UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE INGENIERÍA AUTOMOTRIZ

DISEÑO Y CONSTRUCCIÓN DE UN VEHÍCULO AUTÓNOMO A ESCALA EQUIPADO CON TECNOLOGÍA LIDAR 2D PARA LA NAVEGACIÓN AUTÓNOMA EN UN ENTORNO CON OBSTÁCULOS PREESTABLECIDOS PARA EL LABORATORIO DE AUTOTRÓNICA DE LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO CAMPUS SUR.

Trabajo de titulación previo a la obtención del
Título de Ingeniero Automotriz

AUTORES: FABRIZZIO ALDAIR GALÁRRAGA ALARCÓN
PABLO ISAAC INAQUIZA TIPÁN

TUTOR: CARLOS ALBERTO CARRANCO QUIÑÓNEZ

Quito - Ecuador
2023

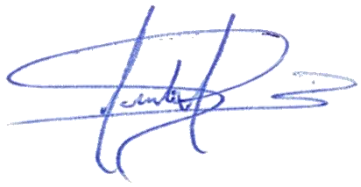
CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

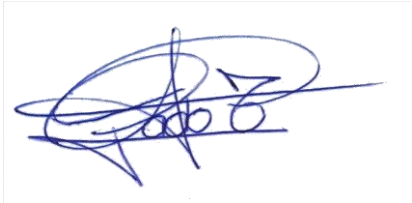
Nosotros Fabrizzio Aldair Galárraga Alarcón con documento de identificación N° 1726010778 y Pablo Isaac Inaquiza Tipán con documento de identificación N° 1718475393 manifiestos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Quito, 25 de septiembre del año 2023

Atentamente,





Fabrizzio Aldair Galárraga Alarcón
1726010778

Pablo Isaac Inaquiza Tipán
1718475393

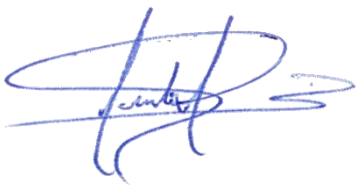
CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

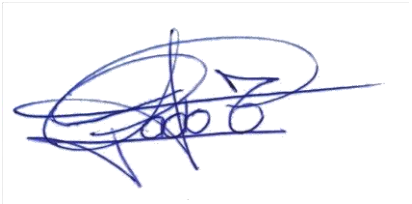
Yo Fabrizzio Aldair Galárraga Alarcón con documento de identificación No. 1726010778 y Pablo Isaac Inaquiza Tipán con documento de identificación No.1718475393, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto Técnico: "Diseño y construcción de un vehículo autónomo a escala equipado con tecnología LIDAR 2d para la navegación autónoma en un entorno con obstáculos preestablecidos para el laboratorio de autotrónica de la universidad politécnica salesiana sede quito campus sur.", el cual ha sido desarrollado para optar por el título de Ingenieros Automotrices, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana

Quito, 25 de septiembre del año 2023

Atentamente,





Fabrizzio Aldair Galárraga Alarcón
1726010778

Pablo Isaac Inaquiza Tipán
1718475393

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Carlos Alberto Carranco Quiñónez con documento de identificación N° 1713629564, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DISEÑO Y CONSTRUCCIÓN DE UN VEHÍCULO AUTÓNOMO A ESCALA EQUIPADO CON TECNOLOGÍA LIDAR 2D PARA LA NAVEGACIÓN AUTÓNOMA EN UN ENTORNO CON OBSTÁCULOS PREESTABLECIDOS PARA EL LABORATORIO DE AUTOTRÓNICA DE LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO CAMPUS SUR., realizado por Fabrizzio Aldair Galárraga Alarcón con documento de identificación No. 1726010778 y Pablo Isaac Inaquiza Tipán con documento de identificación No.1718475393 obteniendo como resultado final el trabajo de titulación bajo la opción: Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 25 de septiembre del año 2023

Atentamente,

Ing. Carlos Alberto Carranco Quiñónez, MsC
1713629564

DEDICATORIA

Agradezco a mi madre por su constancia y perseverancia a lo largo de mi formación académica

A mi padre quien supo darme sus consejos y sabiduría para poder seguir adelante sin rendirme ante cualquier situación

A mi hermano quien me inculcó el valor del trabajo y dedicación que se debe dar en cada momento laboral, por su apoyo incondicional y solidaridad en las peores situaciones donde me sentí solo, le agradezco por ser parte de mi entorno familiar

Finalmente, a mi hermana y mi novia por estar en mis momentos de victorias y tristezas, expresó mis más sinceros agradecimientos.

Pablo Isaac Inaquiza Tipan

Dedico el presente proyecto a mis familiares en general porque cada uno y una han ido poniendo su granito de arena con sus enseñanzas y apoyo para llegar a ser la persona que soy hoy.

A mi madre que ha sabido guiarme en mi formación profesional y con sus consejos he tomado las decisiones más importantes de mi vida.

A mi hermana que ha sido mi razón de seguir adelante dándome la motivación para encontrar la solución a cada problema que se me ha presentado.

Fabrizzio Aldair Galárraga Alarcón

AGRADECIMIENTO

Mi agradecimiento principal a Dios por guiarme en esta vida y en mis estudios, mis ingenieros quienes me inculcaron todo su conocimiento en cada una de sus clases impartidas.

A mis compañeros de clases, quienes dieron un poco de su sabiduría para mejorar mi estilo de estudio.

Finalmente, a la Universidad Politécnica Salesiana quien impartió su misión y visión como institución.

Pablo Isaac Inaquiza Tipan

Le agradezco muy profundamente a mi tutor el Ing. Carlos Carranco, ya que me motivo a profundizar en la electrónica y por esa razón el presente proyecto tomó forma también gracias a sus consejos y correcciones.

Agradezco a la Universidad Politécnica Salesiana en especial a los docentes de la Carrera de Ingeniería Automotriz por brindarme durante toda mi formación las herramientas necesarias para tener los conocimientos y habilidades que me han ayudado a encontrar oportunidades para mi crecimiento laboral.

Fabrizzio Aldair Galárraga Alarcón

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

INDICES DE TABLAS

RESUMEN

La autonomía y evolución de los vehículo está en constante progreso para una mejora tanto de los peatones como el de usuario que están detrás del volante , para generar una mayor seguridad y evitar riesgos de mayor grado como : accidentes , muertes y daños a espacio público ,por parte la imprudencia de los conductores y además de su falta de conocimientos de la reglas de tránsito , se establece una metodología basada en un mapeo y generación de rutas para los vehículos que no pueden manejar o tomar acciones en cuestión de segundos para evitar un accidente.

La aplicación de sensores para la detección de objetos o materiales físicos está destinada por parte del sensor LIDAR, al ser una tecnología aplicada, que utiliza pulsos láser para medir distancias y crear representaciones tridimensionales del entorno, de esta manera se podrá identificar y corregir la trayectoria autónoma, refleja correcciones debido a la programación por parte de una placa receptora de datos, el Arduino Mega es la opción más viable para este tipo de correcciones y recepción de datos. Así se aplicará una reducción en cuestión de momentos y se evitará accidentes en los ciclos de conducciones de las personas. Los sensores LIDAR y los radares tienen la especificación en ondas electromagnéticas donde cada uno de estos sensores mide una longitud de onda corta dependiendo de su programación y tipo de sensor mejoran la resolución y pueden ser graficadas en imágenes de dos dimensiones y tres dimensiones , nuestra aplicación está destinada principalmente a la captación de distancias y rangos destinados por los objetos captados, además de hacer el uso de un prototipo de ensamblado el cual podrá ser guiado y trasladado por un punto específico de trayectoria , este tipo de funcionamiento está basado en condiciones y lugares específicos al ser considerado un prototipo de nivel uno y sus propiedades no son superiores a los demás sensores que pueden captar sonido y vibraciones.

Hoy en día la aplicación de este sensor busca la resolución de un índice y control de los niveles de conducción de las personas es así que nuestro proyecto está destinado a la construcción de un prototipo a escala que podrá ser aplicado en un futuro en la industria automotriz como un nuevo sensor de acompañamiento para la conducción y trayectorias auto dirigidas el cual podrá corregir nuestro tipo de conducción y evitar accidentes de tránsito.

ABSTRACT

The autonomy and evolution of the vehicles is in constant progress to improve both pedestrians and users behind the wheel, to generate greater safety and avoid higher-level risks such as: accidents, deaths and damage to public spaces, due to the imprudence of the drivers and in addition to their lack of knowledge of traffic rules, a methodology based on the mapping methodology and generation of routes for vehicles that cannot drive or take actions in a matter of seconds to avoid a accident.

The application of sensors for the detection of physical objects or materials is intended by the LIDAR sensor, as it is an applied technology that uses laser pulses to measure distances and create three-dimensional representations of the environment, in this way it will be possible to identify and correct the trajectory. autonomously, it reflects corrections due to programming by a data receiver board, the Arduino Mega is the most viable option for this type of corrections and data reception. In this way, a reduction will be applied in a matter of moments and accidents in people's driving cycles will be avoided. The leading sensors and radars have the specification in electromagnetic waves where each of these sensors measures a short wavelength depending on their programming and type of sensor improve the resolution and can be plotted in 2-dimensional and 3-dimensional images, our application is mainly intended for capturing distances and ranges intended by the captured objects, in addition to making use of an assembled prototype which can be guided and moved by a specific point of trajectory this type of operation is based on specific conditions and places as it is considered a level one prototype and its properties are not superior to other sensors that can capture sound and vibrations.

Today the application of this sensor seeks the resolution of an index and control of people's driving levels, so our project is aimed at building a scale prototype that can be applied in the future in the automotive industry. as a new monitoring sensor for driving and self-guided trajectories by this type of sensor and its sending of data to a central board, which will be able to correct our type of driving and avoid traffic accident

INTRODUCCIÓN

En la actualidad, gran parte de la tecnología aplicada está vinculada a la industria automotriz, la comunicación e información son la base utilizada en los vehículos modernos, es decir, para mejorar el rendimiento de funciones y mejorar los componentes de: seguridad, comodidad, ergonomía y confort. Establecemos sistemas secundarios para agilizar la interfaz entre el automóvil y el conductor, por tanto, la industria automotriz opta por los sistemas LIDAR de mapeo y percepción de objetos. “Algunas características de LIDAR son la longitud de onda, el consumo de energía, el ancho de pulso, los filtros ópticos y las capacidades informáticas” (Roriz et al., 2014)

Muchas de sus aplicaciones y capacidades en base a la percepción y movilidad de datos en tiempo real, están emitidos y configurados por los componentes y partes designadas para la recopilación de datos presentados en el Capítulo 1 del presente trabajo. Por otro lado, se hablará de la comunicación que existirá entre las partes móviles (Físicas) y la eléctrica. Se implementará un esquema similar al auto cotidiano con diferentes sistemas dependiendo del campo donde será expuesto y usado para las pruebas de campo. Además de su sistema configurado y recolección de datos que estarán vinculados con la carrocería prototipo del automóvil serán presentados en el Capítulo 2.

En el capítulo 3, analizaremos la interfaz, comunicación y programación implicada para la interacción del dispositivos móvil, al recopilar los datos en tiempo real y procesarlos con la interfaz proteus, obtenido resultados de conexión del sensor LIDAR en conjunto con su funcionamiento, es decir, verificar una trayectoria ah detalle con los parámetros de movimiento y redirigirlos generalmente a los comando previamente programados en el software .Además apreciaremos los componentes de armado y fabricación de las partes como: carrocería, ejes, ruedas, etc.

Seguidamente, en el Capítulo 4, Se aplicará una variación en las pruebas de campo, evaluando el rendimiento de la carrocería, partes móviles, velocidad de respuesta en tiempo real, etc. Determinado le eficacia tanto de la programación y armado del prototipo.

PROBLEMA

La movilidad y conducción abarcan gran cantidad de las actividades diarias en la población actual, pero un gran régimen de tránsito inapropiado son el resultado de los riesgos automovilísticos. A nivel mundial se registran casos de accidentes basados en "los factores condicionantes que se asocian a malas conducciones del vehículo, a factores intrínsecos del conductor (habilidad, condición emocional y física, presencia de distractores), a la poca visibilidad en las carreteras, a la ingesta de alcohol, al exceso de velocidad y a las malas condiciones de la carreteras, en las que intervienen la mala iluminación, el ancho de los carriles, la ausencia de espaldones, la mala señalización, y las condiciones climáticas de la zona." (Rica et al., 2008). Además, se reconoce que parte de la seguridad esta infringida por situaciones basadas en los factores humados, es decir, una falta de principios y condiciones sobre la conducción cotidiana aplicada hasta la actualidad.

Temas de seguridad y accidentes, son más susceptibles en América Latina, están destinadas a datos estadísticos donde "fuentes como la Organización Mundial de la Salud (OMS), algunos otros componentes de las Naciones Unidas y otras. Por otra parte, el número de muertos por accidentes de tránsito a nivel mundial es de unos 1.2 millones de personas cada año. Los accidentes de tráfico constituyen la segunda causa de muerte para personas entre los 5 y 29 años y la tercera para personas entre los 30 y 44 años." Se plantea una muestra estadística para entender el crecimiento de una falta de precaución e inseguridades de las personas al momento de transitar por las carreteras, "Para realizar una mejor comparación esta se hace en base a indicadores como los muertos por a cada 10.000 vehículos, o por muertos por cada 100.000 habitantes" (Planzer, 2005). La tasa de mortalidad por desastres automovilísticos es un claro indicador de la falta de conciencia por parte de las personas al momento de la conducción, para entender esta tendencia de números de accidentes, se toma en consideración zonas donde se cuenta con la mayor información de incidentes automovilísticos, donde países como: Nicaragua, Colombia Venezuela, Brasil y Ecuador. Al considerar que los índices pueden aumentan cada año, el dato está basado entre los años 1990 y 2016. El mayor aumento por parte de Latino América esta atribuido a Colombia con una tasa de incremento de 225.000 accidentes en el año 2000 y por parte de Ecuador tenemos un rango de 15.000 accidentes (muertes, lesiones y accidentes).

OBJETIVO GENERAL.

