 2011.

[29] J. Rodríguez, Introducción a la programación. Teoría y práctica. Alicante: Editorial Club Universitario, 2003. p. 7..

[30] M. Peña y J. Cela, Introducción a la programación en C., Barcelona: Ediciones UPC, 2000.

[31] P. Abel, Lenguaje ensamblador y programación para IBM PC y compatibles, 3ª ed., Estado de México: PRENTICE HALL HISPANOAMERICANA, 1996.

[32] G. Bronson, C++ para ingeniería y ciencias, 2ª ed., México D.F.: Cengage Learning, 2007.

[33] L. Olivares, Manual de Programación en Lenguaje C++, México: Trabajo de investigación, UNAM, 2008. Disponible en https://paginas.matem.unam.mx/pderbf/images/mprogintc++.pdf.

[34] Didact, Lenguaje C++: Manual de programación, 2ª ed., Alcalá de Guadaira: MAD, 2005. p. 11..

[35] A. Sarasa, Gestión de la información web usando Python, Barcelona: UOC, 2016, p. 5..

[36] J. Troyano, F. Cruz, M. Gonzáles, C. Vallejo y M. Toro, «Introducción a la Programación con Python, Computación Interactiva y Aprendizaje Significativo,» *Actas de las Jenui,* vol. 3, pp. 223-230, Nov., 2018.https://www.resear.

[37] I. Challenger, Y. Díaz y R. Becerra, «El lenguaje de programación Python,» *Ciencias Holguín,* vol. 10, nº 2, pp. 1-13, Abr.-Jun., 2014. [En línea]. Disponible en https://www.redalyc.org/pdf/1815/181531232001.pd

[38] Soluciones uno, «Python, el mejor lenguaje de programación,» 16 Abr. 2014. [En línea]. Disponible en: https://www.solucionesuno.com/ blog/python-el-mejor-lenguaje-de-programacion/v.

[39] A. Becerra, Introducción a la programación con Python, 2a ed., Santiago de Cali: Sello Editorial Javeriano, 2009.

[40] R. Pallas, Sensores y Acondicionadores de Señal, 4a ed. Barcelona: MARCOMBO, 2003. p. 3..

[41] L. Corona, G. Abarca y J. Mares, Sensores y Actuadores. Aplicaciones con Arduino, Mexico D.F.: Grupo Editorial Patria, 2014, pp. 17-34..

[42] A. Serna, Guía Práctica de Sensores, España: Creaciones Copyright, 2010, p. 3..

[43] Señales, «Unidad de apoyo para el aprendizaje,» [En línea]. Disponible en: https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/824/mod_resource/content/5/contenido/index.html.

[44] F. Torres, J. Pomares, R. Puente , S. Puente y R. Aracil, Robots y Sistemas Sensoriales, Madrid: Pearson Education, 2002. pp. 170-182.

[45] «Super Robótica,» [En línea]. Disponible en: http://www.superrobotica.com/S320110.htm.

[46] «Robot Electronics,» SRF04 – Ultra-Sonic Ranger Technical Specification, 2012. [En línea]. Disponible en: http://www.robot-electronics.co.uk/htm/srf04tech.htm.

[47] Kuongshun Electronic Limitedv, «szks-kuongshun,» 25 sep 2018. [En línea]. Disponible en: http://www.szks-kuongshun.com/info/complete-guide-for-ultrasonic-sensor-hc-sr-29545111.html.

[48] «Proyecto Arduino,» [En línea]. Disponible en: Disponible en https://proyectoarduino.com/sensor-de-ultrasonidos-medir-distancia-con-arduino/.

[49] Vistrónica , «Vistrónica. Tienda Virtual De Electrónica,» [En línea]. Disponible en: https://www.vistronica.com/aeromodelismo/sensor-de-ultrasonido-gy-us42-para-controlador-apm--detail.html.

[50] Pololu Corporatio, «Pololu,» [En línea]. Disponible en: https://www.pololu.com/product/960.

[51] Pololu Corporation, «Pololu,» [En línea]. Disponible en: https://www.pololu.com/product/2456.

[52] Pololu Corporation, «Pololu,» [En línea]. Disponible en: https://www.pololu.com/product/2458.

[53] A. Barrientos, L. Peñín, C. Balaguer y R. Aracil, Fundamentos de Robótica, 2a ed., España: : McGraw-Hill, 2007.

[54] S. Yaguana, Automatización y Monitoreo de un Brazo Robótico para la Manipulación, Transporte y Clasificación de piezas en un Área de Trabajo, Quito, Ecuador: Universidad San Francisco de Quito , 2010. Disponible en http://repositorio.usf.

[55] E. Mendoza, Implementación de un sistema de captura de paquetes en redes inalámbricas 802.11 y Bluetooth, Huajuapan de León, México: Universidad Tecnica De Mixteca, 2005. Disponible en http://jupiter.utm.mx/~tesis_dig/9583.pdf.

[56] M. Otero, Conexión de dispositivos Bluetooth en un entorno virtual creado con el motor unity, Vigo - España: Universidad De Vigo, 2019. Disponible en http://castor.det.uvigo.es:8080/xmlui/bitstream/handle/123456789/361/TFG%20Miguel%20%C3.

[57] Electronilab, [En línea]. Disponible en: https://electronilab.co/tienda/modulo-bluetooth-hc-05-serial-rs232/.

[58] Naylamp Mechatronics, [En línea]. Disponible en: https://naylampmechatronics.com/blog/24_configuracion-del-modulo-bluetooth-hc-05-usa.html.

[59] Geek Factory (, [En línea]. Disponible en: https://www.geekfactory.mx/tienda/radiofrecuencia/hc-05-modulo-bluetooth-maestro-esclavo/.

[60] Naylamp Mechatronics, [En línea]. Disponible en: https://www.naylampmechatronics.com/blog/12_Tutorial-B%C3%A1sico-de-Uso-del-M%C3%B3dulo-Bluetooth-H.html.

[61] Instituto de Tecnología de Massachusetts, «MIT APP INVETOR,» [En línea]. Disponible en: https://appinventor.mit.edu/about-us.

[62] C. Peña, Estudio de baterías par vehículos eléctricos, Madrid, España, : Universidad Carlos III de Madrid, 2011. Disponible en https://e-archivo.uc3m.es/bitstream/handle/10016/11805/PFC_Carlos_Pena_Ordonez.pdf?sequenc.

[63] J. Cabezas, Diseño e implementación de un sistema electrónico de potencia híbrido para la carga de un batería tipo polímero de litio, Quito, Ecuador: Escuela Politécnica Nacional, 2016. Disponible en https://bibdigital.epn.edu.ec/.

[64] A. Barrientos, L. Peñín, C. Balaguer y R. Aracil, Fabricación digital: Introducción al modelado e impresión 3D, España: Secretaría General Técnica, 2016. pp. 8-9..

[65] Geek Factory, «Sensor ultrasónico HC-SR04 y Arduino,» 16 May. 2014. [En línea]. Disponible en: https://www.geekfactory.mx/tutoriales/tutoriales-arduino/sensor-ultrasonico-hc-sr04-y-arduino/.

[66] Nyereka Tech, « Nyereka Tech,» [En línea]. Disponible en: https://www.nyerekatech.com/wp-content/uploads/2019/10/nano4-300x300.png.

[67] CORE Electronics, « CORE Electronics,» [En línea]. Disponible en: https://core-electronics.com.au/tamiya-70100-track-and-wheel-set.html.

[68] Shopify, [En línea]. Disponible en: https://cdn.shopify.com/s/files/1/0409/9041/products/HC-05-03_miniATmode_03_compact.jpg?v=1615326473.

[69] Mercado Libre, [En línea]. Disponible en: https://articulo.mercadolibre.com.pe/MPE-441292110-kit-de-makeblock-mbot-robot-programable-con-scratch-20-_JM#position=6&search_layout=stack&type=item&tracking_id=ad0e257 4-2c1b-4f23-96f2-f796f980018f.

[70] Mercado Libre, [En línea]. Disponible en: https://articulo.mercadolibre.com.pe/MPE-441268177-set-de-robot-educativo-transformable-mbot-ranger-stem-de-mak-_JM#position=12&search_layout=stack&type=item&tracking_id=d2de38 2e-4706-4114-9ef5-47b1b0b3a782.

[71] Mercado Libre, [En línea]. Disponible en: https://listado.mercadolibre.com.pe/lego-mindstorm-ev3-education.

ANEXOS

ANEXO 1. GUIA 1: ENSAMBLAJE DEL ROBOT

UNIVERSIDAD CATÓLICA DE SANTA MARÍA **ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA**		
CODIGO ASIGNATURA		GUÍA DE LABORATORIO Nº 01
MODULO I	ENSAMBLAJE DEL ROBOT	**Docente (s):** Bruno Fernando Valencia Rodriguez
		Fecha: 2021.05.04.

I. OBJETIVOS DE LABORATORIO

a. Identificar los elementos que forman parte de nuestro robot.
b. Armar el robot para su posterior programación

II. TEMAS A TRATAR

- Partes de un robot.
- Diferencia entre sensores y actuadores.
- Componentes de la tarjeta de control.

III. HERRAMIENTAS

1. Computador con ARDUINO IDE.
2. Modulo educativo de robótica.
3. Desarmador punta estrella
4. Cable USB.
5. Guía de laboratorio.

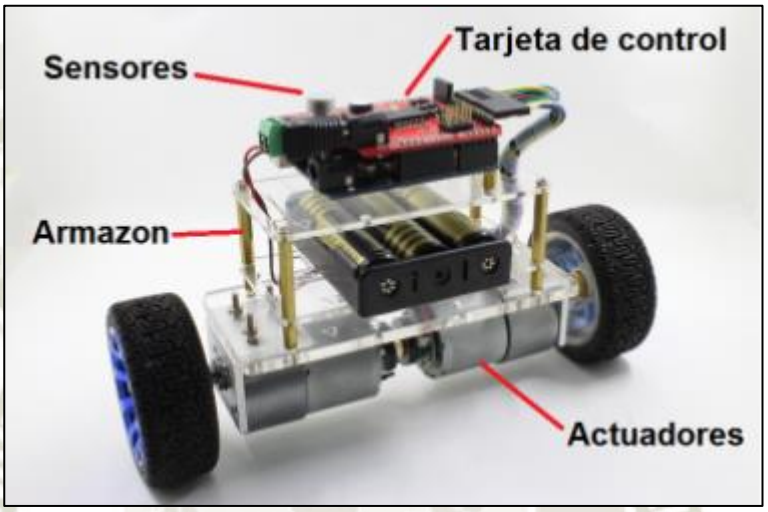
IV. BIBLIOGRAFIA

- R. Hernández, *Introducción a los Sistemas de Control: Conceptos, aplicaciones y simulación con MATLAB,* 1ra. Edición, México: Pearson, 2010.
- *Control System Toolbox For Use with MatLab – Getting Started,* The MatWorks Inc., Natick, Massachusetts, 2004.
- *Control System Toolbox For Use with MatLab - Using the Control System Toolbox*, The MatWorks Inc., Natick, Massachusetts, 2002.
- Arduino (s.f.). “Guía de Referencia de Arduino” [Internet]. Disponible en https://www.arduino.cc/reference/es/
- Electrónica Educativa (2018, Jun. 06). “Las partes de un robot” [Internet]. Disponible en https://electronicaeducativa.com/index.php/2018/06/06/partes-de-robot/

V. MARCO TEORICO

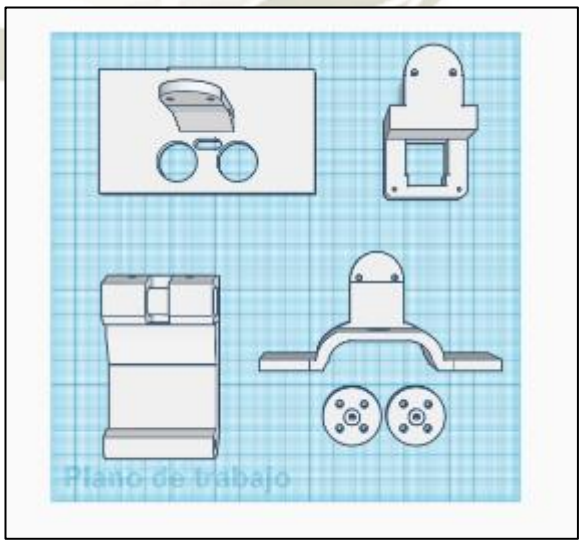
Partes de un robot:

La estructura general de un robot se compone de cuatro partes fundamentales, donde la importancia de cada una de ellas dependerá de la tarea concreta para la que fue construido. Presentamos en la imagen las siguientes partes que se describen a continuación:



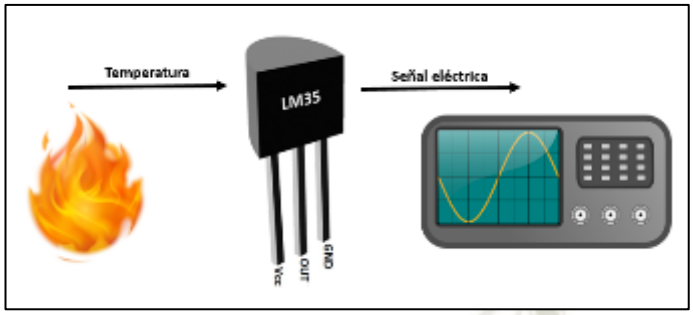
a) Armazón o esqueleto del robot

El armazón es como el esqueleto de un ser humano. Es la parte que soporta los componentes del que está compuesto el robot. Una característica es su robustez, el tipo de material, facilidad para el cambio y del tipo de trabajo a desempeñar.