Diseñar y construir un vehículo autónomo a escala equipado con tecnología LIDAR 2D para la navegación autónoma en un entorno con obstáculos preestablecidos para el laboratorio de Autotrónica de la Universidad Politécnica Salesiana Sede Quito Campus Sur.

OBJETIVOS ESPECÍFICOS.

- Construir un vehículo a escala donde se implementará los sistemas de conducción autónomo.
- Instrumentar el sensor LIDAR aplicando un programa en JAVA para poner en operación el prototipo autónomo.
- Analizar el comportamiento del sensor LIDAR mediante la implementación de una placa Arduino.
- Establecer pruebas de campo utilizando objetos de diferente geometría con el propósito de evaluar la eficiencia de funcionamiento.

MARCO TEÓRICO

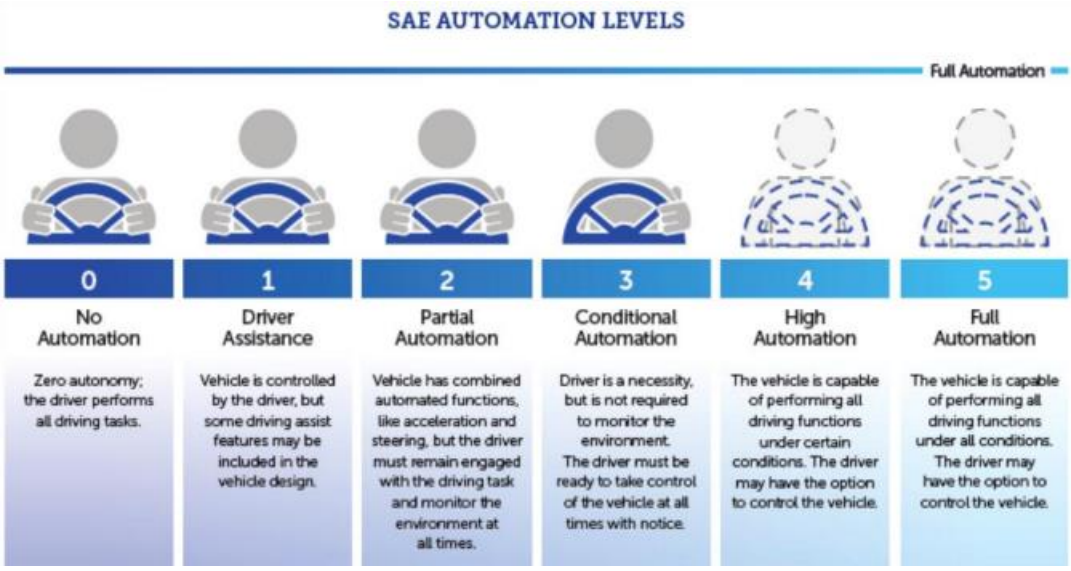
La autonomía nace de la forma de una capacidad de un objeto o un individuo, basados en la inteligencia que puede coordinar la acciones sin necesidad de una intervención directa de las personas. En el trasporte y uso de nuevas tecnologías este tipo de modalidades está directamente influenciado a una capacidad del automóvil para poder desplazarse sin un conductor.

La percepción y los reconocimientos ambientales rápidos, precisos y estables no solo son información esencial para que los vehículos presentados, los mismos completan las decisiones de planificación y el control del vehículo, sino también la garantía fundamental para la conducción segura sin supervisión. La detección de objetos, como tarea principal en la captación, percepción y reconocimiento de la conducción autónoma, ha atraído una gran atención y logrado un rápido desarrollo en los últimos años, mientras que todavía se enfrenta a grandes desafíos en escenas reales con el complejo acoplamiento del humano, el vehículo, la carretera y entorno (Ruan et al., 2023)

Cada desarrollo de la autonomía es posible en base a las diferentes inteligencias artificiales generadas por parte de: computadoras, sensores y actuadores que estén equipados dentro o fuera del objeto destinado a cumplir un recorrido. El entorno es primordial al momento de una conducción completa, es decir, la conducción autónoma debe estar preparada para tener una respuesta rápida y precisa ante altercados donde la propia vista del ser humano está delimitada hasta ciertos puntos, generalmente los usuarios al volante tienen un límite de visión, de igual forma los sensores de detección y mapeo, muestran un final del reconocimiento, al no poder notar obstáculos y objetos presentes a la vista.

Gran parte de la colisiones y actos imprudentes de los conductores esta influenciada en los factores ambientales, muchos de los casos son las condiciones ambientales (neblina y lugares con poca iluminación) en la que el conductor está expuesto. Este tipo de factores implica una falta de detección para los sensores creando una dispersión de datos y mala toma de los objetos captados por este tipo de sensor. Generalmente cada autonomía de conducción considerada por la junta de ingenieros en el año de 2014 define una clasificación para la autonomía en cada uno de sus cinco niveles. Empezando por el nivel 0, donde la conducción está destinada al usuario conductor , el mismo realizara cada una de las acciones acompañado de asistencias leves , sin tomar en cuenta asistencias de control lateral o longitudinal , el nivel 1 y 2 toman en consideración las asistencias longitudinal y gran parte de los movimientos laterales, sin embargo, el conductor tiene la mayor responsabilidad al saber que el vehículo no está listo para una detección de obstáculos imprevisto , seguidamente los niveles 3 y 4 denota un sistema de conducción automatizada ,sin la necesidad que el usuario conductor ejecute alguna demanda o intervención manual hacia el vehículo ,es decir , solo en casos de una situación de peligro el sistema ejecutara acciones correctivas en la trayectoria y finalmente el nivel 5, comprende una totalidad de las acciones en las que el vehículo puede prescindir de un conductor, en conjunto de asistencias técnicas y una variabilidad de estrategias guiadas para la mejora de conducción , este tipo de niveles lo denotamos en la figura número 1. Continuando con las asistencias vehiculares, en nuestra actualidad, no existe un entorno dedicado esencialmente a la conducción autónoma y asistencias de manejo, de esta manera se crearon soluciones comerciales alternas para obtener un reconocimiento del entorno en tiempo real, con ayuda de sensores de mapeo y recolección de datos.

Figura:1 :Niveles de ciclo de conducción



Fuente: (Marte Collado, 2018)

La detección de objeto cercanos o alejados, son centrados en la recolección de datos y la priorización de la comunicación de los espacios donde el vehículo puede movilizarse al determinar un rango de viaje estable, está basado en el uso de algoritmos de aprendizaje continuos, "los métodos de aprendizaje en profundidad utilizados para el procesamiento de fusión de datos, imágenes y nubes de puntos, se clasifican en las investigaciones relevantes según el nivel de dificulta y desafíos de la investigación académica para la aplicación práctica en los espacios abiertos"(Ruan et al., 2023), cada parte del aprendizaje está destinado a la detección y unión con imágenes generadas por el mapeo de objetos detectados .

El mapeo y detección son la unión de las bases tanto de: vehículos, peatones y señalizaciones en la carretera, generalmente al manejar es normal que a la detección de luz y falta de visión distorsionen el ángulo de captura de las distancias, de esta manera se genera datos erróneos en la percepción de datos de los sensores. En este caso se aplica la detección LIDAR, sea cercano o lejano la detección es notable, sin embargo, la presión es un poco inestable al momento de un mapeo completo de ruta, para estos casos se toma en consideración la

recolección y transferencia a una base de datos, dando una comunicación y mejor conducción con resultados de reconocimiento de objetos. De esta manera cada implemento utilizado para la comunicación de los objetos crea una retroalimentación para puntos similares o de distancias iguales, es así como la detección de objetos se reduce en gran parte el mapeo y generación de puntos de advertencia, haciendo uso de datos similares de la base de recopilación, destinada a una placa madre o de comunicación entre interfaces.

El sistema aplicado para el fácil reconocimiento de áreas no visuales en automóviles es de vital ayuda, en momentos donde no se tenga ningún tipo de visión. “Aunque la percepción colaborativa puede mejorar efectivamente la capacidad de percepción de los automóviles conectados, detectar objetivos ocluidos con mayor precisión y reducir la posibilidad de accidentes, la falta de un protocolo de intercambio de información eficiente en la percepción colaborativa conducirá a que la misma información se envíe entre vehículos repetidamente.”(Ruan et al., 2023). Este sistema es susceptible a la falla de datos erróneos por falta de visión, al tratar de generar un mapeo completo en tiempo real y poder generar recomendaciones de conducción en lapsos de tiempo cortos.

Al hacer uso de este tipo de localización de objetos, reconocemos factores económicos muy altos, además del costo de mantenimiento que requiere este tipo de mapeo completo no es muy accesible, al existir diferentes formas para una mejor calidad sin el uso excesivo de dispositivos de alta gama. Muchas tecnologías usadas en los vehículos actuales son sistemas de radares que detectan y pueden medir una velocidad y distancia entre el objeto y el punto de origen. “El radar es un sensor común en el vehículo con cierto grado de capacidad de penetración, lo que permite que el vehículo detecte e identifique obstáculos. En comparación con otros sensores de vehículos, el radar puede capturar la información de velocidad directamente en un amplio rango y se ve menos afectado por el clima”(Roriz et al., 2014). Para estimar una mejor adaptación y reconocimiento se establece controles adaptativos y se toma en consideración los diferentes sistemas de radares utilizados frecuentemente en vehículos. Como base de detecciones tenemos un radar de punto ciego, el mismo manda una señal o alerta para poder obtener una ruta alterna en caso de emergencia.

La gran parte de esta recolección de datos tiene una limitación en la capacidad de penetración de información, en base a los radares actuales se usa principalmente como una ayuda secundaria para los automóviles y su capacidad de obtener información de distancia y la

velocidad de otros objetos. En concreto, el área de detección de objetos se limita a un rango dinámico de entre los 6 metros a 8 metros acorde con la distancia del auto, su velocidad y la aceleración obtenida por un radar, genera un confort más estable en un ciclo de conducción, basado en alternativas secundarias de vías y toma de decisiones. Por otra parte, el rastreo de automóviles está destinado principalmente en el monitoreo de su localización y ubicación de cada movimiento efectuado en su transcurso. Hoy en día, este tipo de sistemas es usado tanto en la Autonomía de los vehículos, rastreo y monitoreo de flotas de transporte.

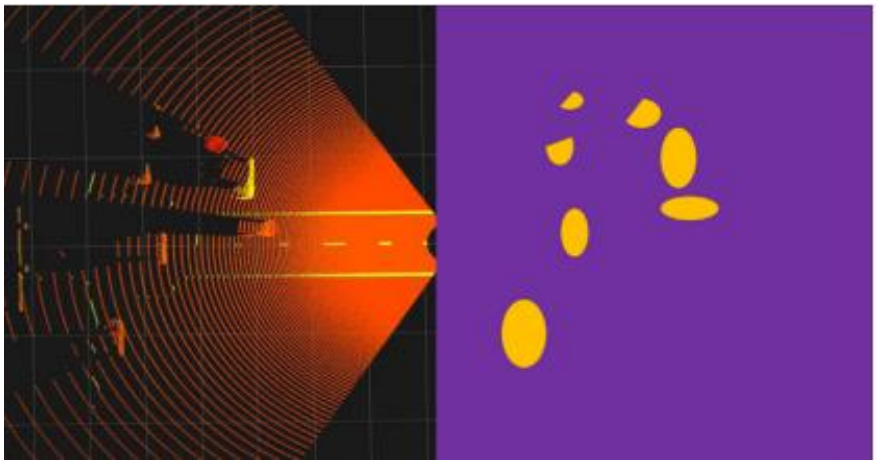
El acompañante de este sistema está dirigido al GPS, utilizado por medio de datos emitidos y reconocido por satélites, estos darán un tiempo real de la ubicación, sin embargo, tiene el defecto de no obtener el trayecto por cual fue recorrido. Es así que la distancia recorrida de los vehículos será obtenida, pero las vías utilizadas no se verán reflejadas para un seguimiento constante, para evitar este tipo de problemas se aplica tecnologías de seguridad y mayor autonomía. El radar y la autonomía de los autos está destinado como una opción viable , al poder ejecutar reacciones en caso de peligro evidente para la conducción, de esta manera se evidencia que el peligró en carreteras destinado por parte de autos delanteros o enfrente del usuario será más estable, para esto se propuso “un método de detección basado en la detección infrarroja para mejorar la detección del vehículo delantero, mientras que el vehículo siempre mantiene la distancia y la velocidad adecuada para evitar accidentes.” (Ruan et al., 2023).

La distancia en cuanto a los vehículos es un factor de área, al saber que la detección de obstáculos está limitado a un margen entre el punto de origen y la distancia recorrida por el láser al objetivo, factores a tomar en cuenta en este rango son velocidad y aceleración obtenida por el sensor, seguidamente se complementa con una trayectoria con el sistema de rastreo de los vehículos, sin embargo, si la medida de los vehículos captados es demasiado grande, la trayectoria es errónea y toma tiempo generar una trayectoria viable para el sistema dirigido en la parte autónoma .