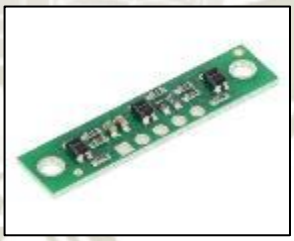
b) Sensores o receptores de estímulos

Un sensor es un dispositivo eléctrico y/o mecánico que convierte magnitudes físicas (luz, magnetismo, presión, etc.) en valores medibles de dicha magnitud.



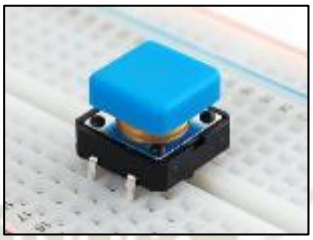
Todo robot debe está diseñado para cumplir una tarea en función a los estímulos externos que recibe del exterior. Para estos los sensores deben ser seleccionados y colocados de manera estratégica sobre la estructura. Los principales tipos de sensores son:

Sensor óptico/reflexivo IR (Línea): El LED infrarrojo emite luz infrarroja, o sea, de mayor longitud de onda (o menor frecuencia) que la podemos ver los humanos, así que para nosotros es invisible. Si esta luz choca contra una superficie blanca se reflejará y llegará al fototransistor. Si por el contrario golpea en una superficie negra, el material absorberá la mayoría de la luz y no llegará al fotorreceptor.



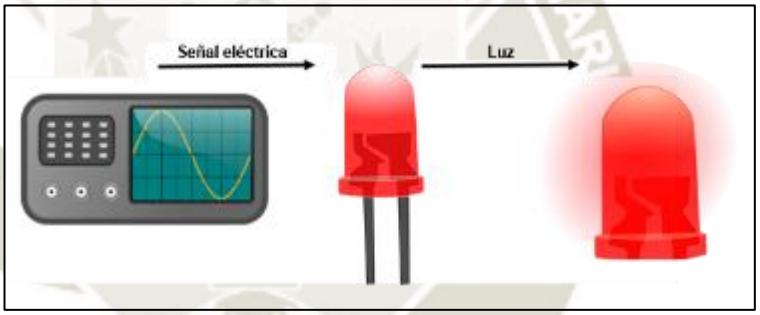
Sensor de ultrasonido (Distancia): Son muy frecuentes en los robots móviles y de forma significativa en los AUVs (Vehículos autónomos bajo el agua) por sus buenas propiedades de medición en entornos acuáticos y sirven para detectar objetos y medir distancias. Se utilizan para construir mapas del entorno y evitar obstáculos.

Sensor de presión (Botón): Se utilizan las palabras botón, tecla o pulsador para describir a una pieza que al ejercer presión sobre ella (generalmente con un dedo), produce un efecto determinado. Los botones son de diversas formas y tamaños y se encuentran en todo tipo de dispositivos, aunque principalmente en aparatos eléctricos y electrónicos. Permiten el flujo de corriente mientras son accionados. Cuando ya no se presiona sobre él vuelve a su posición de reposo.



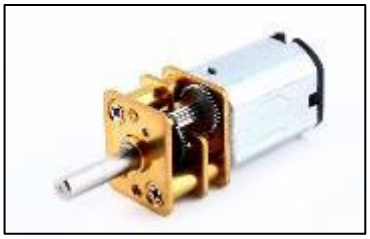
c) Actuadores

Un actuador es un dispositivo capaz de transformar energía hidráulica, neumática o eléctrica en la activación de un proceso con la finalidad de generar un efecto sobre un elemento externo.



El robot luego de captar y acondicionar los datos del entorno en señales eléctricas, deberá procesarlos para resultar en una tarea programada, como puede ser desplazarse o activar una alarma. Esto se lleva a cabo mediante el uso de actuadores, los actuadores comúnmente utilizados son:

Motores: El motor eléctrico es un dispositivo que convierte la energía eléctrica en energía mecánica de rotación por medio de la acción de los campos magnéticos generados en sus bobinas. Son máquinas eléctricas rotatorias compuestas por un estator y un rotor.

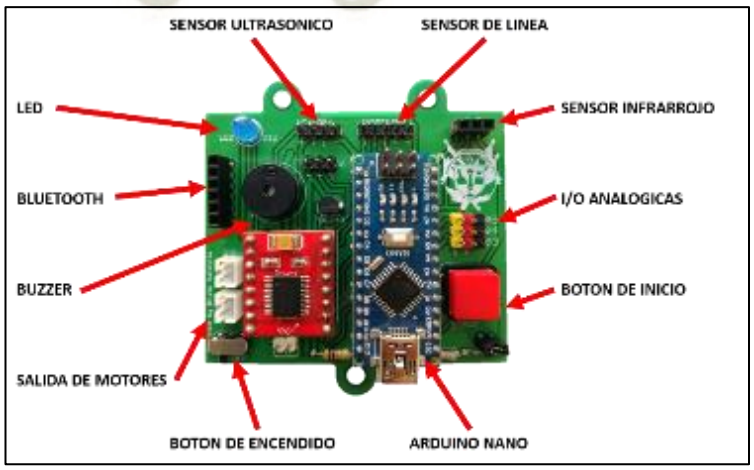


Luces LED: Un diodo emisor de luz o LED, es una fuente de luz constituida por un material semiconductor dotado de dos terminales. Se trata de un diodo que al aplicar una tensión adecuada emite luz.

Bocina: Un zumbador (en inglés: buzzer) es un transductor electroacústico que produce un sonido o zumbido continuo o intermitente de un mismo tono (generalmente agudo). Sirve como mecanismo de señalización o aviso y se utiliza en múltiples sistemas, como en automóviles o en electrodomésticos, incluidos los despertadores.

d) Tarjeta de control o cerebro del robot

Para que exista un movimiento o acción del robot por parte de los estímulos externos, se utiliza una lógica de control que rige el comportamiento de la máquina. Por lo general se trata de sistemas basado en microcontroladores que programados de manera conveniente resuelven de forma óptima una tarea. Hoy en día ya viene insertados en tarjetas de desarrollo en función a la aplicación que se le desee dar.

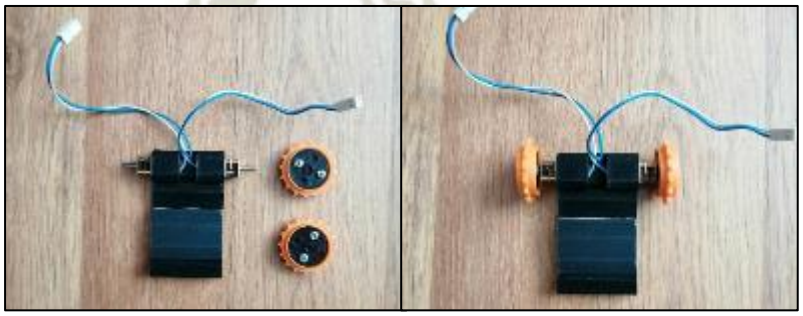


VI. ACTIVIDADES

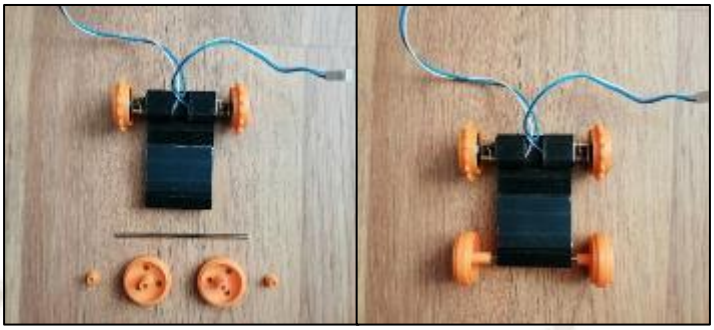
1. Identificar las partes del módulo separándolas en sensores actuadores, tarjeta de control y piezas mecánicas.

2. Seleccionar los 2 motores para colocarlos en el chasis.

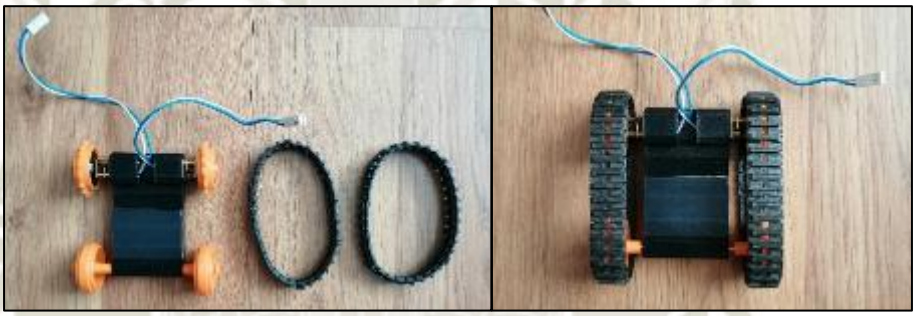
3. Insertar el eje tipo D del motor en las ruedas dentadas.



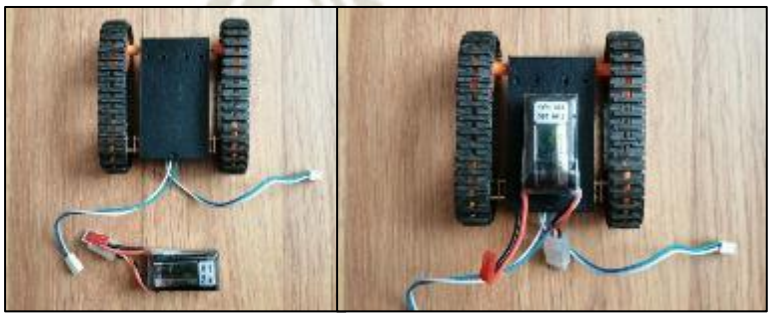
4. Colocar las ruedas libres en la parte delantera del chasis con la ayuda del eje de acero y los sujetadores.



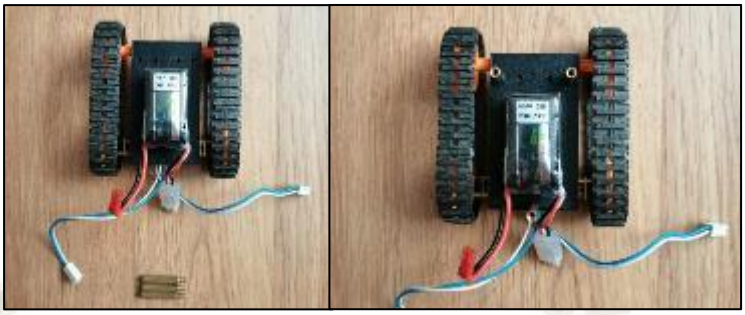
5. Luego de tener aseguradas correctamente las 4 ruedas, procederemos a sujetar las orugas como se indica a continuación.



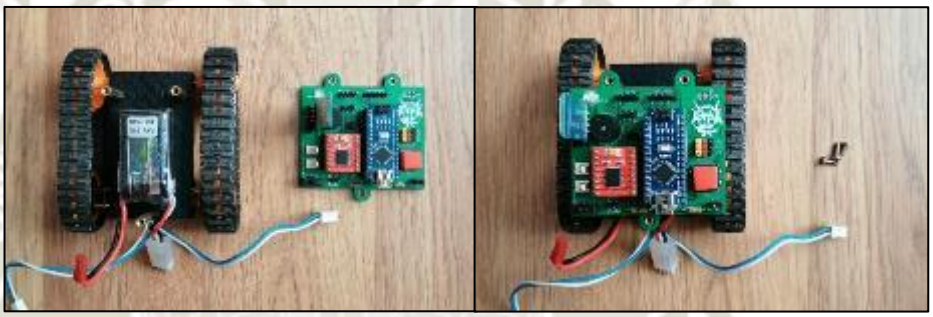
6. Con la ayuda de cinta de doble contacto aseguramos la batería en el chasis, respetando la disposición de los cables de alimentación (negro y rojo), para posteriormente poder conectarlos a la tarjeta del control.



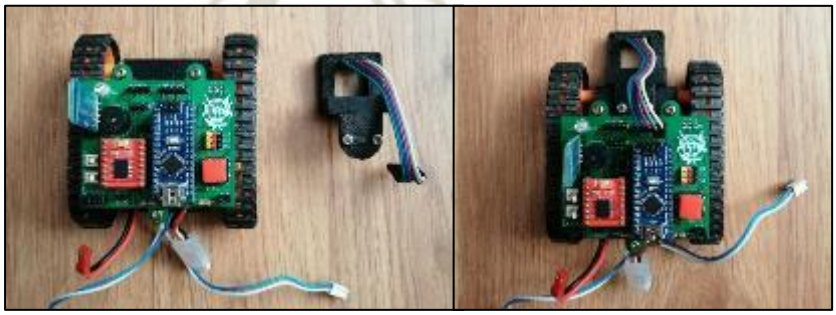
7. Atornillamos los espaciadores de bronce que nos permitirán sujetar la placa al chasis.



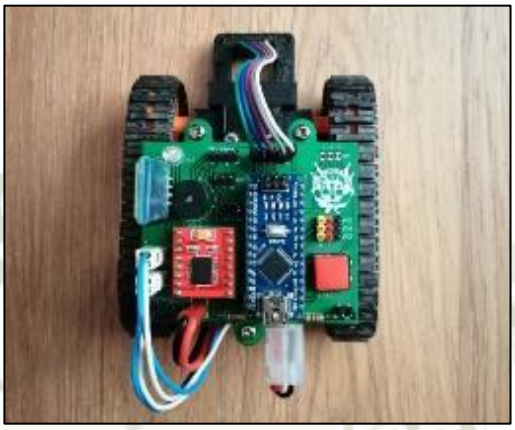
8. Sujetamos la placa al chasis del robot utilizando los 3 tornillos que vienen con los espaciadores de bronce.



9. Adicionalmente los 2 agujeros presentes en el chasis nos permiten sujetar distintos tipos de brackets para sensores o aplicaciones específicas, como se verá más adelante. Sujetaremos el sensor de línea con su respectivo bracket para utilizarlo en las siguientes prácticas.



10. El último paso es muy importante y requiere la supervisión de un profesor o asesor, primero se debe conectar los motores, el motor izquierdo ira en el conector situado en la parte inferior izquierda de la tarjeta de control donde dice **M1** y el motor derecho ira conectado a **M2**, luego conectamos el sensor de línea que cuenta con 5 cables a los pines señalados como **LINESENSOR** y por último conectamos la batería respetando la señalización de **POSITIVO "+" con Rojo** y **NEGATIVO "–" con NEGRO.**



11. Una vez realizados todos los pasos anteriores, ya contamos con un robot listo para ser programado en la siguiente práctica.

VII. CONCLUSIONES

Emita al menos cinco conclusiones acerca de las partes de un robot y los tipos de sensores y actuadores utilizados en la robótica.

ANEXO 2. GUIA 2: PROGRAMACION DEL ROBOT

UNIVERSIDAD CATÓLICA DE SANTA MARÍA ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA		
CODIGO ASIGNATURA		GUÍA DE LABORATORIO Nº 02
MODULO I	PROGRAMACION DEL ROBOT	**Docente (s):** Bruno Fernando Valencia Rodriguez
		Fecha: 2021.05.04.

I. OBJETIVOS DE LABORATORIO

a. Familiarizarse con el entorno de programación ARDUINO IDE.
b. Programación del microcontrolador ARDUINO.

II. TEMAS A TRATAR

- Estructura de un programa.
- Compilar un sketch en el IDE de ARDUINO.
- Cargar una rutina al robot.

III. HERRAMIENTAS

1. Computador con ARDUINO IDE.
2. Modulo educativo de robótica.
3. Cable USB.
4. Guía de laboratorio.

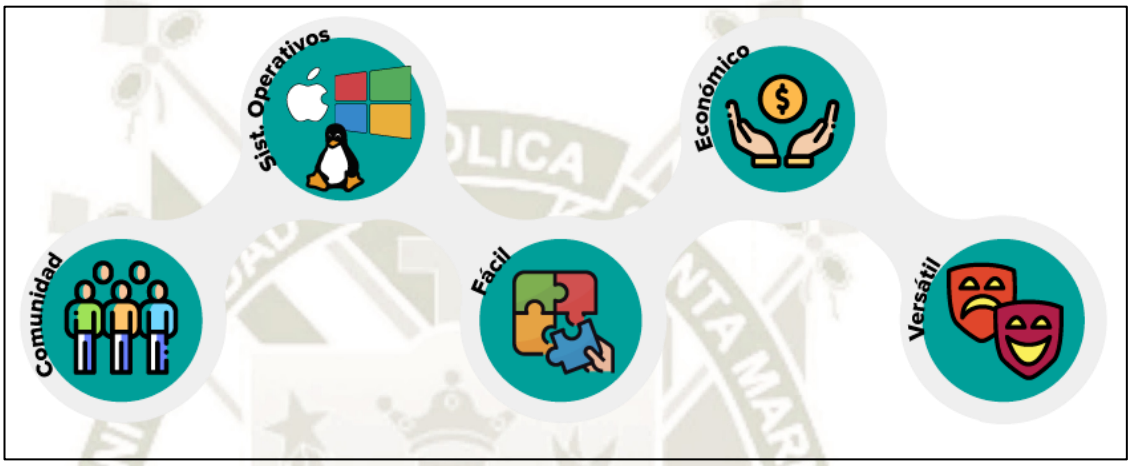
IV. BIBLIOGRAFIA

- Wordpress (2017, Jun. 19). “Aprendiendo Arduino” [Internet]. Disponible en https://aprendiendoarduino.wordpress.com/2017/06/19/estructura-sketch-arduino/
- Arduino (s.f.). “Foundations” [Internet]. Disponible en https://www.arduino.cc/en/Tutorial/Foundations#programming
- Openwebinars (2015, Feb. 03). “Tutorial Arduino: IDE Arduino” [Internet]. Disponible en https://openwebinars.net/blog/tutorial-arduino-ide-arduino/

V. MARCO TEORICO

Arduino:

Arduino es una plataforma de desarrollo basada en una placa electrónica de hardware libre que incorpora un microcontrolador re-programable y una serie de pines, los que permiten establecer conexiones entre el microcontrolador y los diferentes sensores y actuadores de una manera muy sencilla (principalmente mediante cables).



Arduino es libre y extensible, esto quiere decir que cualquiera que desee ampliar y mejorar el diseño hardware de las placas como el entorno de desarrollo, puede hacerlo sin problemas. Esto permite que exista un rico ecosistema de placas electrónicas no oficiales para distintos propósitos y de librerías de software de tercero, que pueden adaptarse mejor a nuestras necesidades.

Programación de Arduino:

Programar Arduino consiste en traducir a líneas de código las tareas automatizadas que queremos hacer leyendo de los sensores y en función de las condiciones del entorno programar la interacción con el mundo exterior mediante unos actuadores.

Arduino proporciona un entorno de programación sencillo y potente para programar, pero además incluye las herramientas necesarias para compilar el programa y "quemar" el programa ya compilado en la memoria flash del microcontrolador. Además, el IDE nos ofrece un sistema de gestión de librerías y placas muy práctico. Como IDE es un software sencillo que carece de funciones avanzadas típicas de otros IDEs, pero suficiente para programar.

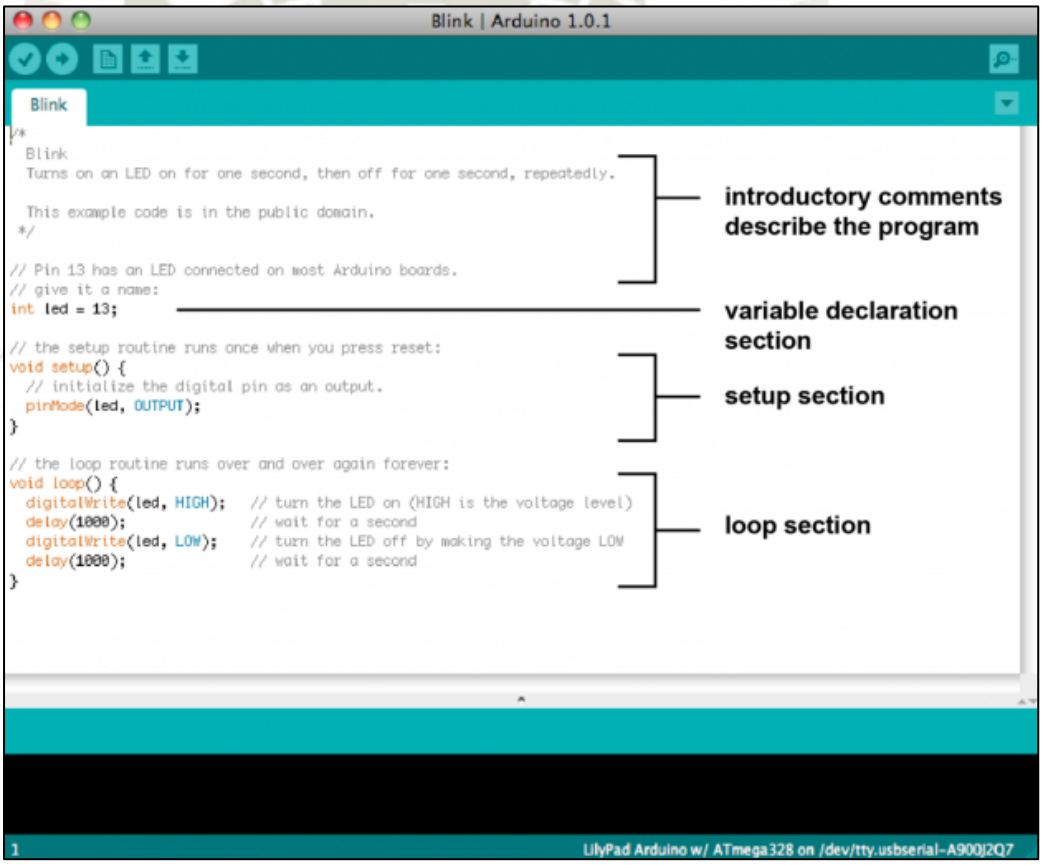
Estructura de un sketch:

Un programa en Arduino es denominado sketch o proyecto y está conformado por 2 partes principales, setup() es la parte encargada de recoger la configuración

y loop() es la que contiene el programa que se ejecuta cíclicamente (de ahí el término loop –bucle-). Ambas funciones son necesarias para que el programa trabaje.

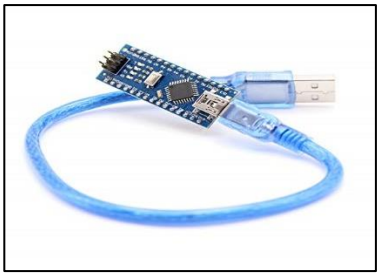
La función de configuración (setup) debe contener la inicialización de los elementos y esta función sólo se ejecuta una vez justo después de hacer el reset y no se vuelve a ejecutar hasta que no haya otro reset. Es la primera función a ejecutar en el programa y se utiliza para configurar, inicializar variables, comenzar a usar librerías y otras funciones definidas por el usuario.

La función bucle (loop) contiene el código que se ejecutará continuamente (lectura de entradas, activación de salidas, etc.). Esta función es el núcleo de todos los programas de Arduino y se usa para el control activo de la placa. La función loop se ejecuta justo después de setup.