Por lo tanto, se propone una alternativa de un umbral con una tasa de reconocimiento alta en base a su estudio, para poder verificar si el objeto está obstruido o no mediante el uso del sensor RPLIDAR A1M8-R5 para la detección de objetivos. De manera similar, la nube de puntos obtenida por LIDAR muestran objetos ocluidos. Tomando los datos LIDAR como la entrada de origen, cada uno de los datos de la nube se convertirán en un mapa de trayectoria,

mediante el uso de codificación estadística, que puede observar fácilmente la posición de los objetos en la escena (Ruan et al., 2023). Como se muestra en la figura 2

Figura 2 :*Detección de objetos, con sensor laser*

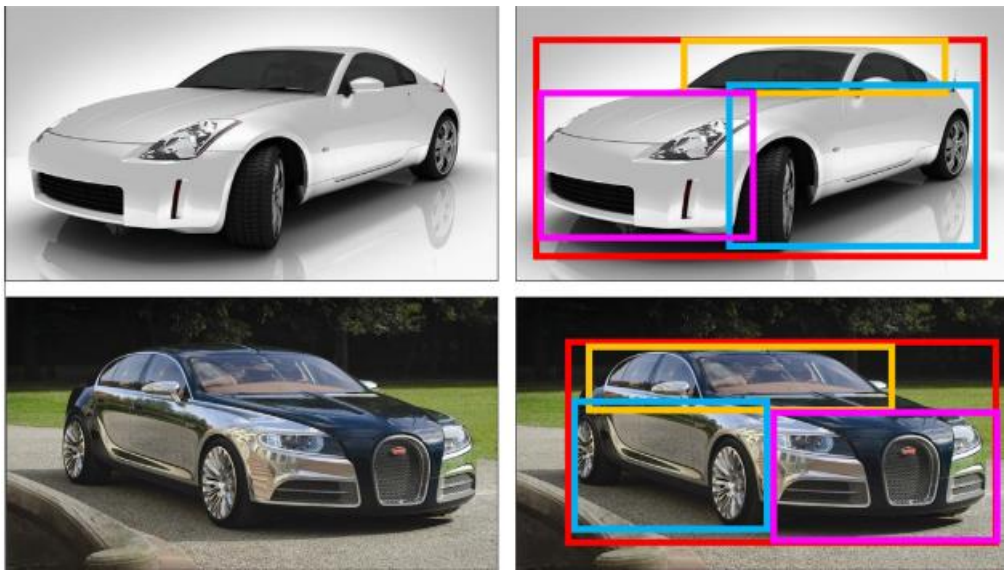


Fuente: (Avella Neira, 2011)

La detección de los objetos es primordial, de esta manera se fomentó que el mapeo y no tendrá datos faltantes, lo cual es factible para la identificación de puntos ciegos destinada en los objetos censados, para corregir este tipo de inconvenientes se planteó un método de división de áreas y partes principales que se pueden detectar, el objetivo completo se basa en codificar las piezas según el campo de afinidad del componente antes de volver a combinarlas en un marco de identificación para mejorar la precisión del reconocimiento de vehículos ocultos en la vista. Específicamente, dividen el objeto detectado, es decir, el vehículo, en cuatro partes semánticas, la parte superior, lateral, delantera y trasera del vehículo, está identificado en la Figura 3. Al hablar tanto del lado izquierdo y derecho del auto, son consideradas iguales lo cual reduce el mapeo completo, así, se plantea la identificación de la parte frontal y trasera. Este tipo de método puede evitar el mapeo de parte innecesarias para la captación de datos, ya que puede generar un error en la precisión de distancias o generalmente las imágenes mapeadas pueden estar distorsionadas. Las malas captaciones de las imágenes del sensor LIDAR crean datos erróneos, estos datos pueden ser reutilizados aplicando el método de mapeo de perspectiva inversa, donde se puede mejorar la precisión de objetos, ya sea con poca coloración e iluminación o de una estatura mínima del objeto censado.

Para evitar una oclusión se generó diferentes formas para reducir el impacto en los objetos censados. La alternativa secuencial para este tipo de errores está destinada a la instalación de micrófonos dentro del punto de partida (sensor), es decir, donde estará el sensor emitiendo el láser, el ruido producido mejorará la captación de objetos y además “soluciona el ruido ambiental, mientras que la última depende en gran medida de conductores experimentados (Ruan et al., 2023).

Figura 3 :Mapeo general del vehículo



Detección de puntos ciegos del automóvil, Fuente: (Millán Valbuena, 2020)

Continuando con la parte de segmentación de imágenes, esta implementación ayuda en gran parte a mapear zonas bloqueadas de la visión. En este caso se aplicará una mejora efectiva para la capacidad de detección de objetos ocluidos en lugares donde la visión emitida con el láser no pueda detectarlos, esto permitirá la ayuda de la recolección de datos y la translación a la interfaz correspondiente, la misma generara ajustes de los parámetros, además de planificar la segmentación y un aprendizaje autónomo que facilitará la distinción de un plano principal y uno secundario de los objetos alrededor del punto fijo.

CAPÍTULO I

1. Avances tecnológicos en la conducción autónoma

En nuestra actualidad un vehículo autónomo dispone de una variedad de tecnologías para la conducción sin la necesidad de un usuario humano, al tomar este rol y percibir el entorno que lo rodea. Al hablar de esta autonomía se conoce de los niveles de conducción existentes hasta la actualidad, estos conceptos están basados en la reconocida norma "SAE (Society of Automotive Engineers), organización enfocada en la ingeniería aeroespacial, automoción y las industrias comerciales especializadas en la construcción del vehículo"(Gómez, 2022). Este tipo de modelo mencionado define las lógicas y clasificaciones de la conducción autónoma además estandariza la comunicación y acciones que se tomarán en la navegación general.

Para el inicio de un vehículo autónomo, se caracteriza las 3 principales partes y funciones para la aplicación de este tipo de sistemas, como primer punto tenemos la percepción, esta tecnología trata de incluir nuevos modelos que puedan percibir cualquier tipo de objeto o hacer visual cualquier entorno alrededor o cerca de él, seguidamente tenemos la planificación, la misma se encarga del proceso de toda la información general y cálculo de una ruta alterna más óptima y acordé a la trayectoria fijada y finalmente tenemos al control, el cual es el encargado de la movilización y traslado del vehículo tanto en la parte longitudinal como en la parte transversal.

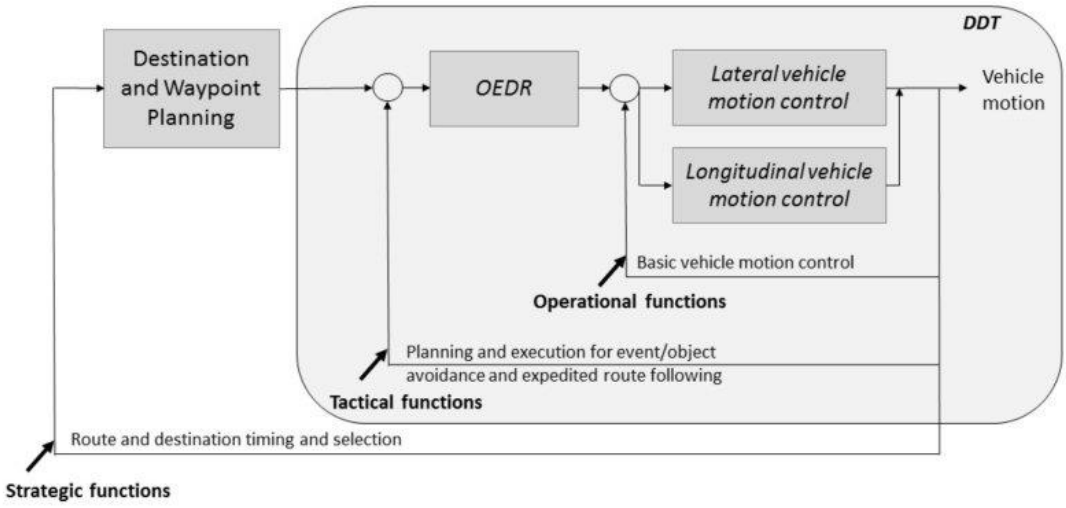
Uno de los conceptos esenciales para este tipo de conducción autónoma está basado en la conducción dinámica, también conocida como DTT (Dynamic driving task), consiste generalmente en las múltiples acciones y situaciones por los que pasa la conducción en nuestra actualidad, es decir, abarca diferentes habilidades necesarias para la conducción, tomando en cuenta la toma de decisiones de manera autónoma al percibir condiciones de las carreteras inestables, además de considerar las señales, tráfico, peatones y principalmente vehículos en torno a nuestra conducción. Por otra parte, el DTT toma en consideración las mencionadas características de la conducción autónoma: control, planificación y control. Este tipo de consideraciones están representadas en la figura 4.

- Monitoreo del entorno: los conductores usuarios al manejo, deben estar considerando el entorno en el que se dirige o una trayectoria fijada, al observar constantemente

cada vehículo alrededor de él mismo, además de las condiciones de tráfico y flujo de los vehículos, esto dará como resultado una anticipación a situaciones de peligro y poder tomar decisiones con mayor facilidad.

- Monitoreo de control: el usuario conductor estará visualizando advertencias mecánicas del vehículo, recalcando la parte de instrumentos y salpicadero, en conjunto de reconocer los niveles de combustible y la verificación adecuada que deba tener en el reloj marcador, finamente la posición de las luces asegura que el vehículo esté en óptimas condiciones para realizar una trayectoria segura.
- Monitorio de señales: Este apartado considera de manera puntual las señales de tráfico y letreros a lo largo de toda la trayectoria, ya que esto incluye los límites de velocidad y diferentes indicaciones en cada trayectoria como son: dirección, seguridad y control vehicular.

Figura 4: Control DTT



Fuente: (Marte Collado, 2018)

La característica principal de un diseño autónomo está basada en la parte tecnológica y también en la parte de añadir el sistema de DTT, considerando que este tipo de sistema es el encargado de una detección de objetos y eventos a su alrededor, el mismo genera una localización y sondeo del vehículo, ya sea una carretera o en lugares que no se reconozcan con la vista general. La percepción está ligada principalmente a la detección de vehículos, gracias a los datos suministrados en tiempo real se detalla los diferentes tipos de tecnologías

más comunes y aplicados como punto base de detección y sucesivamente hasta la actualidad la cual genera mapeos en tres dimensiones y generación de rutas, entre las cuales están:

Sensor Ultrasónico: Este tipo de sensores trabaja mediante la emisión y detección de ultrasonidos, el cual detecta la presencia, la distancia y el movimiento de los objetos. Su base de recolección está destinada en sensores que emiten pulsos de sonido ultrasónico, los cuales miden el tiempo que tarda en recibir las señales emitidas mediante el eco que rebota en los objetos y de esta manera localiza el objeto a distancia. Al hablar de las ondas ultrasónicas que detectan el objeto, estos parten de una energía suministrada y lo emiten sin dirección fija, simplemente detectan objetos cercanos a él, la ventaja de este tipo de sonidos emitidos es la velocidad a la que puede ser programado, oficialmente el rango estándar para la emisión de estos sonidos está entre los 20 kHz y 100 kHz de esta manera los sensores ultrasónicos detectan el objeto cuando parte de la energía es reflejada sobre el mismo y la cantidad de energía reflejada varía en torno a su salida programada, además el tiempo en el que tarda en reflejarse es directamente proporcional a la distancia en la que estará el objeto, por consecuente el receptor captará cada una de estas ondas y las convertirá en señales eléctricas, dichas señales pueden ser procesadas en una programación previa y a futuro generar un mapa estable con las diferentes dimensiones.

RADAR (Radio Detection and Ranging sensor): La aplicación de este sistema está dirigida a la emisión de ondas de radio para una detección y localización de objetos medianos y grandes, ya que su funcionamiento es secundario y sólo emite pulsos de radiofrecuencia, dichos pulsos de ondas están determinados en una frecuencia y una sola dirección, es decir, simplemente generan oscilaciones y cambios de señales para poder detectar los objetos y de esta manera amplificarlos antes de ser enviados de vuelta al punto de emisión, este tipo de pulsos que regresan al punto de origen son captadas mediante una antena parabólica o una antena de ranura, también conocida como matriz de antena.

La programación de este tipo de ondas de sonido es destinada a través del aire, agua o espacios abiertos y la velocidad a la que están siendo emitidas y redirigidas son más rápidas que la velocidad de la luz, de esta manera el receptor (RADAR) capta cada uno de los ecos reflejados y los convierte en señales eléctricas y este tipo de información podrán ser utilizados para el análisis de distancia velocidad y dirección. Generalmente son usados como sistemas de navegación aérea y marítima, en la parte automotriz está destinada

secuencialmente al control de tráfico y velocidad de movimiento de los vehículos, al ofrecer ventajas significativas en condiciones climáticas altas (lluvia, calor y humedad).

Cámara de detección: El objetivo principal de las cámaras es trabajar en conjunto con el software de inteligencia, de esta manera puede detectar y reconocer con mayor facilidad cada uno de los objetos e incluso clasificarlos al determinar su medida estándar, además se puede registrar la visión y los diferentes cambios al momento de ser captados por la cámara, comúnmente son utilizados en los campos de seguridad y vigilancia aérea. Por otro lado, también son utilizados en métodos para la detección de movimiento como la luz y detección de calor a diferentes niveles o análisis de imágenes en constante movimiento, los mismos hacen uso de luces infrarrojas y en la parte de detección de calor se aplica sensores PIR , son encargados de detectar una variación de radiación por medio de infrarrojos y los datos obtenidos por estos sensores son procesados por las señales de retroalimentación, al determinar un movimiento en el área cubierta por el sensor, el sensor detecta un cambio y activa una señal de salida consecutivamente enviando una variación de datos en tiempo real.

Sensor LIDAR: Este sensor es una de las tecnologías qué retoma el uso de los pulsos láser mejorado, al poder medir distancias y crear mapas tridimensionales en todo su entorno captado, al mencionar este sistema cada uno de los mencionados anteriormente son una recolección y conjunto para recalcar a este sistema LIDAR, al tener el funcionamiento de enviar con mayor agilidad los pulsos láser y medir el tiempo en que tardan en regresar al sensor con el efecto rebote, determina el escaneo de un área completa usando un espejo giratorio, su testeo es más rápido, al emitir el láser en los 360° cada objeto será determinado mediante su dirección y ángulo.

El avance de la tecnología y detección de objetos a distancias es un factor muy esencial para el escaneo del entorno, generalmente fijo y además proporciona un sondea entre los objetos detectados sea toma en cuenta lugares como: topografía o lugares con demasiados desniveles. Este dispositivo es basado en la emisión de luces o pulsos de luz laser, percibidos a distancias en tiempo real, en base a datos recopilados son generadores de mapas tridimensionales de gran precisión como: 2D y 3D. "Las técnicas de medición BASADAS EN LA LUZ se utilizaron por primera vez para calcular la densidad del aire en la atmósfera superior simplemente leyendo la intensidad del reflejo del haz de un reflector."(Roriz, n.d.).

El funcionamiento de esta tecnología está dedicada a la detección de objetos, dependido de su rango o capacidad de alcance muchos de estos sensores son capaces de ser más o menos eficientes, el punto de partida este tipo de sensores está destinado en los radares implementados en las décadas de 1940 donde investigadores y parte de la fuerza área militar aplicaron este método de pulsos de luz destellantes en lugar de las ondas de radios convencionales utilizadas en la trasmisión de señales de comunicación.