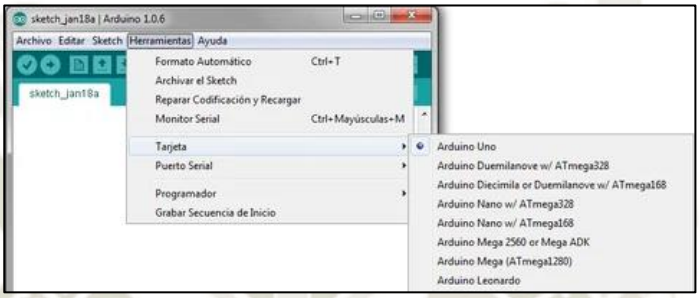


Subir un programa a la placa:

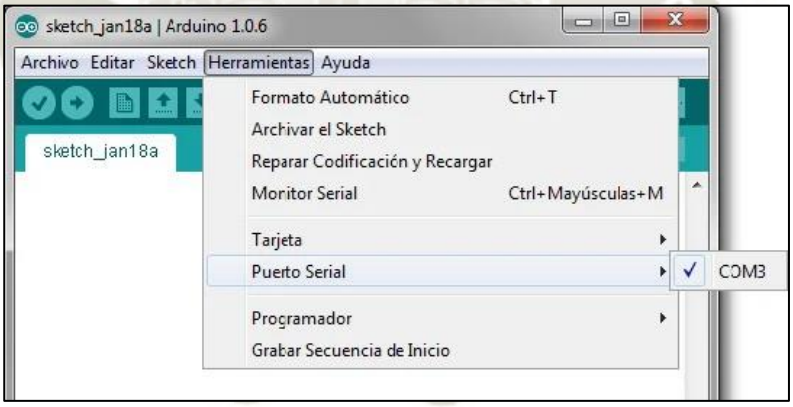
1. Conectar su Arduino con el cable USB, el extremo del cable con el puerto micro USB se conecta a Arduino y el extremo plano se conecta a un puerto USB en la computadora.



2. Elegir Herramientas → Tarjetas → Arduino Nano para encontrar la tarjeta en el menú Arduino.



3. Elegir el puerto serie correcto para la placa: Puede encontrar una lista de todos los puertos serie disponibles seleccionando Herramientas → Puerto serie → COM"X". "X" marca un número asignado de forma secuencial o aleatoria. En Windows, si acaba de conectar un Arduino, el puerto COM normalmente será el número más alto, como COM3 o COM5.



4. Haga clic en el botón Cargar: Este es el botón que apunta a la derecha en el entorno Arduino. También puede utilizar el método abreviado de teclado Ctrl + U para Windows.

VI. ACTIVIDADES

1. Primero debemos declarar las variables que vamos a utilizar, muy independientemente de la función que cumplirá el módulo, se declararan todas las variables necesarias para cada una de las funciones definidas, esto permitirá trabajar más rápido al momento de realizar el código, los tipos de variables serán:

   - Variables para el control de motores
   - Variables para lectura de sensores
   - Variables para utilizar los actuadores integrados en la placa
   - Variables globales para almacenar valores

Con la ayuda de los comentarios en el código "//" vamos a diferenciar cada una de ellas.

```
//SERVOMOTOR
#include <Servo.h>
Servo servo;

//MOTOR DERECHA
int PWMA = 3; //Control de velocidad
int AIN1 = 4; //Dirección
int AIN2 = 2; //Dirección

//MOTOR IZQUIERDA
int PWMB = 5; //Control de velocidad
int BIN1 = 6; //Dirección
int BIN2 = 7; //Dirección

//SENSOR ULTRASONICO
int echo = 9;
int trig = 10;
int tiempo, distancia, lectura;  //Variables para el cálculo de distancia

//SENSOR DE LINEA
int SensorIzq  = A0;
int SensorCen = A1;
int SensorDer = A2;
int Sizq = 0;
int Scen = 0;
int Sder = 0;
int linea = 400;  //Variable para definir el tono de la linea

//INTEGRADOS EN LA PLACA
```

```
int pulsador=11;
int estado=0;
int led=12;
int ldr=13;
```

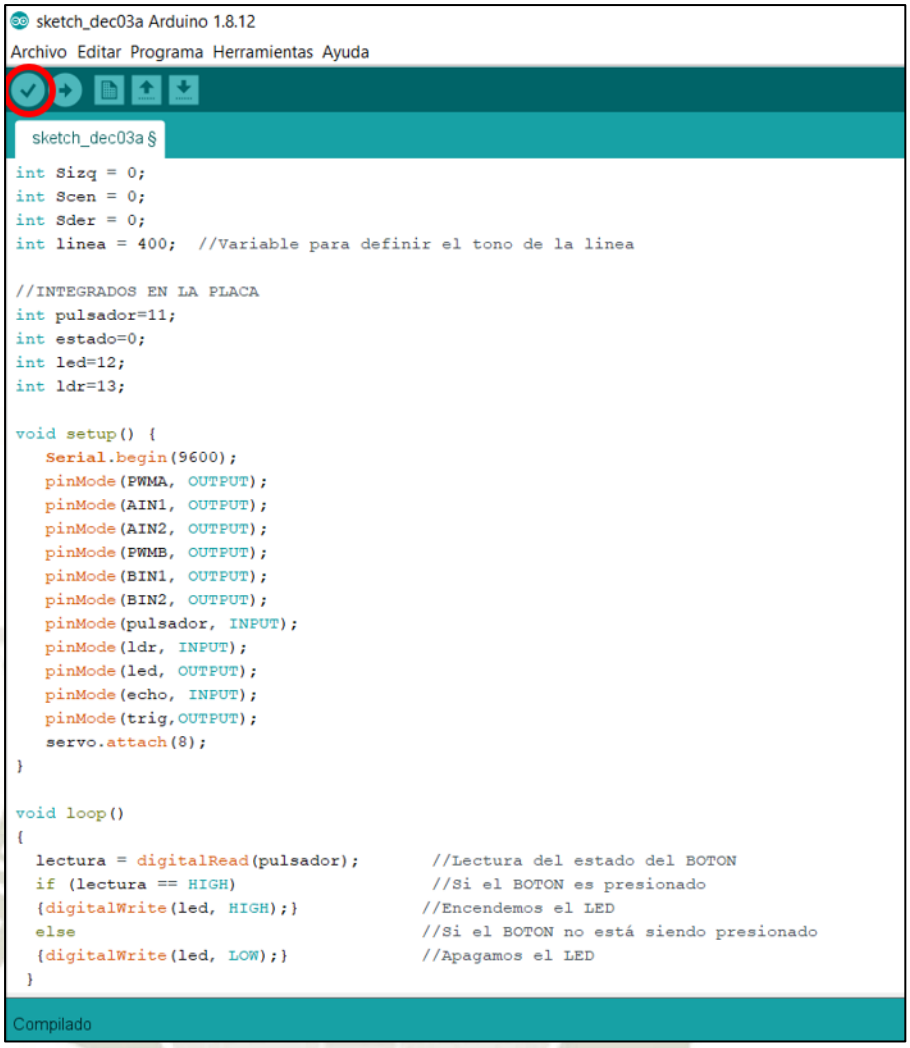
2. El siguiente paso es configurar el **void setup()**, el cual se encarga de especificar al microcontrolador los comandos que ejecutará en el momento del arranque

```
void setup() {
  Serial.begin(9600);
  pinMode(PWMA, OUTPUT);
  pinMode(AIN1, OUTPUT);
  pinMode(AIN2, OUTPUT);
  pinMode(PWMB, OUTPUT);
  pinMode(BIN1, OUTPUT);
  pinMode(BIN2, OUTPUT);
  pinMode(pulsador, INPUT);
  pinMode(ldr, INPUT);
  pinMode(led, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(trig,OUTPUT);
  servo.attach(8);
 }
```

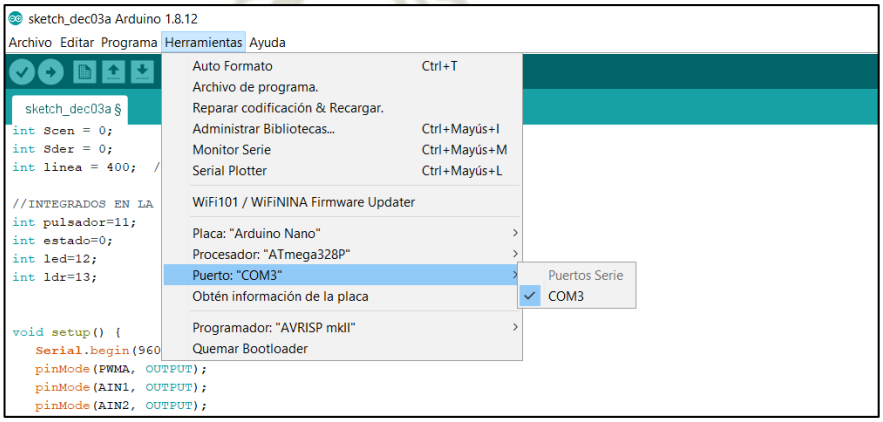
3. La función de bucle **void loop()** es la función principal, esta parte del programa se encargará de definir la aplicación de nuestro modulo, ya sea seguidor de líneas, esquiva obstáculos o permitirnos controlar el led integrado para que se encienda cada vez que presionamos el pulsador como el siguiente ejemplo:

```
void loop()
{
 lectura = digitalRead(pulsador);     //Lectura del estado del BOTON
 if (lectura == HIGH)              //Si el BOTON es presionado
 {digitalWrite(led, HIGH);}              //Encendemos el LED
 else                                    //Si el BOTON no está siendo presionado
 {digitalWrite(led, LOW);}               //Apagamos el LED
 }
```

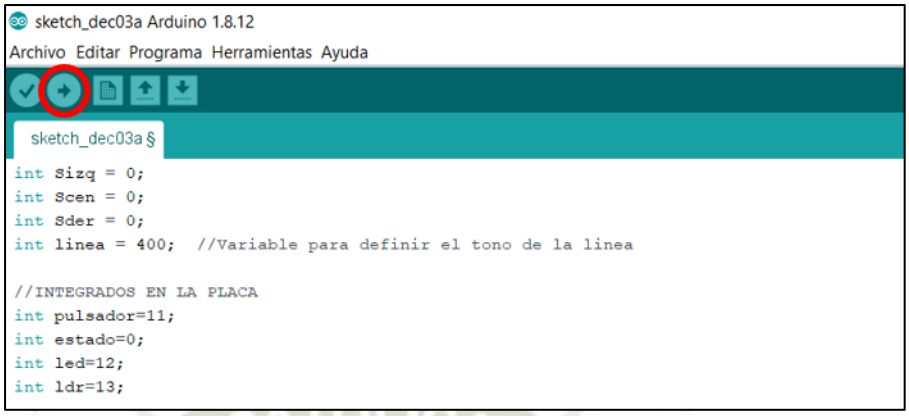
4. Cargar en orden los códigos brindados en el **paso 1, 2 y 3**, en el IDE de Arduino, luego **COMPILAREMOS** el programa, para eso debemos hacer clic en el símbolo **"√"** situado en la parte superior izquierda del entorno de programación debajo de la pestaña "Archivo"



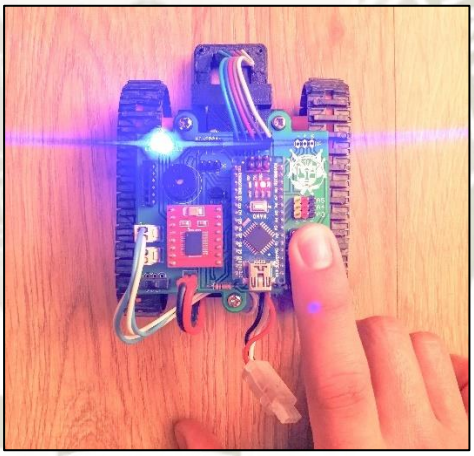
5. Luego de verificar que el código compila sin mostrar ningún error, procederemos a cargarlo a la placa, primero debemos revisar en la pestaña de herramientas que la tarjeta seleccionada sea **ARDUINO NANO** y seleccionar el puerto especifico **COM**



6. Realizados los ajustes anteriores podemos proceder a subir el programa al robot, haciendo clic sobre la flecha "→" situada al costado del botón verificar.



7. Realizados los pasos anteriores, podemos desconectar el robot de la PC y comprobar que el programa se está ejecutando de forma correcta, el módulo encenderá la luz LED cada vez que se presione el pulsador situado en la parte inferior derecha de la tarjeta.



8. Pruebe modificando el tiempo que se mantiene encendido el LED añadiendo un retardo

```
void loop()
{
 lectura = digitalRead(pulsador);
 if (lectura == HIGH)
 {digitalWrite(led, HIGH);delay(2000);}     //Retardo de 2 segundos
 else
 {digitalWrite(led, LOW);}
 }
```

VII. CONCLUSIONES

Emita al menos cinco conclusiones acerca de las condiciones **if–else** y las instrucciones **delay().**

ANEXO 3. GUIA 3: ROBOT SEGUIDOR DE LINEA

UNIVERSIDAD CATÓLICA DE SANTA MARÍA **ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA**		
CODIGO ASIGNATURA		GUÍA DE LABORATORIO Nº 03
MODULO I	ROBOT SEGUIDOR DE LINEA	**Docente (s):** Bruno Fernando Valencia Rodriguez
		Fecha: 2021.05.04.

I. OBJETIVOS DE LABORATORIO

a. Controlar el desplazamiento del robot.
b. Interpretar las lecturas de los sensores.
c. Implementar la función de seguidor de línea.

II. TEMAS A TRATAR

- Control de motores utilizando el puente H.
- Lecturas analógicas.
- Instrucciones condicionales anidadas.
- Creación de funciones.

III. HERRAMIENTAS

1. Computador con ARDUINO IDE.
2. Modulo educativo de robótica.
3. Cable USB.
4. Guía de laboratorio.

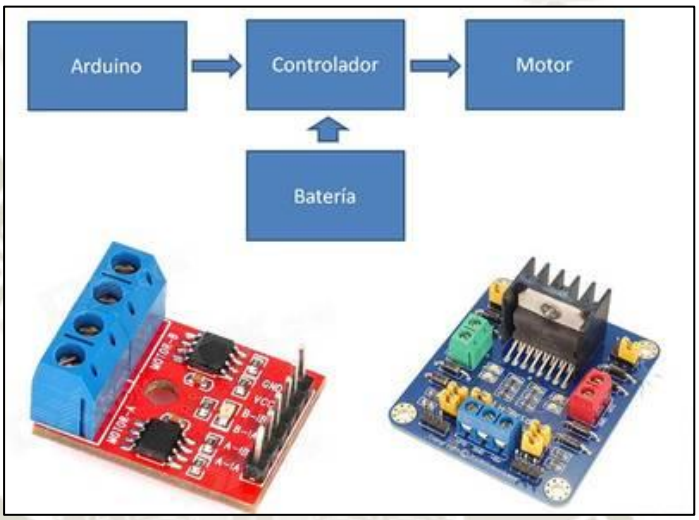
IV. BIBLIOGRAFIA

- BricoLabs Wiki (2018, May. 11). “Control de motores con Arduino” [Internet]. Disponible en https://bricolabs.cc/wiki/guias/control_de_motores
- L. Llamas (2014, Set. 23). “Entradas Analógicas en Arduino” [Internet]. Disponible en https://www.luisllamas.es/entradas-analogicas-en-arduino/
- L. Del Valle (s.f.). “Programarfacil” [Internet]. Disponible en https://programarfacil.com/blog/arduino-blog/if-else-arduino/

V. MARCO TEORICO

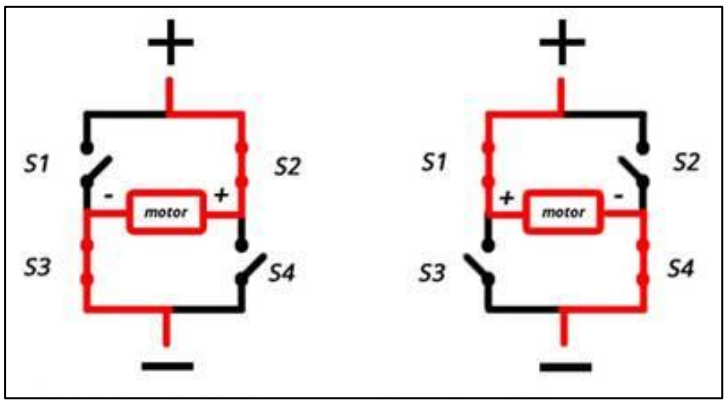
Control de motores con Arduino:

Arduino tiene pines de entrada y de salida para comunicarse físicamente con su entorno. Los pines de salida pueden dar una pequeña cantidad de energía que sirve por ejemplo para encender un LED, pero no pueden alimentar a un motor. Para ello necesitamos un circuito que haga de intermediario. Este circuito, el driver de motores o puente H, tomará energía de otra fuente (una pila, batería o equivalente) y siguiendo las instrucciones de Arduino hará funcionar el motor.



Puente H:

El circuito básico que permite las dos funciones que dijimos es un puente H. Un motor DC cambia el sentido de giro cuando invertimos los polos positivo y negativo en sus bornes, y este circuito juega con una serie de interruptores para hacer ese cambio de polaridad. Podemos construir un puente H con cuatro interruptores, pero lo que usaremos es un circuito integrado que tiene esos interruptores -o puertas- y las abrirá y cerrará siguiendo instrucciones del Arduino.



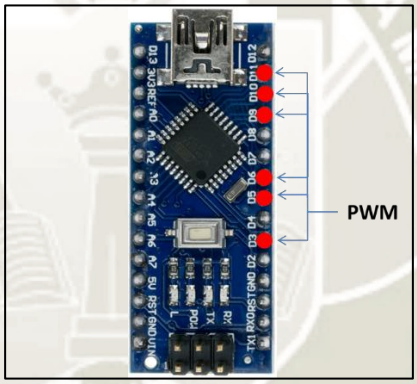
Señales PWM:

Un motor DC está preparado para funcionar a una tensión que nos dará el fabricante. Si le damos una tensión menor girará más despacio, llegando en el extremo a pararse. Así regulamos la velocidad.

Arduino no puede regular directamente los voltios que salen por un pin, pero utiliza una función llamada PWM que tiene como resultado una acción parecida a bajar la tensión. Lo importante y que nos interesa ahora es:

- Arduino puede generar señales PWM y así simular una tensión menor en voltios.
- Esta señal sirve para controlar motores a través de un puente.
- Los motores DC aceptan esta señal y podemos ajustar así la velocidad.

No todos los pines de Arduino pueden hacer esta función PWM. En los diagramas de funcionamiento de cada modelo puede verse cuáles la tienen y cuáles no.



Lecturas analógicas:

Una señal analógica es una magnitud que puede tomar cualquier valor dentro de un intervalo –Vcc y + Vcc. Por ejemplo, una señal analógica de tensión entre 0V y 5V podría valer 2,72V, o cualquier otro valor con cualquier número de decimales. Por contra, recordemos que una señal digital de tensión teórica únicamente podía registrar dos valores (en el ejemplo, 0V o 5V).

Por norma general en las tarjetas, las entradas analógicas son más escasas, más lentas y caras que las entradas digitales. En el caso de Arduino nano disponemos de 8 entradas analógicas que disponen de 10 bits de resolución, lo que proporciona 1024 niveles digitales, lo que a 5V supone una precisión de la medición de +-2,44mV.

Sentencia If - else:

Generalmente la sentencia condicional if se aplica en Arduino con números, pero se pueden utilizar otro tipo de datos. Podemos comparar temperatura, presión atmosférica, radiación ultravioleta, tiempo, intensidad, voltaje, nivel de agua y cualquier magnitud susceptible de ser medida.

Algo que creo que todos entendemos es la temperatura. Por ejemplo, podemos escribir un código donde dependiendo de la temperatura mueva un servomotor que activa un ventilador.

```
if(temperatura > 25){
  Servo.move(30);
}
```

En este caso estamos diciendo que siempre que la temperatura sea mayor que 25, mueva un servomotor. Pero también podríamos haber programado si la temperatura es menor que 30 o si es mayor o igual que 25.

Todo lo expuesto anteriormente son comparaciones. Lo único que hacemos es comparar dos valores y evaluamos si es verdadero o falso. Para poder comparar se utilizan los operadores de comparación. La sentencia if en Arduino permite utilizar 6 operadores de comparación:

> Mayor que

< Menor que

>= Mayor o igual que

<= Menor o igual que

== Igual a

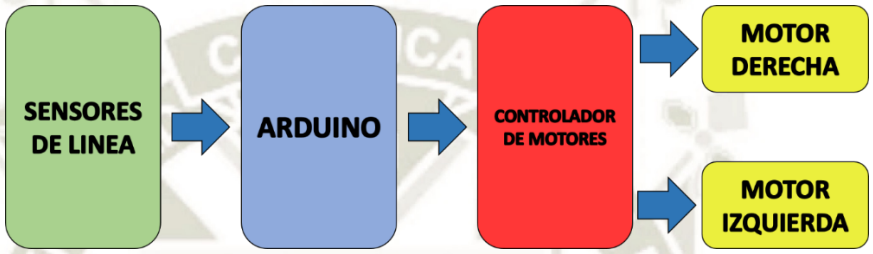
!= Diferente a

Los 4 primeros (mayor que, menor que, mayor o igual que o menor o igual que) son bastante explícitos. Sin embargo, los dos últimos (igual a y diferente a) requieren de una pequeña explicación.

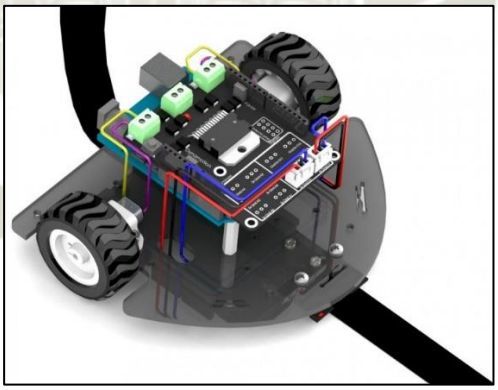
Robot seguidor de líneas:

Los robots seguidores de línea son robots muy sencillos, que cumplen una única misión seguir una línea marcada en el suelo normalmente de color negro sobre un fondo blanco. Son considerados los "Hola mundo" de la robótica. En esta sesión se presentarán las partes de un robot móvil seguidor de línea negra con fondo blanco, utilizando una placa de ARDUINO NANO.

Todos los rastreadores basan su funcionamiento en los sensores. Sin embargo, dependiendo de la complejidad del recorrido, el robot debe ser más o menos complejo (y, por ende, utilizar más o menos sensores).



Los seguidores de línea más simples utilizan 2 sensores, ubicados en la parte inferior de la estructura, uno junto al otro. Cuando uno de los dos sensores detecta el color blanco, significa que el robot está saliendo de la línea negra por ese lado. En ese momento, el robot gira hacia el lado contrario hasta que vuelve a estar sobre la línea.



VI. ACTIVIDADES

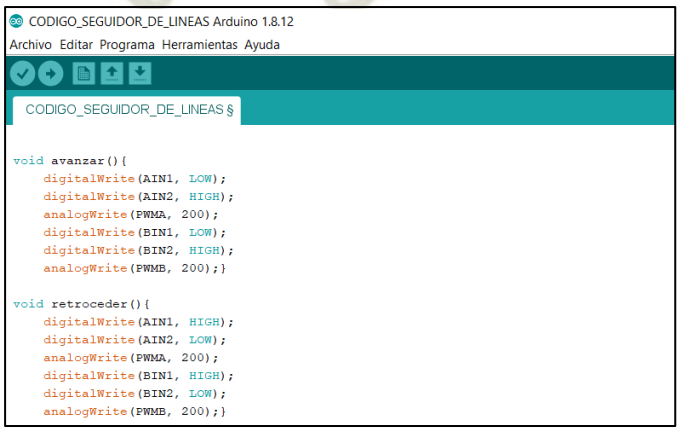
1. Vamos a declarar funciones que van a permitir al robot desplazarse utilizando los micromotores, para eso es necesario elaborar una tabla que nos permita entender el funcionamiento de nuestro driver de motores, el cual nos brinda **2 pines de dirección (señal digital) y 1 pin de velocidad (señal PWM) por motor**.