"Los sensores LIDAR dan información absoluta e inmediata sobre la distancia a un punto, mientras que los sistemas basados en cámaras necesitan procesar e interpretar las imágenes antes de obtener un resultado fiable" (Gómez, 2022). De igual manera que las cámaras los sensores LIDAR tienen una dificultad al momento de captar objetos de poca densidad, esto corresponde a una falla de recolección de datos, Por otro lado, los sensores LIDAR tiene una ventaja favorable sobre los radares, al demostrar una precisión en la distancia que se encuentra el objeto, comúnmente los radares son más aplicados en este ámbito al tener un acceso más fácil en la parte de su precio, es así que los sensores LIDAR no son aplicados con gran frecuencia, ya que tienen un precio elevado en la actualidad.

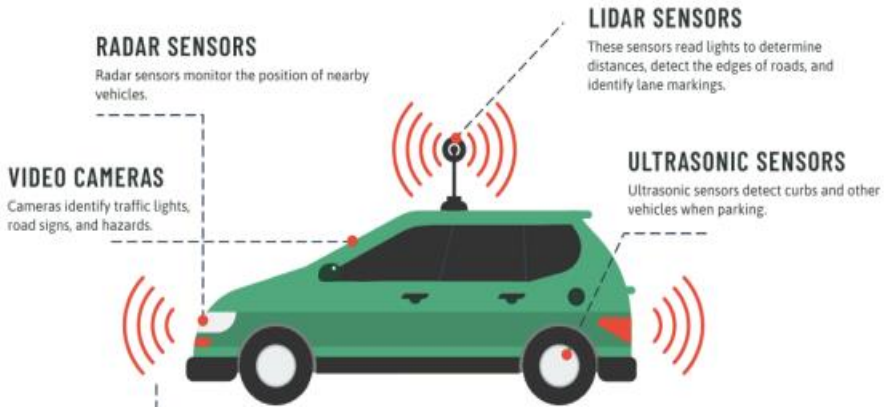
Cada uno de los sensores aplicados para la **detección de objetos** ya sea, al usar un láser, ultrasonido, etc. Reconocen objetos de diferentes distancias y en muchos casos generan un mapeo la distancia en tiempo real y esta manera evidenciamos cada una de sus características en común que tiene cada uno de los sensores, como lo son:

1. Distancias con precisión: Este tipo de sensores tiene una amplia gama de medios de objetos de largo alcance, cuenta con un rango máximo de 40 metros, y tiene incorporado una precisión notable, usualmente tiene designado con un rango de centímetros o milímetros, facilitando la función de un mayor detalle del lugar a su alrededor.
2. Conducción de información: "Se especifica la interfaz que se usa para comunicar los datos de la medición. Generalmente se aplica un bus serial y se especifican las velocidades de transmisión soportadas." (Avella, 2021)
3. Funcionamiento independiente: La emisión y represión de los datos reflejados no dependen de una luz ambiental, en comparación con las cámaras convencionales, este tipo de tecnología genera su propio laser y sonido, al percibirlo regresa con el

reflejo sondeado, ayudando al funcionamiento de testeo de lugares en condiciones bajas de visibilidad como: vías con neblina, túneles sin luz, etc.

Al mencionar cada uno de los sistemas aplicables para la detección de objetos, se entiende las diferentes formas en las que se puede determinar una distancia en un área determinada, cada uno de los sensores está especificado en la figura 5.

Figura 5 : Tecnologías de detección



Fuente: (Gomez, 2008)

Los sensores LIDAR ofrecen muchos beneficios para los vehículos autónomos, debido a su precisión en la detección. Su característica principal está destinada al uso de los datos obtenidos, los cuales son reflejados en puntos centrales en un plano cartesiano de esta manera detectan la distancia en la que se encuentra con mayor precisión. Asimismo, los datos generados pueden ser procesados por un sistema informático. Es muy notable visualizar uno de estos tipos de sensores en la parte automotriz en la actualidad, ya que se trata de buscar una autonomía para el conductor, evitando diferentes accidentes al momento de conducir y realizar una maniobra inesperada(Alcaraz, 2021).

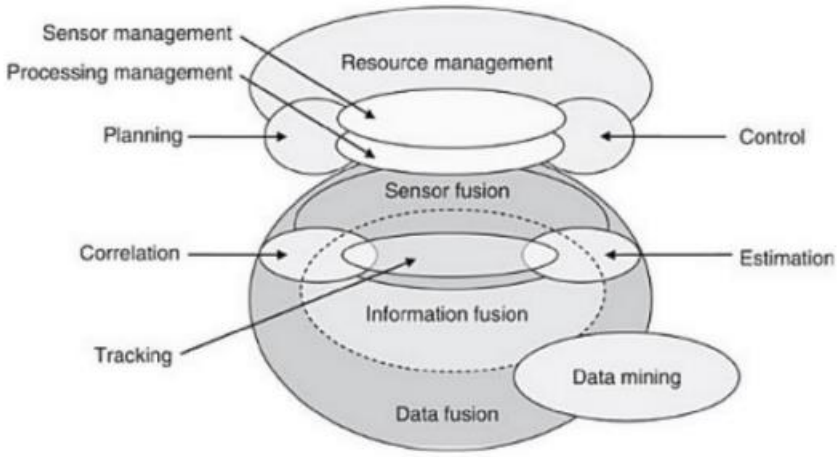
1.2 Fusión de datos

Al recibir la información de cualquiera de estos sensores mencionados, sea en la parte de datos por ultrasonido, luz infrarroja o un láser emitido entre 360°, cada uno de los datos recolectados y enviados a su punto base deben ser unidos de manera general, existen diferentes técnicas para la unión de datos y gestión de información, donde su principal

objetivo es conseguir los resultados prácticos para cada uno de los datos almacenados sea en alta o baja calidad, se busca generar un flujo de datos y acciones para cualquier sistema físico o mecánico al que esté destinado.

En la parte de los vehículos autónomos al tener gran cantidad de sensores, se obtiene una cantidad excesiva de datos de información, para esto se crean técnicas de fusión de sensores y gestión de información, de manera terminológica está representado en la figura 6. Anteriormente el costo computacional para este tipo de gestión de datos era excesivamente costoso, al no contar con técnicas muy elevadas para la captación de estos tipos de datos, en la actualidad se aplica la recolección mediante el uso de placas electrónicas y software de procesamiento. De esta manera se puede recolectar cada uno de los datos de manera estandarizada al ser uso de una comunicación IDE. Este tipo de programas conocido como un software de desarrollo integral, es utilizado principalmente para las placas Arduino y sus diferentes versiones para la generación de interfaces gráficas, el programa tiene una facilidad en su parte escrita, al permitir escribir y poder editar los diferentes códigos fuente. Al momento de generar un código sea extenso o con un número elevado de datos, el programa cuenta con una amplia gama de bibliotecas que podrán corregir y discernir el elevado número de datos, de esta manera es como su generación de programación tiene una accesibilidad para facilitar la lectura y escritura de códigos.

Figura 6: Fusión de datos



Fuente: (Alcaraz, Seguimiento lidar, 2016)

La codificación es la parte importante de este tipo de programas, al determinar que el programa sea extenso o pequeño, el mismo será trasladado a una compilación de las diferentes placas mediante un USB. Para este tipo de información trasladada se aplica un método serial qué proporcionará el monitoreo de cada una de estas partes codificadas, además de visualizar y enviar los datos desde la base madre hacia el receptor, este tipo de información y gestión son conocidos como IDE.

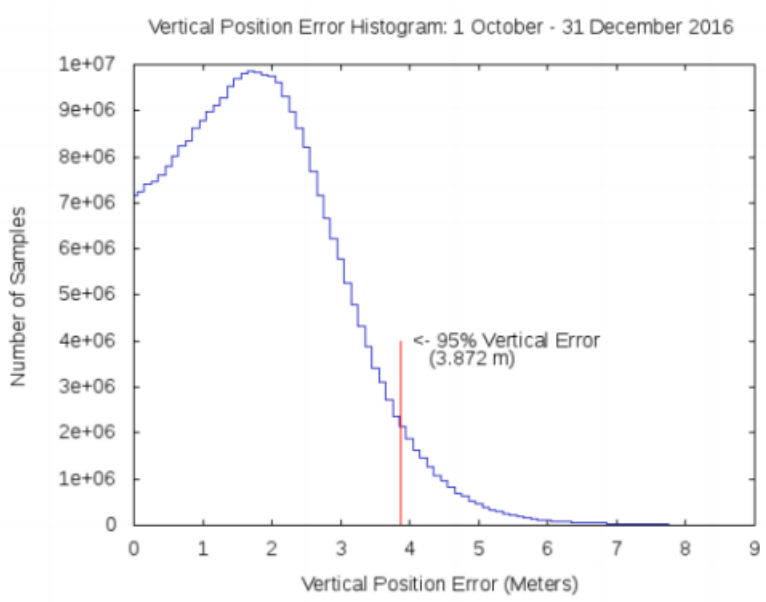
Además para este tipo de comunicación qué constan con un flujo amplio de datos se trabaja en conjunto con el IDE y se aplica la interfaz de comunicación Arduino IDE la cual brinda un fácil acceso y programación para gente principiante, donde proyectos sea electrónicos de manera inalámbrica o física (a través de cables), es tomado muy en cuenta en la actualidad al tener varias alternativas de programación y gestión de datos, sin embargo existen alternativas más avanzadas donde se aplicará una mayor programación, plataformas como Plataformero y Visual Studio Cod son las herramientas más populares para el software y la programación generalmente este tipo de programas, también hacen uso del IDE, enfocándose en el desarrollo de programas con microprocesadores y extensiones físicas como: el Raspberry, Arduino, entre otros. Hablando de Visual Studio Code, podemos decir que es una alternativa un poco más viable, ya que su funcionalidad es más rápido y su recolección de datos en tiempo real con sus variables extensiones ayuda a la velocidad y tasa de refresco con un alta calidad, por último hablamos de su gran extensión y admisión de códigos en diferentes lenguajes de programación, podemos catalogarlo como una recopilación del lenguaje madre, ya que admite lenguajes populares como : Python , Java , C++, entre otros. Este tipo de admisión aligera en gran cantidad el traslado de codificaciones y programación de un lugar a otro sin necesidad de cambiar escritos y verificar la sintaxis al momento de ejecutar el programa.

1.3 Algoritmo de mapeo

Para la conducción autónoma, es necesario tomar en cuenta que el vehículo debe estar establecido con un sistema GPS de esta manera sabrá la posición exacta en dónde se encuentra. Este sistema es el más conocido y aplicado hasta la actualidad, al realizar tareas de posiciones globales y punto de ubicación, sin embargo, en la mayoría de los casos la ubicación no es 100% real, ya que al estar en constante movimiento su tasa de flujo de datos varía constantemente, es así que se genera un error de ubicación en casos extremos de entre

80 metros a 100 metros, a continuación se muestra un ejemplo de una posición destinada por el GPS del año 2016, dónde en el mismo se muestra un error de posición vertical de 3.872 metros de su punto original 1.872 metros de distancia.

Figura 7 : Posición del GPS



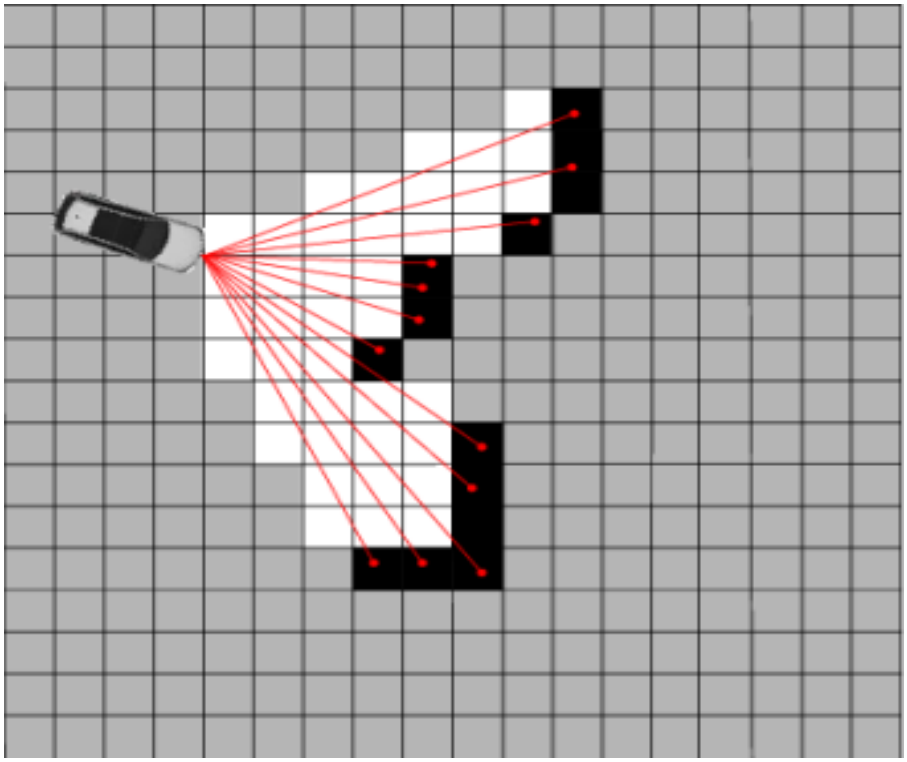
Fuente: (Gomez, 2008)

"La familia de algoritmos basados en OGM (Occupancy Grid Mapping), fueron introducidos por Hans Moravec y Albert Elfes en la década de los 80"(Alcaraz, 2021). Este tipo de representación es usado generalmente en proyectos robóticos y eléctricos, dónde se basan en el uso de un sistema de navegación autónoma, ya que el mismo tiene la función de un mapa, en el cual divide su entorno en celdas o cuadrículas y cada espacio que detecte un objeto o un movimiento se ocupara una cuadrícula respectiva. Los algoritmos basados en el OGM están destinados a la representación de gráficas, con la finalidad de generar tareas cómo planificación de rutas, evasión de objetos u obstáculos, localización en tiempo real y generación de mapeos múltiples.

La combinación de los diferentes sensores, como el LIDAR, cámaras o sensores de detección, sensores de ultrasonido, etc. Generan una ubicación en la cuadrícula mencionada en el OGM, Al integrar la información de los sensores y determinar una ocupación o no, en la celda, cada una de estas celdas tiene una posición y generalmente puede determinar el tamaño del objeto u obstáculo detectado. De esta manera puede generar una ruta más segura

y eficiente, en el caso de la autonomía se puede aplicar para un proyecto o una parte física implementada para el uso de automóviles, es usada generalmente para espacios donde no se reconozca el entorno o se tenga poca visibilidad. “Este tipo de técnicas parten de la suposición de conocimiento de la posición del vehículo, por lo que es necesaria la aplicación de métodos de posicionamiento con antelación”(Alcaraz, 2021). Para tener un mejor resultado en el reconocimiento de la posición, este depende frecuentemente del tamaño definido de cada celda, es decir, cuanto más pequeño se defina, mayor será la precisión que se tenga del objeto y por el contrario entre más grande sea la cuadrícula la ubicación tenderá a presentar errores en la distancia. Para entender este tipo de metodología marcada por la OGM se presenta la figura 8, dónde se puede determinar un claro ejemplo de un mapeo ejecutado en conjunto de un sensor LIDAR y la generación de mapeo con cuadrículas.

Figura 8 :Método de cuadriculas



Fuente: (Alcaraz, 2021)

CAPÍTULO II

2 Descripción y uso componentes aplicados

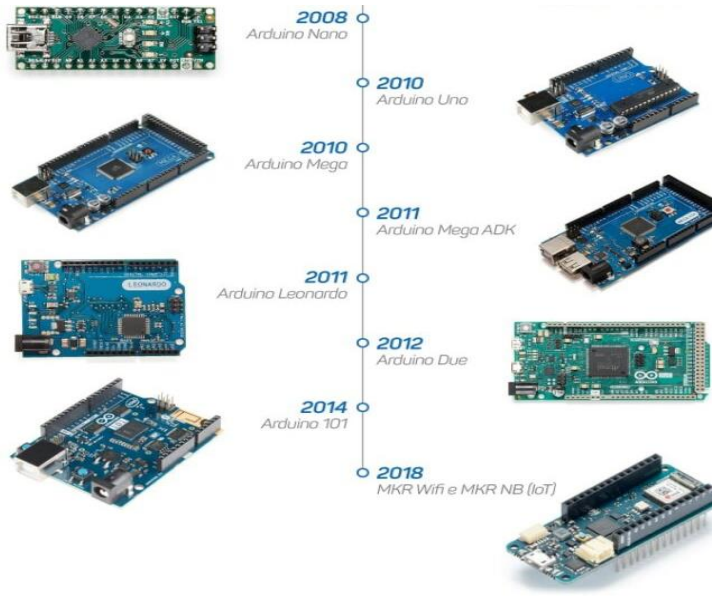
2.1 Definición de Arduino

Arduino es considerado como un prototipo electrónico, tiene como base una placa de hardware y además consta con una amplia plataforma de desarrollo tanto de hardware como de software de código abierto, que tiene como fin la reducción de costos al momento de realizar trabajos de programación convencionales y profesionales, mediante la venta asequible de placas electrónicas basadas en un microcontrolador y el acceso libre a sus diseños. “Estas placas contienen todos los elementos electrónicos necesarios, además de un entorno de programación con las herramientas y las bibliotecas esenciales listas para iniciar a desarrollar sin tener que empezar desde cero”(Avella, 2021)

Es conocido por tener una interfaz muy accesible y multiuso, con el propósito de crear circuitos simples y extensos, al ser una placa integrada consta de microcontroladores con entradas y salidas programables, este tipo de placa permite una programación y conexión en sensores, actuadores, y demás componentes electrónicos. “Arduino puede determinar el entorno mediante la recepción de entradas desde una variedad de sensores, además de implementar microcontroladores dentro de la placa que serán programados usando el Arduino Programming Language donde la autonomía se puede comunicar con el software en ejecución dentro de un ordenador”(Enríquez Herrador, 2009). Muchos de los proyectos autónomos son basados en esta placa, al considerarlo muy versátil en su manejo y programación electrónica, en este caso indicamos los tres más reconocidos. Primero, contamos con una programación de seguidor de línea, esta programación explica como un robot es guiado por una línea trazada en el suelo y mediate una luz infrarroja, el Arduino percibe la información de los sensores y puede controlar el movimiento autónomo de la línea marcada. El segundo, es un monitoreo de energía, al crear un dispositivo capaz de dar datos en tiempo real del consumo de los electrodomésticos, utilizando sensores de corriente que brindan un testeo cada cierto periodo programado desde la placa procesador y finalmente, aplicado en la industria automotriz están los vehículos autónomos, conocidos principalmente por no hacer uso de conductores y tener un sistema autoguiado, haciendo uso de: cámaras, radares y sistemas de navegación, toman una ruta alternar mediate el análisis de toda la

información brindada por los diferentes componentes, de esa manera la respuesta de guiar el auto está basada en tiempo real.

Figura 9 : Tipos de Arduino

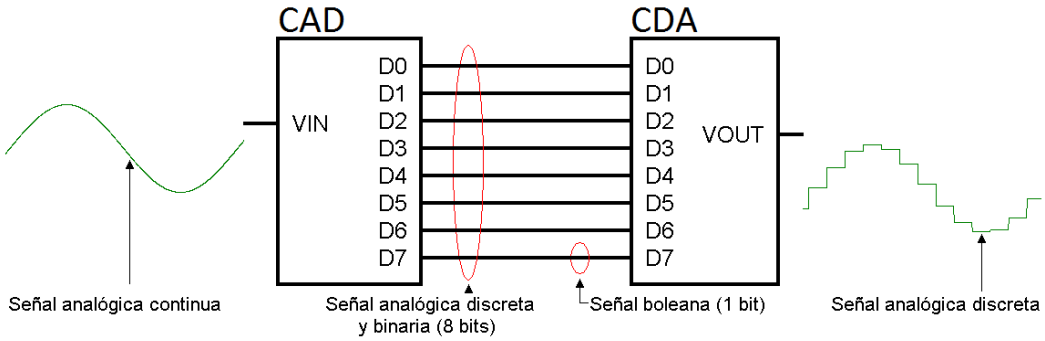


Fuente: (Marques, 2018)

Este tipo de placas ofrece una variedad de programación y creación de bases de proyectos e interacciones con microcontroladores, también conocido como "Un sistema microcontrolador mono placa, de hardware libre, de fácil uso y bajo coste, desarrollado inicialmente para facilitar el uso de electrónica en diseños artísticos e interactivos y la aplicación de esta por personas no expertas"(Herranz, 2015). Al contar con diversas entradas y salidas de comunicación, digitales y analógicos. Por una parte, las señales digitales provienen de la comunión estándar de 1 y 0, esto son ejecutados mediate un botón, pulsador o interruptor. Por otro lado, la señal analógica que proviene de datos más continuos, estos casos presentes son una estandarización de un acceso donde "Se puede tomar información del entorno a través de sensores conectados a sus entradas analógicas y digitales, puede controlar luces, motores y otros actuadores directamente o partir de las señales de control generadas en sus salidas. Hay modelos de Arduino específicos para desarrollos más fáciles al llevar tecnología (weareables), en e-textiles. Además, se puede

comunicar con otras placas Arduino o y con otros sistemas, mediante Wifi, Ethernet, Bluetooth, etc., esto permite también la interacción a distancia y el Internet"(Herranz, 2015)

Figura 10: Señal analógica y digital



Fuente: (Llamas, 2016)

2.2 Comunicación Arduino

La capacidad de datos intercambiables en la placa Arduino y los diferentes componentes programados en un código, son el resultado de una serie de comandos a cumplir por parte de los componentes físicos adaptados a dicha placa. "Hoy en día la manera más común de comunicación entre dispositivos electrónicos es la comunicación serial y Arduino no es la excepción. A través de este tipo de comunicación podremos enviar datos y desde nuestro Arduino a otros microcontroladores o a un computador corriendo alguna plataforma de medios (Processing, PD, Flash, Director, VVVV, etc.)"(Ruiz Guti, 2007). Arduino muestra una amplia gama de metodologías para una correcta comunicación que nos permiten establecer una interfaz entre la parte programada y la parte física o mecánica, en gran parte es conocido como la principal característica de poder interactuar con el entorno y las diferentes interfaces propuestas hasta la actualidad.

Una de las variaciones y recomendación para la integración y comunicaciones con la placa Arduino está dada por los lenguajes de programación más utilizados como:

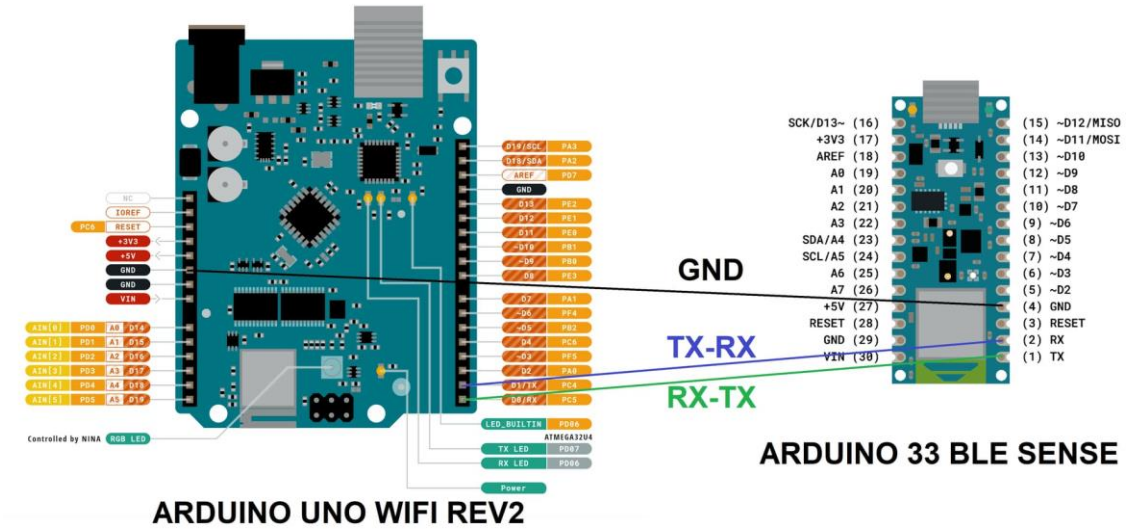
- Comunicación serial: También conocida como la comunicación y transferencia de datos, basado en la comunicación por bits es decir, la información sea captada o recopilada por un sensor o sistema digital está destinada a recibir un dato en código binario, generalmente está representado en la lectura de 0 y 1. Además este tipo de

comunicación hace un uso de los protocolos específicos de una transmisión de datos seguros y precisos, las aplicaciones más comunes para la comunicación serial son: RS-232 Y RS485. Este tipo de estándares de comunicación son aplicados en placas de Arduino Uno, Arduino Mega Y Arduino Nano, generalmente son usadas para la comunicación externa o dispositivos basados en señales wifi o bluetooth, al transmitir la información y sincronizar el funcionamiento correcto entre el emisor y receptor.

- Comunicación I2C: Este tipo te protocolo de comunicación o extensión serial, genera una interfaz entre múltiples conexiones y dispositivos. Sea uno o dos dispositivos, los mismos pueden tener un mismo lenguaje de comunicación y programación donde ambos pueden tener una misma línea de datos o recepción de datos, es decir, la comunicación usa una línea de reloj donde se pueden sincronizar todos los dispositivos: el maestro y el esclavo, en este caso el maestro es aquel base que genera un inicio y un control de comunicación, mientras que el esclavo es el encargado de responder las peticiones o solicitudes generadas por el maestro. Este tipo de comunicación es utilizado en aplicaciones como el uso de microprocesadores, sensores y dispositivos de visualización o captación del entorno mediate un sondeo del área y la recepción de los datos obtenidos.

- Comunicación Inalámbrica: Este tipo de placa además de una comunicación física, permite una conexión de datos y transferencia de señales inalámbrica haciendo uso de módulos de transferencia como: Bluetooth, Wi-Fi, entre otros. Por una parte, el módulo basado en bluetooth permite una comunicación de corto alcance entre los dispositivos Arduino y otros sistemas, como computadoras o celulares inteligentes mediate el uso de los pines RX y TX mismo que usan una comunicación serial, este tipo de pines específicos son usados para recibir y enviar datos, están destinados a los pines 1 y 2 en la placa Arduino, esta representación está especificada en la figura 11. Además de ser una entrada y salida de datos, los mismos pines pueden ser usados como conectores USB, que permitirán una comunicación con la computadora generando una agilidad en la trasferencia de datos.

Figura 11 :Señal de comunicación



Fuente: (Marques, 2018)

2.3 Programación inicial Arduino

La programación de este tipo de placas se basa en un lenguaje C++, proporcionando una amplia gama de bibliotecas de interacción y funciones especiales para un desarrollo más exacto, en base a proyectos la misma placa tiene "una estructura básica que se compone de una función setup, donde se configuran los pines del microcontrolador (entrada/salida), empezando la inicialización con las variables globales, entre otros. También, se compone de una función loop donde se ubica el código principal que ejecutará el microcontrolador y, como su nombre lo indica, es una función que se ejecuta continuamente"(Avella, 2021)

La función mencionada, está dirigida a una parte esencial del programa, al ejecutarse más de 3 veces y consecutivamente crea un bucle infinito para determinar tanto la programación como la ejecución de acciones. Adicionalmente, la placa Arduino tiene una vinculación con la biblioteca IDE, las mismas se ejecutan para una extensa variedad de comandos de programación para cada pin o comunicación serial.

El entorno de desarrollo integrado o también conocido como IDE está basado en una aplicación para la programación encargada de codificación en las placas de Arduino, esto con el conjunto de herramientas y una facilidad en desarrollo de proyectos. El IDE cuenta con características como:

- “La compilación de código es tan fácil que incluso una persona común sin conocimientos técnicos previos puede mejorar con el proceso de aprendizaje” (Ferazi & Dahoud, n.d.).
- Una edición de código que se puede escribir y modificar con el lenguaje conocido como C/C. Además de contar con funciones especiales de autocompletado para una facilidad de escritura del código.
- Su variación de IDE está en contenidos de: Edición y Compilación, por un lado, está la edición, esta es la responsable del ingreso de comandos de programas que dirigen a microprocesadores. Por otra parte, la compilación es el apartado donde la información que realizamos es entregada y cargada a la tarjeta Arduino.