	MOTOR DERECHA			MOTOR IZQUIERDA		
	AIN1	**AIN2**	**PWMA**	**BIN1**	**BIN2**	**PWMB**
AVANZAR	HIGH	LOW	1-254	HIGH	LOW	1-254
RETROCEDER	LOW	HIGH	1-254	LOW	HIGH	1-254
DERECHA	HIGH	LOW	1-254	LOW	HIGH	1-254
IZQUIERDA	LOW	HIGH	1-254	HIGH	LOW	1-254
FRENAR	LOW	LOW	254	LOW	LOW	254

2. Sabiendo esos parámetros, procederemos a desarrollar la función **AVANZAR** con la sintaxis adecuada para cargarla en el código, se trabajará de la misma forma con el resto de las funciones **(retroceder, derecha, izquierda, frenar)**.

```
void avanzar()
{
  digitalWrite(AIN1, HIGH);
  digitalWrite(AIN2, LOW);
  analogWrite(PWMA, 150);
  digitalWrite(BIN1, HIGH);
  digitalWrite(BIN2, LOW);
  analogWrite(PWMB, 150);
}
```

3. Recordar que estas funciones serán declaradas al final del programa y que para hacer uso de ellas solo es necesario ingresar el nombre asignado después del **void**, por ejemplo si deseamos que el robot avance ingresaremos la sentencia **"avanzar();"**



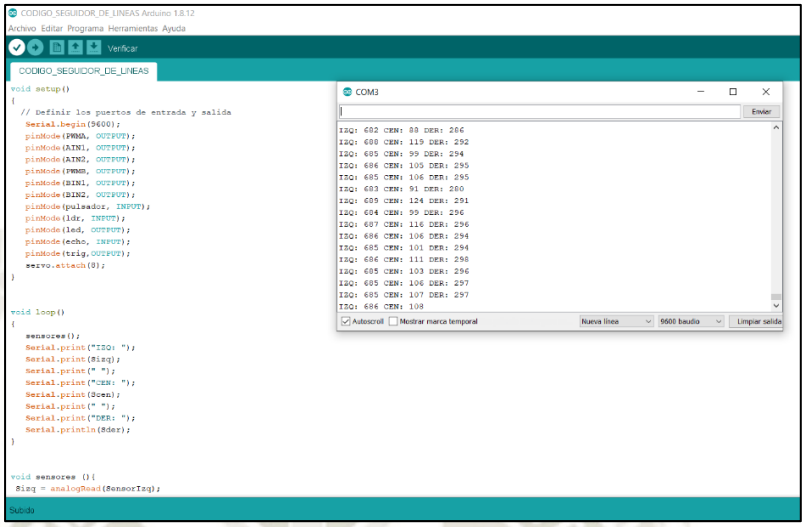
4. Sera necesario elaborar una función denominada **void sensores()**, la cual se encargará de acondicionar las señales analógicas brindadas por los sensores de línea. Utilizaremos la instrucción **delayMicroseconds()** para agregar un retardo de 250µs, con la finalidad de evitar errores en la detección de la línea y devolver 3 variables **"Sizq", "Sder" y "Scen"** con las que se podrán trabajar.

```
void sensores()
{
 Sizq = analogRead(SensorIzq);
 delayMicroseconds(250);
 Sder = analogRead(SensorDer);
 delayMicroseconds(250);
 Scen = analogRead(SensorCen);
 delayMicroseconds(250);
}
```

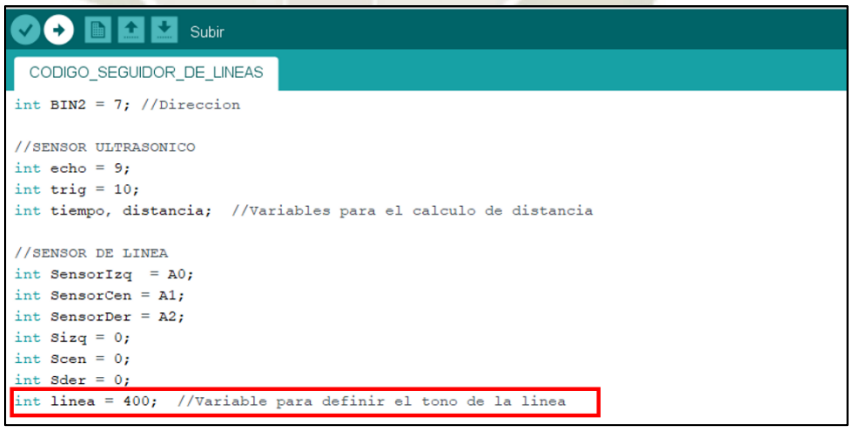
5. Ahora podemos cargar en el **void loop()** este código que nos permitirá visualizar las lecturas de los 3 sensores de línea presentes en el robot.

```
void loop()
{
  sensores();
  Serial.print("IZQ: ");
  Serial.print(Sizq);
  Serial.print(" ");
  Serial.print("CEN: ");
  Serial.print(Scen);
  Serial.print(" ");
  Serial.print("DER: ");
  Serial.println(Sder);
}
```

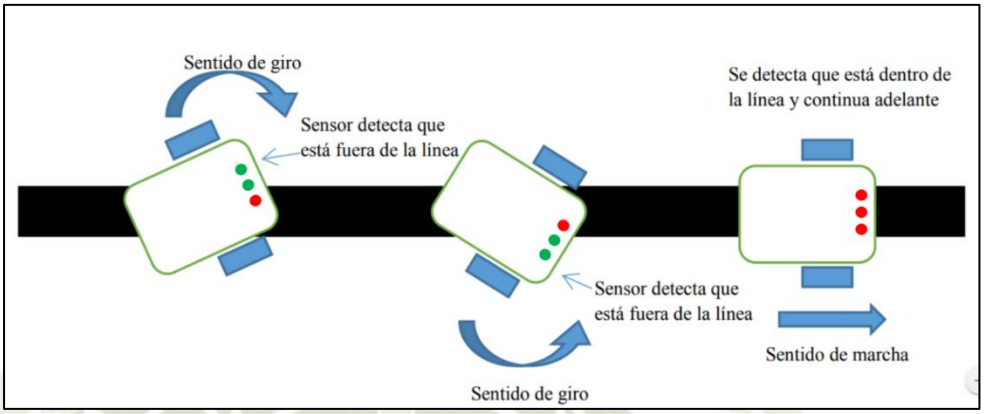
6. Para poder visualizar las lecturas de los sensores, el robot debe estar conectado a la PC y el **monitor serie** debe estar abierto como se muestra a continuación. **Identificar el valor que nos otorgan los sensores al detectar la línea negra.**



7. Luego de identificar el valor que nos otorga la línea negra, debemos almacenarla en la variable **línea** recordando que los valores más cercanos a **1024** son generalmente valores de colores oscuros. Sabiendo eso y si el valor de la línea negra es de aproximadamente **700**, es recomendable asignar un valor menor como podría ser **550** para tener un margen más amplio de detección de la línea.



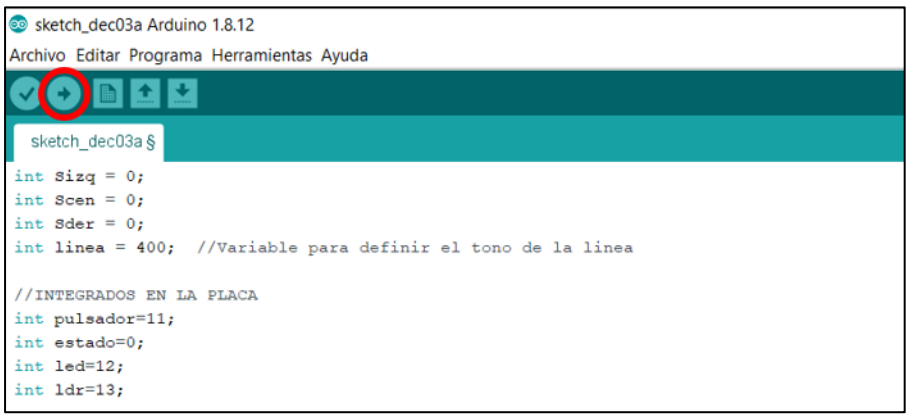
8. Una vez probada la función **void sensores()** podemos invocarla en el **void loop()** y trabajar con las variables que devuelve haciendo uso de las condicionales **"if"** y **"else"**. Este grafico nos ayudara a entender las 3 posibles condiciones que se presentan al seguir una línea cuando se utiliza 3 sensores.



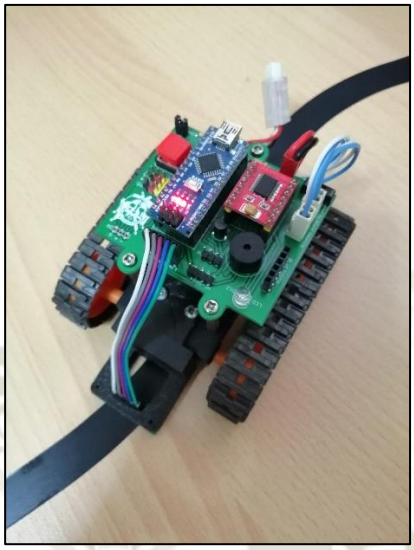
9. Usando de referencia el grafico anterior, podemos empezar a definir en forma de instrucciones las 3 situaciones descritas anteriormente, también podemos añadir un **else** que se encargara de definir una condición que se ejecutara solo cuando no se cumpla una de las condiciones anteriores.

```
void loop()
{
  sensores();
  if(Sizq<=linea)       {izquierda();delay(1);}
  if(Scen<=linea)       {avanzar();delay (1);}
  if(Sder<=linea)       {derecha();delay (1);}
  else          {avanzar();delay (1);}
}
```

10. Realizados los ajustes anteriores podemos proceder a subir el programa al robot, haciendo clic sobre la flecha "→" situada al costado del botón verificar.



11. Realizados los pasos anteriores, podemos desconectar el robot de la PC y comprobar que el programa se está ejecutando de forma correcta, el módulo debería ser capaz de seguir la línea trazada en el circuito definido por el docente.



12. Implementar instrucciones más restrictivas con la ayuda del operador Booleano "**AND**" el cual solo es cierto cuando ambas sentencias cumplen la condición, permitiendo elaborar condiciones cuando 2 sensores detectan la línea al mismo tiempo o inclusive los 3 sensores.

```
void loop()
{
  sensores();
  if((Sizq>=linea)&&(Sder>=linea)&&(Scen>=linea)){ //definir accion }
  if((Sizq>=linea)&&(Scen>=linea)&&(Sder<=linea)){ //definir accion }
  if((Sder>=linea)&&(Scen>=linea)&&(Sizq<=linea)){ //definir accion }
}
```

13. Modificar el tipo de giro del módulo, ya que implementamos por defecto el autogiro, en el que ambas ruedas giran en sentido contrario permitiendo al módulo girar en lugares de espacio reducido, **es recomendable que un robot seguidor de líneas siempre se desplace hacia adelante**, es por eso que el tipo de giro aplicado será el de **pivote**, el cual mantiene una rueda detenida mientras que la otra describe el arco que define el giro.

```
void derecha()
{
   digitalWrite(AIN1, HIGH);
   digitalWrite(AIN2, LOW);
   analogWrite(PWMA, 0);              //Rueda detenida
```

```
  digitalWrite(BIN1, LOW);
  digitalWrite(BIN2, HIGH);
  analogWrite(PWMB, 200); //Rueda encargada de ejecutar el giro
}
```

VII. CONCLUSIONES

Emita al menos cinco conclusiones acerca de las funciones **void()** y las instrucciones **if anidadas.**

ANEXO 4. GUIA 4: ROBOT CONTROLADO POR BLUETOOTH

UNIVERSIDAD CATÓLICA DE SANTA MARÍA **ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA**		
CODIGO ASIGNATURA		GUÍA DE LABORATORIO Nº 04
MODULO I ROBOT CONTROLADO POR BLUETOOTH		**Docente (s):** Bruno Fernando Valencia Rodriguez Fecha: 2021.05.04.

I. OBJETIVOS DE LABORATORIO

a. Elaborar una aplicación de control bluetooth.
b. Controlar el robot mediante una aplicación bluetooth.

II. TEMAS A TRATAR

- Creación de aplicaciones Android utilizando MIT APP INVENTOR.
- Bluetooth HC-05.
- Comunicación bluetooth con Arduino.

III. HERRAMIENTAS

1. Computador con ARDUINO IDE.
2. Modulo educativo de robótica.
3. Cable USB.
4. Celular Android.
5. Guía de laboratorio.

IV. BIBLIOGRAFIA

- Instituto de Tecnología de Massachusetts, (s.f.). "**MIT APP INVETOR ". [Internet]. Disponible en** https://appinventor.mit.edu/about-us

- DIYMakers (2014, Feb. 20). "Crear App para Arduino con App Inventor" [Internet]. Disponible en http://diymakers.es/crear-app-para-arduino-con-app-inventor/

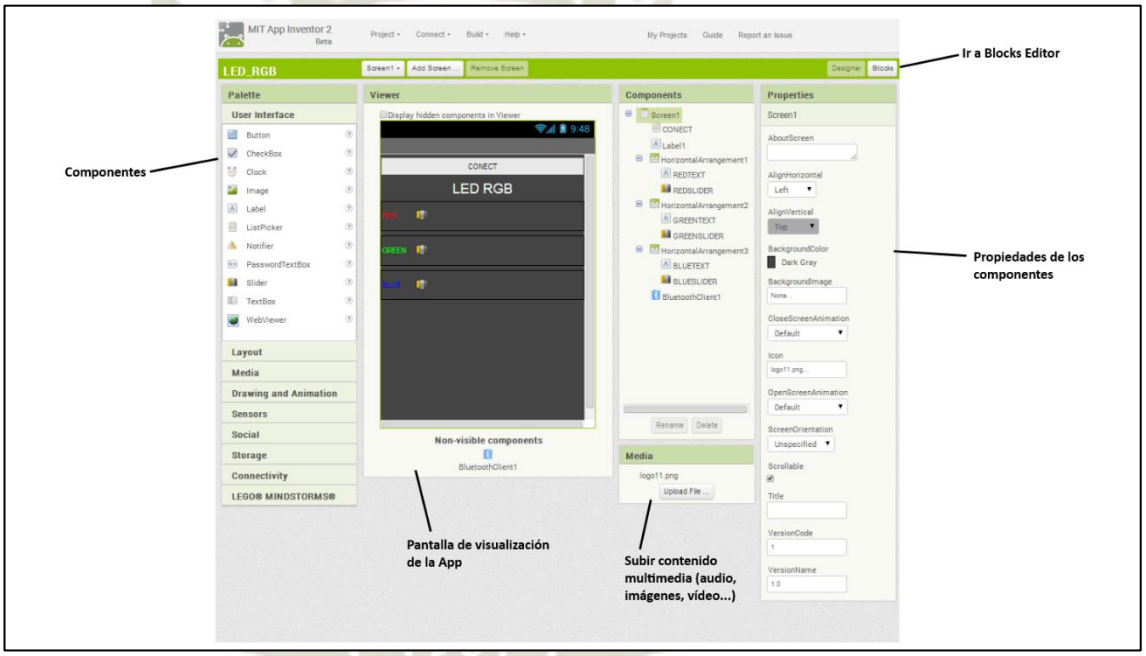
- DIYMakers (2014, Feb. 03). "Arduino + Bluetooth" [Internet]. Disponible en http://diymakers.es/arduino-bluetooth/

- A. Ricoy (2011, Feb. 26). "App Inventor en Español" [Internet]. Disponible en https://sites.google.com/site/appinventormegusta/conceptos/bloques

V. MARCO TEORICO

MIT APP INVENTOR:

MIT App Inventor es un entorno de programación visual e intuitivo que permite a todos, incluso a los niños, crear aplicaciones completamente funcionales para teléfonos inteligentes y tabletas. Aquellos que son nuevos en MIT App Inventor pueden tener una primera aplicación simple en funcionamiento en menos de 30 minutos. Gracias a su herramienta basada en bloques facilita la creación de aplicaciones complejas y de alto impacto en mucho menos tiempo que los entornos de programación tradicionales. El proyecto MIT App Inventor busca democratizar el desarrollo de software empoderando a todas las personas, especialmente a los jóvenes, a pasar del consumo de tecnología a la creación de tecnología.



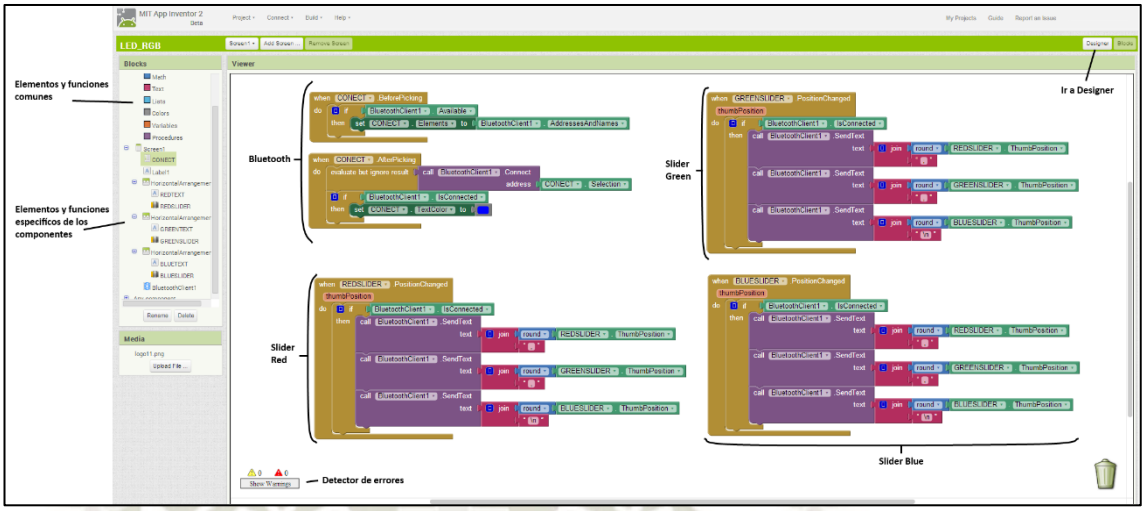
Bloques:

Con el Editor de bloques vamos a definir cómo se comportará la aplicación. Estableceremos lo que los componentes deben hacer y cuándo hacerlo. Por ejemplo, que debe ocurrir cuando el usuario pulsa un botón.

El Editor de bloques se ejecuta en una ventana independiente. Al hacer clic en Abrir el editor de bloques de la ventana de diseño, el archivo del programa del Editor de bloques se descargará y a continuación se debe ejecutar.

El Editor de bloques tiene dos fichas en la esquina superior izquierda: Built-In (incorporados) y My Blocks (mis bloques). Los botones de debajo de cada ficha se amplían y se muestran los bloques cuando se hace clic. Los bloques Built-In son el conjunto estándar de bloques que están disponibles para cualquier aplicación que

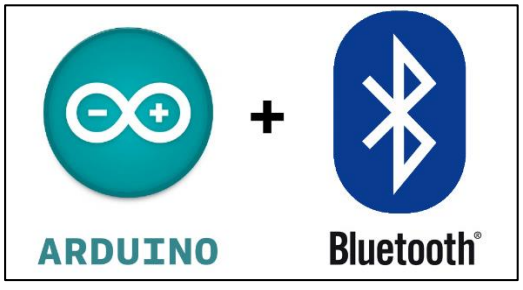
construyas. Los bloques My Blocks contienen bloques específicos que están vinculados al conjunto de componentes que has elegido para tu aplicación.



Comunicación Bluetooth:

El Bluetooth es un estándar de comunicación inalámbrica que permite la transmisión de datos a través de radiofrecuencia en la banda de 2,4 GHz. Existen muchos módulos Bluetooth para usarlos en nuestros proyectos de electrónica, pero los más utilizados son los módulos de JY-MCU, ya que son muy económicos y fáciles de encontrar en el mercado. Son módulos pequeños y con un consumo muy bajo que nos permitirán agregar funcionalidades Bluetooth a nuestro Arduino. Estos módulos contienen el chip con una placa de desarrollo con los pines necesarios para la comunicación serie.

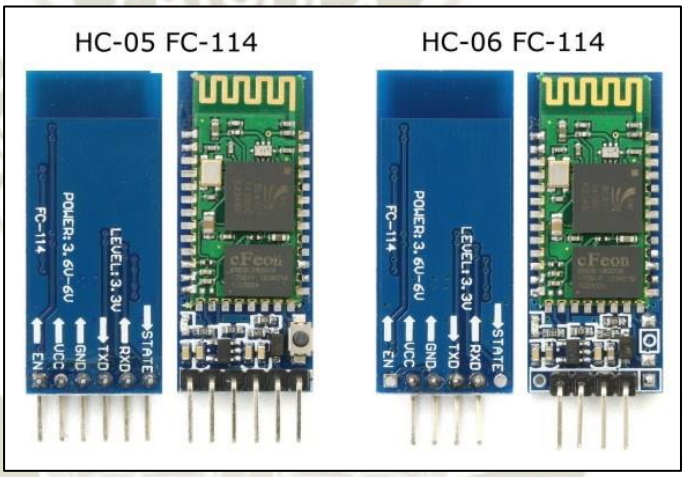
- Vcc: Alimentación del módulo entre 3,6V y 6V.
- GND: La masa del módulo.
- TXD: Transmisión de datos.
- RXD: Recepción de datos a un voltaje de 3,3V.
- KEY: Poner a nivel alto para entrar en modo configuración del módulo (solo el modelo HC-05)
- STATE: Para conectar un led de salida para visualizar cuando se comuniquen datos.



Bluetooth HC-05:

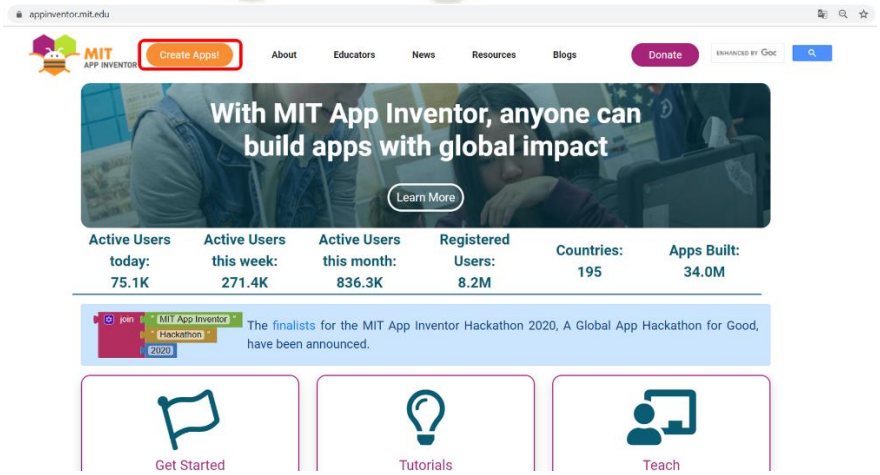
El módulo bluetooth HC-05 utiliza el protocolo UART RS232 Serial, y es ideal para el desarrollo de aplicaciones inalámbricas, debido a su bajo costo, además se puede encontrar el chip para soldarlo uno mismo o ya soldado sobre un módulo (breakout), con los pines para realizar la comunicación y un led indicador.

Cabe destacar que el HC-05 tiene un hermano, el módulo HC-06, y que aparentemente son iguales. Una simple diferencia es que el módulo HC-06 funciona como Slave y el HC-05 como Master y Slave. Físicamente se diferencian por el número de pines. En el HC-06 tiene un conector de 4 pines mientras que el HC-05 trae uno de 6 pines.

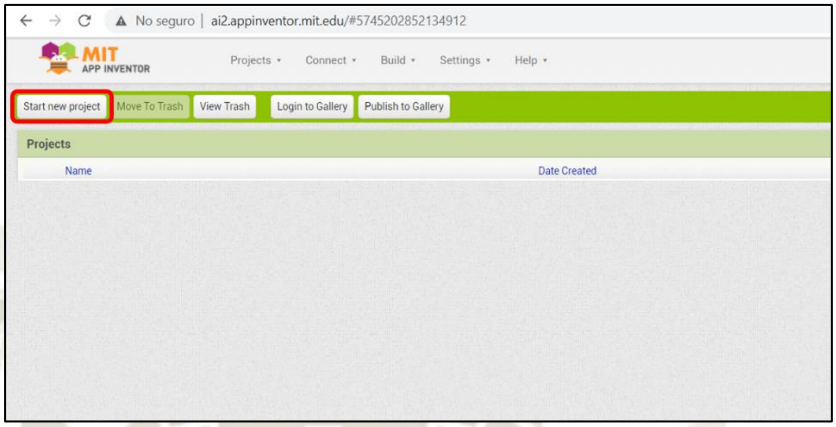


VI. ACTIVIDADES

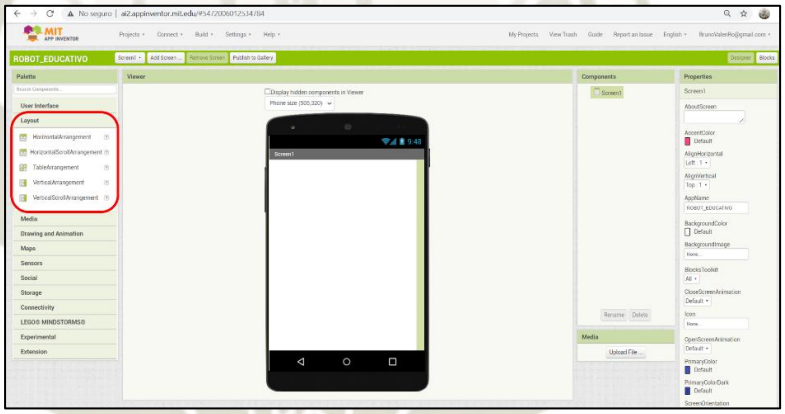
1. A continuación, se detallarán los pasos a seguir para la elaboración de la aplicación bluetooth que nos permitirá controlar nuestro robot, es necesario contar con una cuenta de **Gmail** e ingresar a la página del **MIT APP INVENTOR** https://appinventor.mit.edu/, debemos seleccionar la opción "**Create Apps!**" situada en la parte superior izquierda como se aprecia en la imagen.