2.4 Definición y uso de componentes

La selección y uso de los componentes está basada en la calidad y diferentes formas de presentación del mismo componente. En este caso se toma en consideración solo aquellos componentes que tienen más de dos tipos de presentación para hacer su elección en el proyecto.

2.4.1 Arduino Mega

Este tipo de placa electrónica es conocida por su microcontrolador ATmega2560, también establecido como controlador de 8 bits, misma que cuenta con una velocidad de transferencia de datos o velocidad reloj encargada de ejecutar varias instrucciones preprogramadas y ejecutadas. La misma velocidad se la mide en Hertz (Hz) y se entiende por la capacidad de ciclos completados o ejecutados. Este tipo de controlador está fabricado para un total de 16 Hertz, es decir, que su rendimiento será basado en 16 millones de ciclos reloj por segundo, dando como resultado un microcontrolador vasto de poder que realizara operaciones hasta de 16 millones de ejecuciones o instrucciones por segundo, cada una de las ejecuciones serán realizadas sin ningún problema mediate una correcta configuración del sistema. La placa tiene un alto consumo de energía en cada transcurso y ejecución de instrucciones, por lo cual, se aplica una reducción en la velocidad de reloj y por consiguiente de las ejecuciones para obtener un ahorro de energía o mejoras en la estabilidad del sistema.

La compatibilidad con el entorno de IDE del Arduino facilita la programación al ser una aplicación o extensión de herramientas y bibliotecas funcionales listas para un desarrollo de software, la característica principal esta destina a un depurador y editor de programación que facilita la carga de comandos en la placa , al constar de un código abierto, es decir, que la licencia otorga una libre modificación y entrega una gran cantidad de bibliotecas disponibles, dando resultado de una manipulación abierta en la creación de proyectos y extensión programación en el control de múltiples sensores y actuadores.

2.4.2 Características

Este tipo de placa es más conocido por su gran cantidad de número de pines, considerando la entra y salida de datos recopilados, además de contar con una serie de características que la resaltan de las placas comunes de programación, como lo son:

- Pines de entrada y salida: Al ser denotado como una placa de gran capacidad tiene un total de 54 pies digitales, generalmente 15 pines principales son destinados a salidas con una modulación de ancho de pulso, estos determinan los pulsos o energía entregada, mediante la cual se puede variar dicha característica de la señal. Su mayor utilización y aplicación es en el control y variación de velocidad de los motores o servo motores que se pueda llegar a utilizar.

- Memoria: Siendo un tipo de placa extensa y de gran capacidad de uso de interfaces para la ejecución de comandos, la misma hace uso de una memoria de 256 kB de capacidad, de esta manera se genera una tasa de refresco de datos muy fluida, al contar con una memoria RAM de 8kB, al ser limitada se aplican recomendación de uso en la memoria y programación, evitando retraso en la captación y recolección de información.

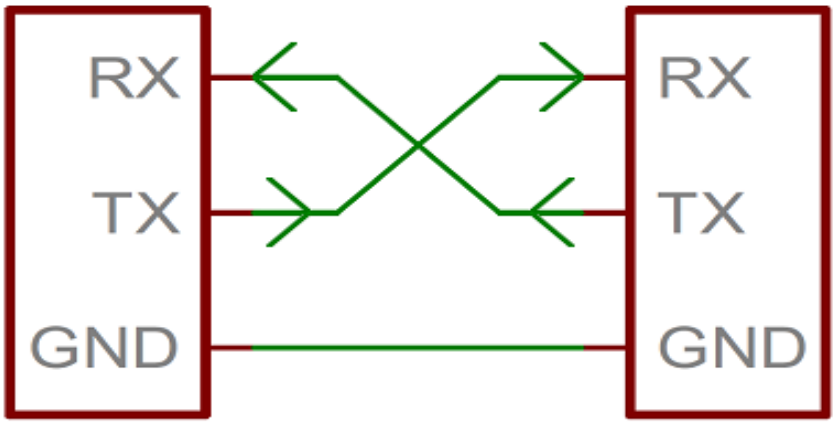
2. 5 Bluetooth HC-06

Estos dispositivos están destinados a la comunicación de marera inalámbrica, generalmente este tipo de módulos también cuenta con el implemento del Bluetooth la misma genera conexiones entre una gran variedad de dispositivos al mismo tiempo. Considerado en la programación como el esclavo, al tener que realizar los comandos o peticiones de los dispositivos conectados o denotados como maestros, para este tipo de dispositivos con base

a proyectos los más comunes en aplicarlos son: Teléfonos, Computadoras y Tabletas, cada uno de estos elementos puede generar una orden mediate su debida programación, la que será cubierta por el módulo mencionado para su correcta ejecución.

La comunicación serial usada en el dispositivo bluetooth es el serial UART, la que permite la transferencia de datos entre dispositivos y puede generar una mejora en la comunicación serie de un punto a punto. Al basarse en la transmisión y recepción de datos bits (0 y 1) por medio de un cable físico, se programa generalmente los puertos TX (transmisión o emisor) y RX (receptor), este tipo de puertos están identificados en la figura 12. La calidad de velocidad transmitida a cualquier tipo de información es muy considerable según el número de bits transmitidos, en este tipo de módulos capta la velocidad de transferencia, la más frecuente está oscilando entre los 9600 bps a 115200 bps (bits por segundo), esta medida cuantifica la velocidad real de transferencia de datos en la aplicación digital de programación lo que permite el uso comúnmente en proyectos electrónicos o que estén basados en la comunicación por redes o red inalámbrica.

Figura 12 :Pines de comunicación entre componentes



Fuente: (Brampton, 2022)

2.5.1 Características

Al considerarse que es un módulo de carácter inalámbrico, sus funciones y aplicaciones varían dependiendo de su correcta programación, además al tener una calidad considerada eficaz en la comunicación de datos, se considera que es por sus múltiples características de las cuales se resaltan las siguientes:

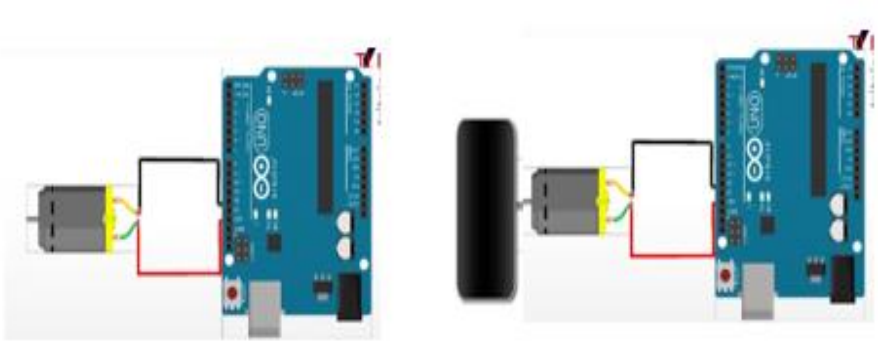
- Alcance de comunicación: Al ser un módulo de tipo HC-06 tiene un rango de alcance nominal, de entre los 6 a 10 metros en condiciones normales, sin embargo, bajo condiciones donde se use más de 5 dispositivos a la vez su taza de rango se reducirá para poder compensar la transferencia de información, evitando retrasos innecesarios en la toma de datos y recopilación.
- Modo esclavo: Este tipo de modo, está basado en una conexión y emparejamiento por dispositivos secundarios, todo tipo de dispositivos que haga uso de una red inalámbrica o Wi-Fi puede generar una interfaz directa con el mismo, además de este tipo de conexión, el bluetooth es la continuación de una alta velocidad de transferencia de datos al oscilar entre los 2.0 y 2.1 Mbps. Ambos sistemas aplicados en cada traspaso de información y recepción de operaciones a realizar o ejecutar son totalmente eficientes.
- Aplicación especifica: Para uso de proyectos de autonomía y control de múltiples funciones tiene una mayor comodidad, reducción energética y al tener una amplia gama de dispositivos que pueden ser vinculados sin gestionar un retraso remoto inalámbrico en la transferencia de datos.

2.6 Puente H L298N

Conocido también como un circuito integrado usado para el control de motores de corriente (DC), como su nombre lo indica puente H, este circuito es capaz de controlar y reducir corriente, de esta manera permite tanto el avance y giro de la velocidad con la que se desea trabajar. Al existir una variedad de modelos controladores de energía, el más frecuente al uso de motores estándar es el modelo L298. Su paso de corriente en direcciones opuestas permite un intercambio de información de ambas direcciones al mismo tiempo, emitiendo o recibiendo comandos de reducción y aumento de voltajes, esto da como resultado el giro consecutivo y sin retraso. Generalmente este tipo de circuito integrado cuenta con una capacidad de almacenaje de datos y trabaja con un máximo de amperios admitidos que oscilan entre los dos amperios por cada uno de los canales, este tipo de corrientes admitidas tienen un límite de flujo de corriente por canal siempre y cuando se trabaje con motores de tamaño mediano o pequeños, caso contrario se puede hacer uso de amperajes más elevados si se trabaja en conjunto de motores de más calidad o de mayor capacidad de almacenaje y paso de corriente.

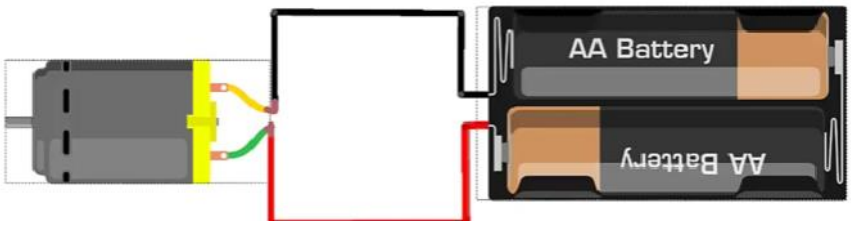
Este tipo de controladores es parte fundamental para el flujo y abastecimiento de corriente, al notar que la gran mayoría de las placas Arduino tiene un bajo flujo de corriente para controlar motores, generalmente este tipo de placas entrega un máximo de 50 mA (mili amperios), y la capacidad de desarrollo de este tipo de motores sería nula y los motores no generarían un giro constante sobre los motores además de poner en riesgo la integridad de la tarjeta, para abastecer de un flujo neto de corriente y evitar sobrecalentamiento en las placas Arduino se implementa de manera externa un puente H. La representación está destinada a la corriente que puede entregar la placa Arduino desde sus pines a los motores de manera directa, por otra parte, cuando los motores se encuentra en vacío y no tiene cargas sometidas no necesita mantenerse de manera constante incluso sin una placa Arduino y simplemente con una fuente de alimentación continua de manera externa (baterías, cargadores, entre otros), cuando el motor no es sometido a ninguna carga puede estar alimentado de manera independiente sin una programación siempre y cuando sea un motor de bajo consumo, representada en la figura 13 y 14 a continuación.

Figura 13: Pines de alimentación



Fuente: (DroneBot , 2015)

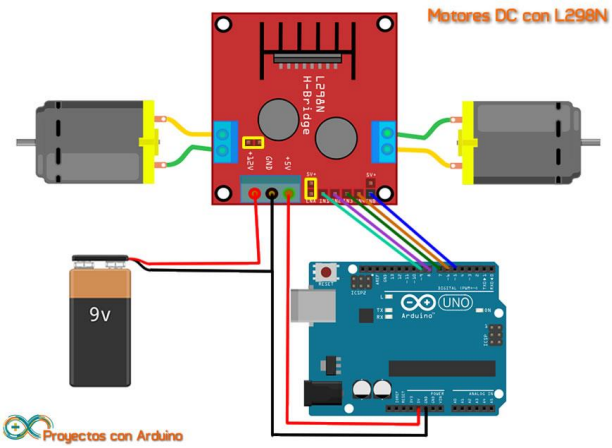
Figura:14 Alimentación externa



Fuente: (DroneBot , 2015)

Generalmente cuando se debe aplicar un giro o variación de corriente y el motor no se encuentra en carga vacía, se aplica una corriente más amplia, además de proceder a un ensamblado entre la placa Arduino, el puente H y una alimentación externa de corriente continua, la misma que se conecta a las salidas de cada motor a los puertos H con cada terminar correspondiente (positivo y negativo), se procede a la programación e indicación de los tiempos de pulsación que requiere en cada interruptor, mientras un puerto se desactiva los otros se activan para dar paso a un giro de motor en sentido horario y según corresponda a las necesidades que se le apliquen tendrá una secuencia de giro estable en base al ancho de pulso que se modula en los microcontroladores o el circuito integrado de la placa Arduino, en la figura 15, a continuación se indica un armado completo del circuito mencionado .

Figura 15 :Circuito base del sensor LIDAR



Fuente: (DroneBot , 2015)

2.6.1 Características

Su funcionamiento está destinado al giro y cambio de corrientes por medio de pulsaciones preprogramadas, este tipo de placas tienes una serie de característica que la destaca como esencial para el uso en trabajo de electrónica y automatización, los principales son:

- Voltaje de operación: Este tipo de placas consta de una variedad de voltajes para su alimentación, están oscilando entre los 5 V a los 35 V (Voltios), esto es una característica muy práctica para poder alimentar dicha placa, dependiendo de las necesidades y utilidad en la que se aplique dicha placa. Generalmente la operación del voltaje suministrado para la placa debe ser de igual o del mismo

rango equivalente que el de los motores a utilizar, ya que este tipo de puentes solo actúa como un paso o interruptor de corriente que deja fluir la misma corriente que se suministra.

- Configuración del puente: La rotación, variación de giros y escala de velocidades está basada en la polaridad de la corriente, la configuración más utilizada en proyectos de uso de motores está destinada al uso de mínimo cuatro controladores de flujo de electricidad, permitiendo un control total de giro. Cada terminal del puente está conectado a interruptores superiores o principales (M1 y M2), los mismos son colocados en serie y por consiguiente los puertos inferiores son colocados en paralelo (M3 y M4). Su funcionamiento se guía en cerrar los interruptores M1 y M4 mientras que los interruptores M2 y M3 se mantienen abiertos dando como resultado un paso de corriente y giro del motor en una dirección especifica.

2.7 Giroscopio Acelerómetro

Considerado un sensor acelerómetro o de tres ejes, son aplicados en el uso de dispositivos móviles o de traslación, este tipo de sensor detecta el movimiento y el trascurso tomado por su trayectoria realizada. Este sensor está destinado a la medición de velocidades o cambio de aceleración en sus tres diferentes direcciones o puntos de trayectoria trazados en su entorno, las tres direcciones más usadas en este tipo de sensores son los ejes X, Y y Z. Al poder detectar el movimiento en estos tres planos pueden servir como datos para una navegación autónoma de carretera o una estabilidad de cámaras en realidad virtual. En la actualidad son aplicadas en temas automotrices al dar una detección en la navegación del GPS.

En resumen, este tipo de sensores al contar con tres ejes centrados en las mediciones de velocidad angular y aceleraciones, están centrados en la aceleración lineal producida por el objeto o dispositivo que toma la velocidad y la varían en un perdido de tiempo. Utilizado para una recolección de datos, usando los principios de conservación de momentos angulares y movimientos de rotación, es decir, que permiten una detección más rápida y concreta del movimiento tridimensional.

2.7.1 Características

Este tipo de procesador tiene una diferencia entre la parte del giroscopio y el acelerómetro. A continuación, se presentan algunas características comunes de estos sensores

- Rango de medición: Dependiendo del modelo y su fabricación, este tipo de sensor tiene rango de medición en unidades de velocidad angular, en medición con una baja calidad y resolución pude estar en una escala nominal de los ±100 °/s (grados por segundo). Por otro lado, al constar de una alta resolución puede tener un rango de medición de ±1000 °/s o más.
- Sensibilidad a las vibraciones: Para hacer uso de la medición, se debe establecer un aparado fuera de las vibraciones bruscas o movimiento instables, al tomar en cuenta que estos modelos son muy susceptibles a la detección de vibraciones, esto afecta en gran parte a la recolección de datos en tiempo real, al saber que los datos pueden variar por una inestabilidad del sensor acoplado.
- Ejes de medición: El giroscopio mida la velocidad angular en los tres ejes, X, Y y Z, de esta manera puede proporcionar datos exactos para un mapeo o creación de rutas en 2D y 3D para un sistema autónomo libre.

2.8 Processing Versión 4.2

Processing es un tipo de lenguaje de programación generalmente orientado para la dedicación de los programas abiertos en la codificación, además de implementar una amplia gama de desarrollo dónde el lenguaje base que está dirigido a la programación en java, facilita la implementación en animaciones y visualizaciones. Este software de programación y entorno de desarrollo integrado de código abierto tratado en las bases de la Java es de fácil utilización, ya que sus creadores Benji y Casey Reas en el 2001, mientras eran estudiantes de posgrado en el instituto tecnológico de Massachusetts decidieron implementar un programa de fácil acceso y orientación para programadores con bases similares para la creación de códigos. El programa está pensado para programadores, artistas y diseñadores que quieran expresarse de manera directa y especial. Otras de las funciones de Processing es acercar las necesidades de los artistas a las nuevas tecnologías por lo tanto muchos artistas nuevos prefieren este lenguaje al ser de contexto simple y gratis para crear arte electrónico,

además que no es necesario tener una gran experiencia en el mundo de la programación para realizar los primeros escritos de programación electrónica.

Además de crear datos estadísticos y cálculos de datos en tiempo real, se puede aplicar en determinadas áreas, tal es el caso de procesamiento de imágenes y visualización, recepción y recopilación de datos en tiempo real para su transferencia de datos a la parte física o mecánica del sistema programado. Al hablar de processing también podemos entender sus diferentes sistemas y niveles, su parte esencial de procesamiento es su software, ya que de esta manera se puede aplicar en las diferentes plataformas de dispositivos electrónicos como: celulares, tabletas, computadoras (Windows o MacOS). En nuestra actualidad processing es aplicada en las plataformas Android e iOS por su facilidad de instalación y uso de programación desde este tipo de dispositivos, al tener una compatibilidad con dispositivos electrónicos (cámaras, micrófonos y sensores).

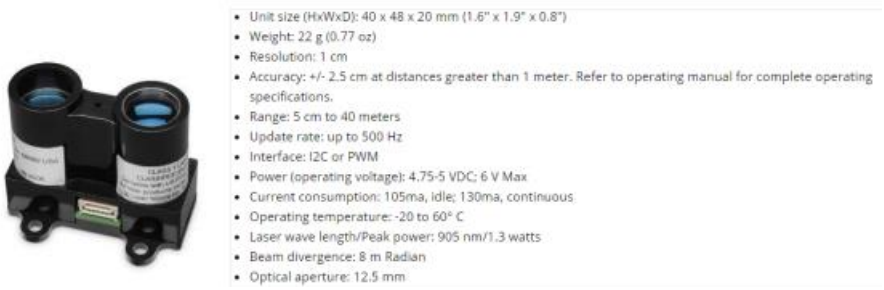
2.9 Definición y selección de los componentes principales

Este tipo de sensor se puede adquirir de diferentes tipos, al abarcar diferentes rangos o distancias que puede variar desde las aplicaciones en topografías y mapeos de áreas, en general puede cambiar el tipo de proyectos que estemos destinados a realizar. En la actualidad se busca la economía y la calidad de producción de datos, así se destacan los principales ejemplares como son:

2.9.1 LIDAR Lite V3

Muestra un alcance máximo de 40 metros, resolución de un centímetro, precisión de 2,5 centímetros para alcance inferiores, a 5 centímetros y 10 centímetros en alcance superiores. La unidad tiene una tasa de actualización de 270 Hz en modo típico y 650 Hz en modo rápido, lo que significa que la unidad puede proporcionar 270 mediciones en 1 segundo en modo típico y 650 mediciones en 1 segundo en modo rápido. El precio general de la unidad es de alrededor de $ 150 dólares.

Figura 16: LIDAR Lite V3



Fuente: (Fegemu, 2019)

Se utiliza para controlar el funcionamiento del LIDAR y la rotación paso a paso. También funciona como generador de mensajes para el ROS a través de una conexión en serie con la computadora que ejecuta ROS. Permite eliminar la complejidad del diseño mecánico necesitará un ensamblado físico especialmente el cable del LIDAR debido a su movimiento de exploración/rotación, se utiliza una conexión basada en Bluetooth entre el Arduino y la computadora(Widodo et al., 2018).

2.9.2 TF02-Pro

Este tipo de sensor está destinado al uso de rango medio estable en los campos abiertos, generalmente tienen una lectura en distancia de los 40 metros. Estos mismos pueden ser utilizados en la medición y control de vehículos aéreos. Este sensor mejora la interfaz en la capacidad óptica y un análisis de datos en ambientes exterior. “Los dos modelos de LIDAR elegidos para el diseño del sistema (TF02-Pro & TF03) presentan en su ficha técnica un buen nivel de precisión tal que no debería presentar inconvenientes de acuerdo al uso dado en este proyecto”(En et al., 2013). Este tipo de modelo es destacado por la escala de rango, es decir, varía en la intensidad de la señal como medio de comprobación de errores en la compilación de datos.

Figura 17 :LIDAR TF02-Pro

Fuente: (Marques, 2018)

2.9.3 TFmini Plus

Está considerado como un sensor de categoría de sensor a distancia, al contar con un rango de detección de entre los 0.1 a 12 metros. Además de contar con un flujo operativo de alimentación de 5 voltios y la energía suministrada para el funcionamiento de 110 amperios. Al ser un sensor compacto y peso promedio de 12 gramos, tiene un uso versátil para el uso en campos abiertos y fácil transportación.

El segundo modelo de LIDAR mostrado para el diseño del sistema (TF02-mini) presentan en su ficha técnica características prometedoras para el desarrollo de este tipo de programación. Sin embargo, para comprobar la veracidad de estos datos, se verificó las especificaciones técnicas de un modelo de LIDAR a escala perteneciente del mismo rango de productos (TF-Luna), para denotar cuál es el uso más exacto y estético dentro de un proyecto técnico y de programación (Ahmed, 2020).

Figura 18: LIDAR TFmini Plus

Fuente: (Marques, 2018)

2.9.4 RPLIDAR A1M8-R5

Este tipo de sensor está dispuesto al mapeo de 2D en 360 grados, al tener un giro total en su entorno muestra un mayor alcance de visibilidad, con un rango máximo de lectura de objetos de 6 metros. Cada dato mapeado es redirigido para la modelación de objetos en el entorno. Su estructura es diseñada para la medición en triángulo laser, donde refleja un láser que rebota por los objetos y es captado por el lente de recopilación de un máximo de 5.5 Hertz. El mecanismo electrónico está basado en un láser y hardware que mide datos de distancia en más de 2500 veces por segundo y una salida de resolución alta.

Figura 19 :RPLIDAR A1M8-R5

Measuring Standard

- Distance Range: 0.15 - 12 m (with white object to be reflected in the test)
- Angular Field of View: 0~360°
- Resolution: 0.5-1.0% (minimum 0.5 mm) @ < 1.5 m
- Angular Resolution: Less than or equal to 1° (when scanning at 5.5Hz)
- Sampling frequency: 4000 - 8000 Hz
- Scan frequency: 1~10Hz, typical value 5.5Hz (when RPLIDAR takes 360 points per scan, scan frequency is the typical value)

Optical property

- Laser wave length: 775~795nm, typical value 785nm (infrared band)
- Pulse width: 110 us

Laser instantaneous peak power:

Fuente: (Fegemu, 2019)

En este caso hemos aplicado cada una de las especificaciones del fabricante y hemos usado el sensor **RPLIDAR A1M8-R5** por tener un uso muy amplio en sus aplicaciones, al notar que puede tener un mapeo competo y aplicación en los vehículos autónomos, al tener una vista de 360 grados contando con ocho puntos de medición por cada rotación es factible y versátil al momento de la detección de objetos específicos. Por último, se lo denoto por su patrón de creación de puntos en 2D médiate su emisión de pulsos laser que podía medir la distancia después de reconocer el objeto.

2.10 Arduino

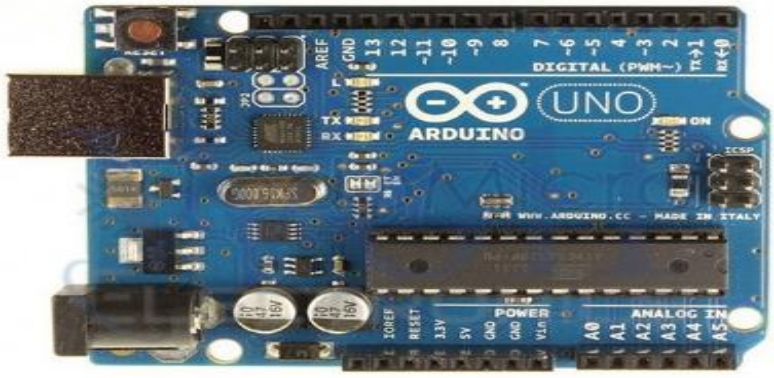
Al ser una plataforma de códigos abiertos, el mismo es utilizado para el control de partes electrónicas, al constar de un hardware y plataforma con amplias bibliotecas, se crearon diferentes modelos y variaciones de placas Arduino. Cada una de las diferentes formas está

basada en la disposición y la diferente utilidad que se le aplica. A continuación, se mostrará la clasificación a tomar en consideración para la implantación del proyecto.

2.10.1 Arduino UNO

Arduino Uno: Es conocida como la placa de microprocesadores más conocida y aplicado en proyectos de electrónica, al tener una facilidad y versatilidad al momento de su programación. Al ser una placa de código abierto y tener la capacidad de poder modificar y realizar diferentes códigos de comunicación es una de las mejores plataformas de software. Uno de sus componentes más usados en este tipo de placa son los controladores ATmega328P al ser un microcontrolador, ofrece una eficaz ejecución de instrucciones y una gran capacidad de bibliotecas donde puede optimizar el consumo de energía aplicado para cada acción a realizar en su tiempo de trabajo. Como se representa en la figura 20, a continuación, además podemos observar su total en pines de entrada y salida digital, y su incorporación de una interfaz USB que permite una conexión directa a la computadora.

Figura 20 :Arduino uno



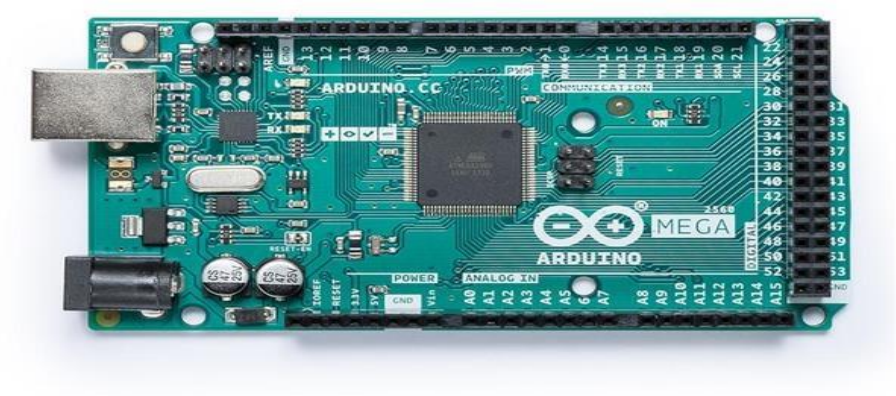
Fuente: (Marques, 2018)

2.10.2 Arduino Mega

Placa con un microcontrolador basado en una versión más amplia y eficaz para el desarrollo de programación y diseño de proyectos, al tener un mayor número de pines, contando con un número de 54 pines. De los cuales 16 de los pines son dirigidos a entradas analógicas, es decir, que permite el acceso y conexiones de sensores, actuadores, entre otros dispositivos. De igual manera cuenta con una memoria de almacenamiento de 256 kB, esta memoria está dispuesta a la transferencia más rápida de información. Al contar con un número mayor de

puertos es mucho más versátil en sus conexiones, en parte de la programación cuenta con una mayor extensión de bibliotecas para el diseño de programas. Como se muestra en la figura 21, a continuación, se observa que su tamaño es el doble a comparación con su versión anterior.