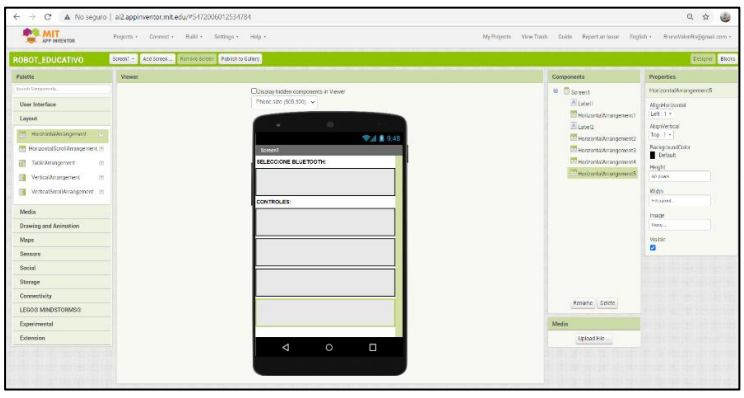
2. Para iniciar con la creación de la aplicación debemos seleccionar el botón **"Start new project"** y nos direccionara a la ventana para diseñar la interfaz de usuario de nuestra aplicación



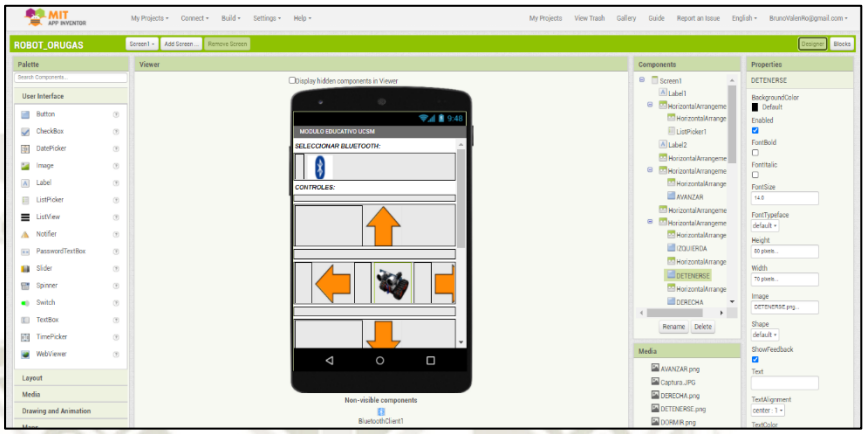
3. Lo primero que debemos hacer es designar los espacios necesarios para colocar imágenes y botones, para eso utilizaremos la paleta de **"Layout"** situada en la parte izquierda de la pantalla.



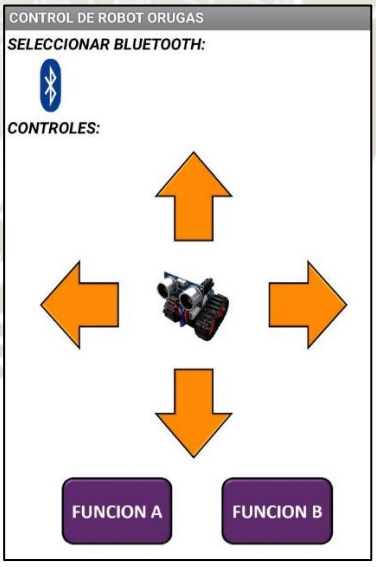
4. Utilizaremos **"HorizontalArragement"** para asignar espacios de **60 pixeles de altura y largo adapto a la pantalla (fill parent)**, donde más adelante asignaremos los botones de **selección de bluetooth y movimiento del robot.**



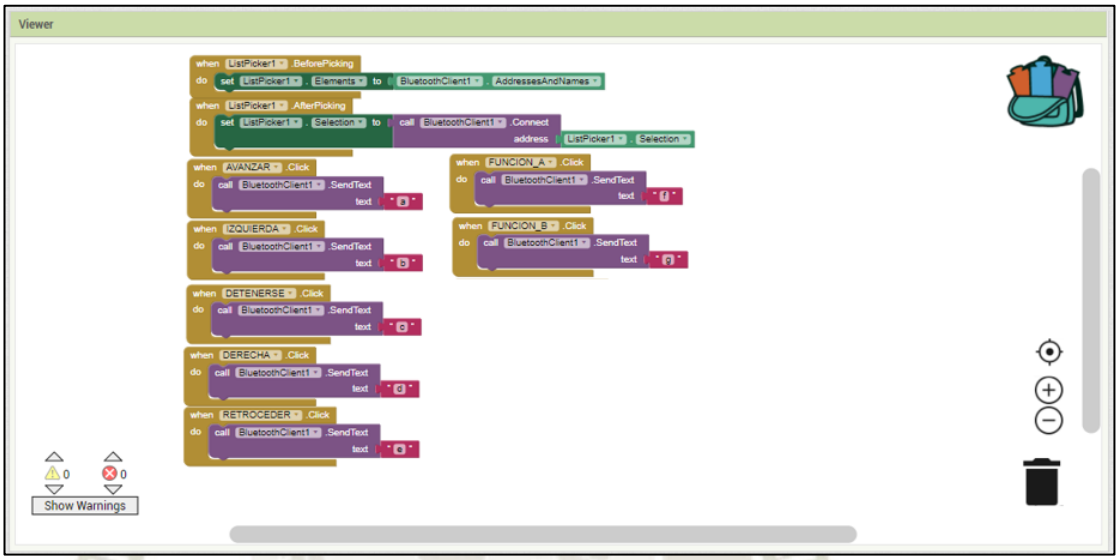
5. La paleta de **"User Interface"** cuenta con una opción llamada **"Button"**, que nos permite añadir los botones necesarios para controlar el movimiento de nuestro robot, definidos los botones es necesario asignar una imagen para identificar la función de cada uno, debemos seleccionar **"Image"** en la parte inferior derecha e importar una imagen**.**



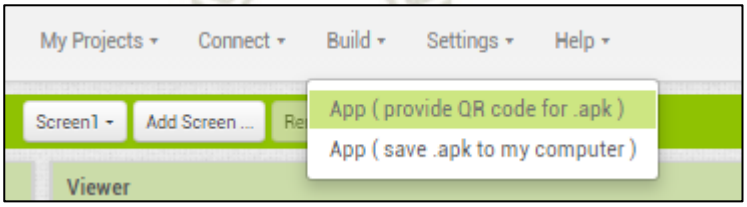
6. Siguiendo los pasos anteriores debemos terminar con una aplicación parecida a la que se muestra en la siguiente imagen, que nos permitirá enlazar el módulo bluetooth del robot al celular, contará con opciones para desplazamiento del robot (avanzar, retroceder, izquierda, derecha y frenar) y también 2 botones asignados a funciones personalizadas.



7. Una vez definida la interfaz de usuario, seleccionaremos la pestaña de bloques para programar el comportamiento de los botones y definir los comandos que estos enviaran al módulo. Primero es necesario **establecer la conexión con el cliente bluetooth para iniciar la comunicación**, luego se crearán bloques para cada botón a los cuales se asignará un carácter del alfabeto, el cual al ser recibido por el robot será interpretado y ejecutará una función determinada.



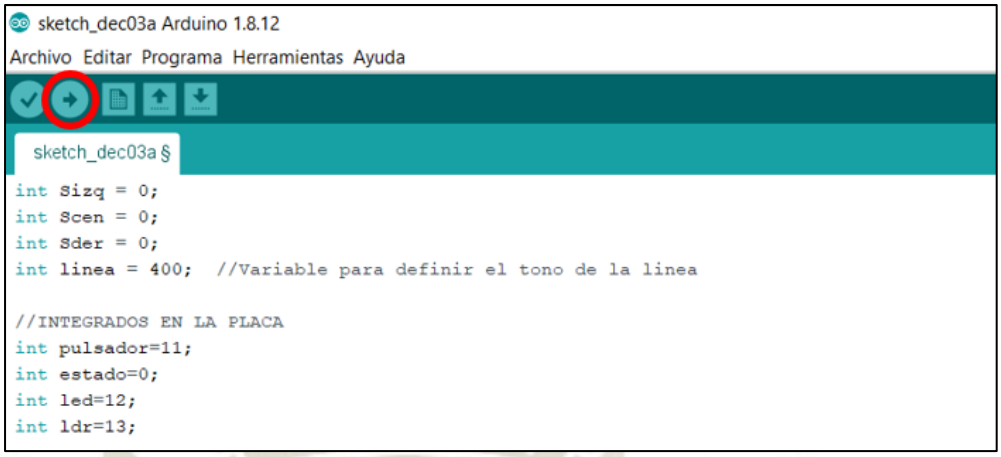
8. Teniendo las dos partes de la aplicación bluetooth completadas, se procede a la construcción de la aplicación android con la extensión **“.apk”**, para eso nos dirigimos a la pestaña **“Build”** en el entorno de diseño de MIT App Inventor y contamos con dos opciones; la opción de **generar un código QR** para escanearlo con un celular o **descargar el archivo** e instalarlo de forma manual en el celular. Es importante recalcar que todo el progreso se almacena en la aplicación online, permitiendo el acceso desde cualquier lugar con acceso a internet y así poder seguir escalando la aplicación en función a las futuras aplicaciones que se deseen implementar.



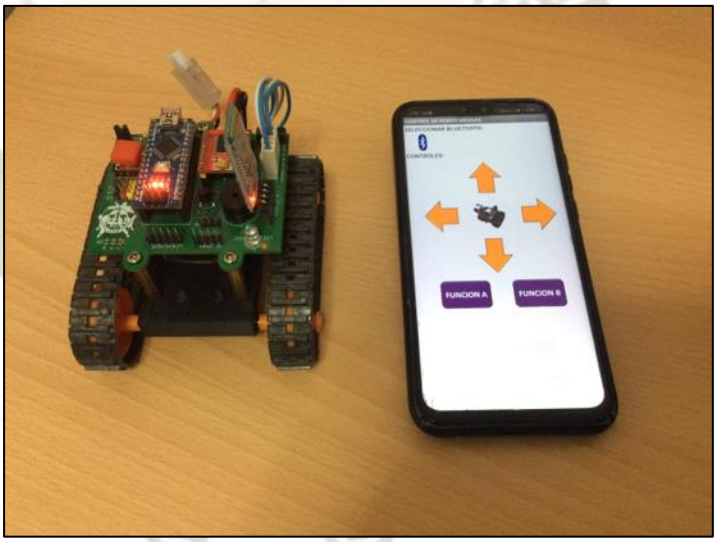
9. Generaremos un código en Arduino que ejecute una acción dependiendo del carácter que se reciba en el puerto serial bluetooth, para eso utilizaremos la instrucción **"Serial.available"** para poder leer los caracteres que nos enviara la aplicación bluetooth, luego definiremos instrucciones condicionales del tipo **"if"** en función al carácter recibido.

```
void loop()
{
  if(Serial.available()>0){       // lee el bluetooth y almacena en estado
     estado = Serial.read();
  }

if(estado=='a')   // Boton Adelante
{adelante();}

if(estado=='b')   // Boton Izquierda
{izquierda();}

if(estado=='c')   // Boton Detenerse
{parar();}

if(estado=='d')   // Boton Derecha
{derecha();}

if(estado=='e')   // Boton Atras
{atras();}

if (estado =='f') // Funcion A
{}

if (estado =='g') // Funcion B
{}
}
```

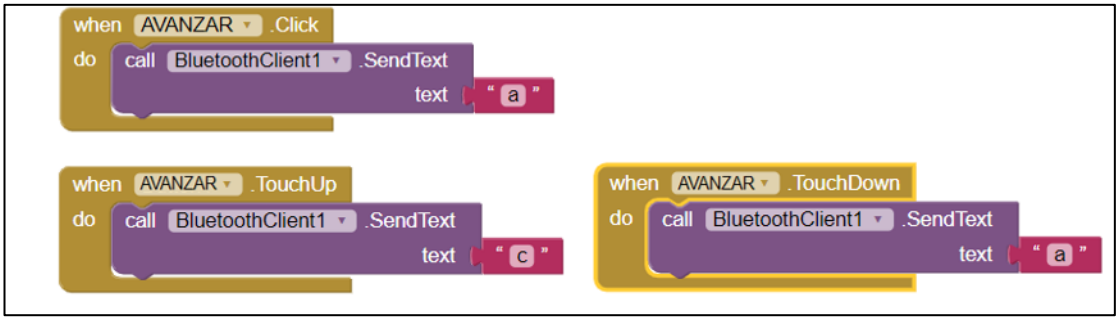
10. Realizados los ajustes anteriores podemos proceder a subir el programa al robot, haciendo clic sobre la flecha **"→"** situada al costado del botón verificar.



11. Realizados los pasos anteriores, podemos desconectar el robot de la PC y comprobar que el programa se está ejecutando de forma correcta, el módulo debería ser capaz de realizar la acción especificada en la aplicación celular.



12. Cambiar la función **"Click"** de los botones por **"TouchUp"** y **"TouchDown"**, estos simulan un botón real y solo funcionan al mantenerse presionados.



VII. CONCLUSIONES

Emita al menos cinco conclusiones acerca del entorno de programación MIT APP INVENTOR y sus distintas aplicaciones.