Figura 21: Arduino mega

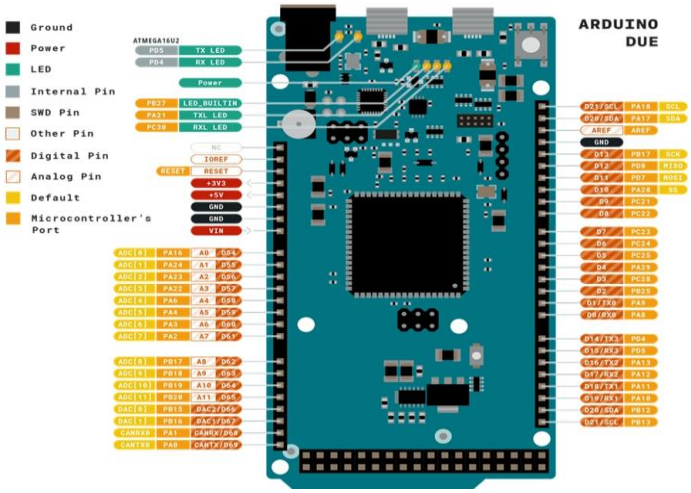


Fuente: (Orenda, 2016)

2.10.3 Arduino DUE

La placa Arduino Due cuenta con un controlador (SAM3X8E) microcontrolador que consta de una fuente de almacenamiento de información de alrededor de 512kB y además cuenta con una extensión de 4kB de almacenamiento para datos no volátiles, es decir, datos que no puedan ser almacenados o retenidos por la placa, una vez interrumpida la energía que lo alimenta para ejercer su funcionamiento. Al contar con 32 bits nos permite tener una respuesta y ejecución más rápida para las instrucciones. Por otro lado, doce de sus pines están destinados principalmente para la precisión y lectura de sensores analógicos, generado un espacio solo para sensores y actuadores. A continuación, se representa en la figura 22, el tamaño estándar de la placa y parte de los pines mencionados.

Figura 22: Arduino Due



Fuente: (Orenda, 2016)

En este apartado y una vez explicado las placas más comunes y usadas para los proyectos de electrónica, hacemos uso del **Arduino Mega,** por su gran número de pines disponibles para las conexiones físicas y digitales, además de contar con una amplia gama de bibliotecas destinadas para una programación más estable y rápida.

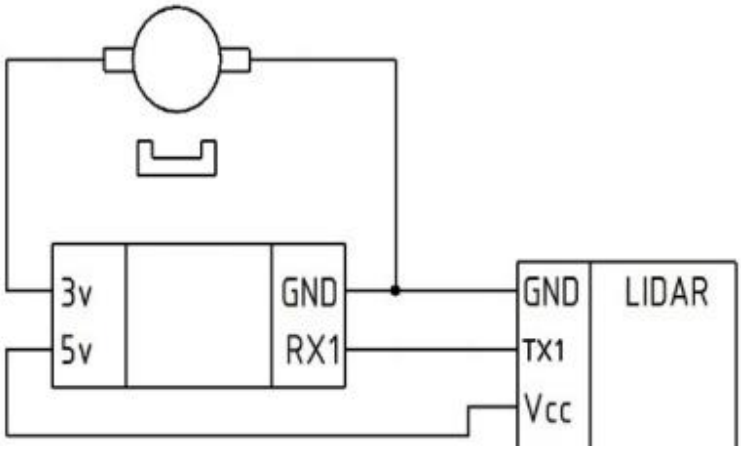
Capitulo III

3 PROGRAMACIÓN

3.1 Soporte técnico del sistema

Para el soporte del sistema, se tomó en cuenta la implementación del Arduino mega debido a que cumple estándares de funcionamiento y requerimiento para este tipo de mapeo. El diseño fue basado en el haz de luces reflejado por el sensor, donde obtenemos los datos de distancia de los objetos que lo rodean. Para empezar, el eje de este sistema de conexión principal está representada por la unión e instalación del Arduino mega en conjunto con sus pines GND, RX1 además de su respectiva entrega de voltajes 3 y 5 voltios representados en la figura 23 a continuación.

Figura 23: Sistema base sensor LIDAR



Fuente: (Denysyuk, 2018)

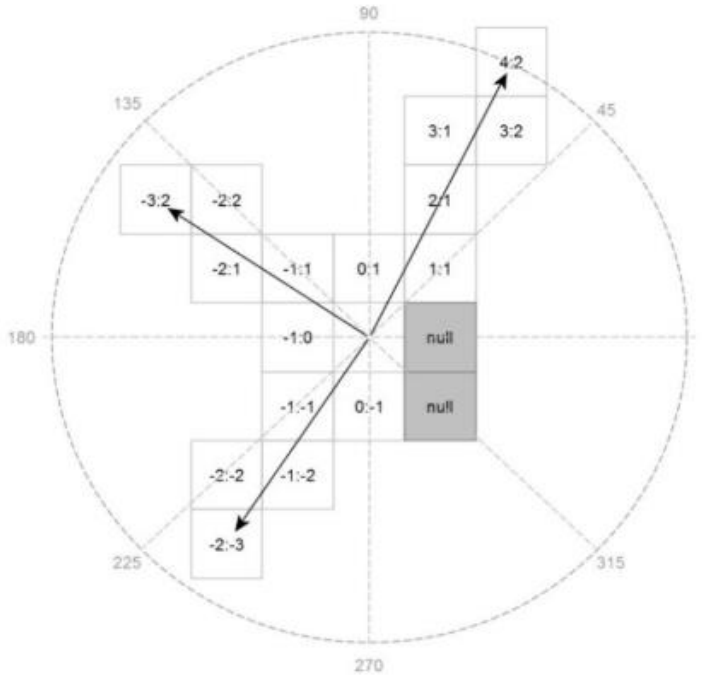
El pin RX1 es parte del Arduino mega el cual establece la comunicación serial mencionado con anterioridad, donde la misma está basada en la recepción de datos transmitidos por el sensor LIDAR. Este tipo de datos es recopilado gracias al pin de salida TX1, que al estar conectado con la placa además de emitir señal, están conectadas en conjunto con la alimentación para poder ejercer el ciclo de trabajo previamente programado.

Para el programa de control se toma en cuenta el uso del microcontrolador de módulo Bluetooth HC-06. Este tipo de sensor es suficiente para el entorno de un cuarto o habitación y cuál oscila entre una medida y rango de 11 m² a 15 m², la medida de detección del rango puede ser más amplia haciendo uso de un microcontrolador más avanzado. Seguidamente se

utilizó el giroscopio acelerómetro de 3 ejes para la medición del ángulo en el que gira, para el inicio se lo activó y se tomó en cuenta su funcionamiento, antes de utilizarlo se debe medir la distancia y corregirlo mediante la implementación de un sensor de reconocimiento, este tipo de orientación y determinación de posición está destinado al giroscopio el sensor LIDAR y la placa Arduino, al saber que cada uno de estos datos conforman la adquisición en tiempo real de los objetos, además se creó el desarrollo de un programa para intercambiar los datos y poder tener una comunicación entre el gestor de adquisición de datos y el programa ejecutor de movimiento.

En la figura 24 a continuación se muestra un ejemplo de la construcción de datos mediante el testeo, esta gráfica está representada en un plano cartesiano determinando la distancia desde el punto base del sensor, al generar puntos por el cual pasan la trayectoria y hasta recibirlas en la placa para su correspondiente trazo de trayectoria. La trayectoria realizada o testeada utiliza como base el acompañamiento de una cuadrícula creada en el plano cartesiano.

Figura 24: Puntos captados en el plano cartesiano



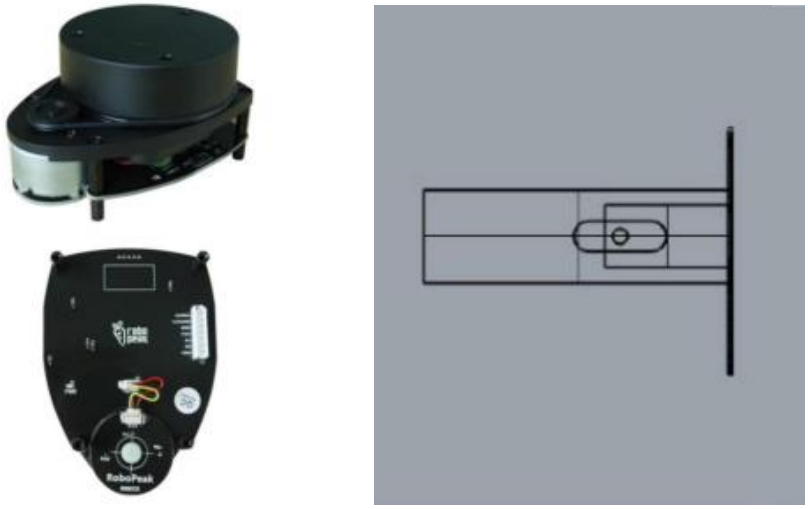
Fuente: (Denysyuk, 2018)

3.2 Construcción

3.2.1 Mecánica del vehículo

El diseño del vehículo en su mayoría sólo se trata del armado y ensamble de las partes prefabricadas qué se mostrarán en el siguiente apartado. Sin embargo es necesario recalcar la parte importante para el giro y mecanismo del sensor LIDAR, partiendo de este punto el servomotor integrado en el motor LIDAR proporciona un giro y la estructura del mismo es aprovechada, para generar un mecanismo de soporte fija sobre la lámina prefabricada del chasis, este tipo de lámina fue estructurado con la finalidad de aprovechar el espacio correspondiente para cada uno de los componentes (ruedas, motores, puente y placa Arduino), la figura 25 a continuación muestra las dimensiones de la parte superior e inferior del sensor LIDAR que nos permitirá saber y aprovechar la estructura realizada.

Figura 25 :Sensor LIDAR y estructura



Fuente: (Denysyuk, 2018)

3.2.2 Electrónica y mecánica del vehículo

El ensamblado y construcción de este prototipo de vehículo está basado en piezas y componentes previamente fabricados, de igual manera el giro generado por el sensor LIDAR es un acople electrónico prefabricado y diseñado para un ensamble y montaje sobre el eje posterior de vehículo. El primer paso consiste en el armado de la parte inferior, es decir, parte del chasis, cuatro ruedas, motores DC, discos separadores que son colocados en los

cuatro ejes y finalmente se arma un montaje cerca de los respectivos motores para tener un control de la cantidad de deslazamiento del vehículo. Tomando en cuenta que el chasis del vehículo soportará cuatro motores se generó un espacio correspondiente al lado de cada rueda, para poder fijar con mayor seguridad cada uno de los motores del ensamblado.

A continuación, la mecánica principal para el ensamblado del vehículo autónomo fue destinado para el acopio de piezas prefabricadas como se lo mencionó anteriormente, además se procedió a dejar espacios para el acople y ensamblado tanto de las partes electrónicas, cómo las fuentes de alimentación tal es el caso de: puente h, motores, sensor LIDAR, acople del sensor Bluetooth y cableado externo que va dirigido a la placa Arduino

Figura26 :Estructura lateral

Fuente: Autores

Figura 27 :Parte superior

Fuente: Autores

Seguidamente, el segundo paso de este ensamblado está dirigido al acople del sensor LIDAR sobre la parte superior del vehículo, se estableció un giro de 360° para el sensor pueda girar libremente sin ninguna interrupción, en el mismo se colocó alzas y una base para sostenerlo

fijamente en el automóvil. Tomando en cuenta y se recalca que la parte que sostiene el sensor LIDAR es un acople fabricado en plástico, el cual unirá tanto la parte del chasis directamente al movimiento del sensor LIDAR. A continuación, se muestra una figura del acople y ensamblado sobre el chasis.

Figura 28 :Base del sensor LIDAR

Fuente: Autores

Para finalizar, el último paso está representado en la figura 29, a continuación, el que consiste en la fijación de su fuente de alimentación para poder generar los movimientos independientes una vez programado y almacenados los datos con el sensor LIDAR, para poder corregir los movimientos al momento de preparar su trayectoria.

Figura 29 :Montaje de la base sobre la estructura

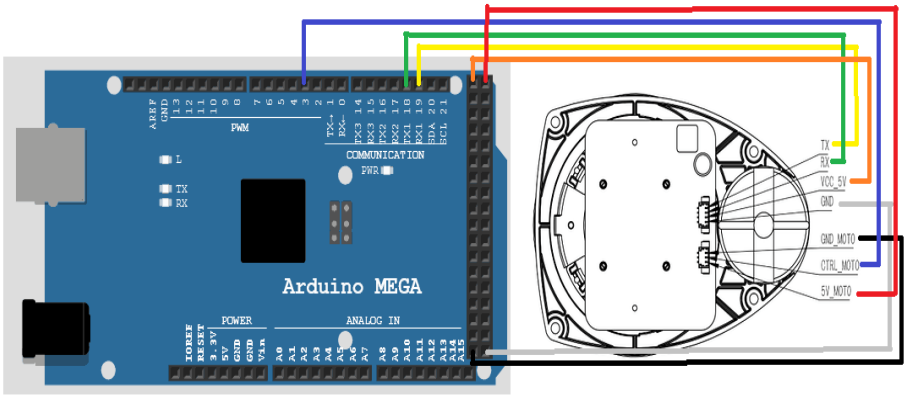
Fuente: Autores

3.3 Conexiones físicas de los módulos a utilizar

La programación se encuentra estrictamente ligada al hardware utilizado, ya que muchas veces al utilizar un módulo es necesario disponer de las librerías necesarias para que este funcione correctamente, además para comprender el código es necesario tener claro los puertos que están conectados en la placa Arduino.

- La alimentación no supone ningún problema, ya que esta es suministrada mediante unas pilas externas, el cual da un voltaje de salida de 5 voltios estables, por lo que es posible alimentar la placa mediante el cable USB. El sensor LIDAR por su bajo consumo de energía, al ser controlado por PWM es totalmente seguro, ya que se encuentre alimentado directamente por alguno de los pines que brinde los 5 voltios tal como se muestra en la Figura 30, además la comunicación se establece mediate serial 1 esto quiere decir TX1 y RX1.

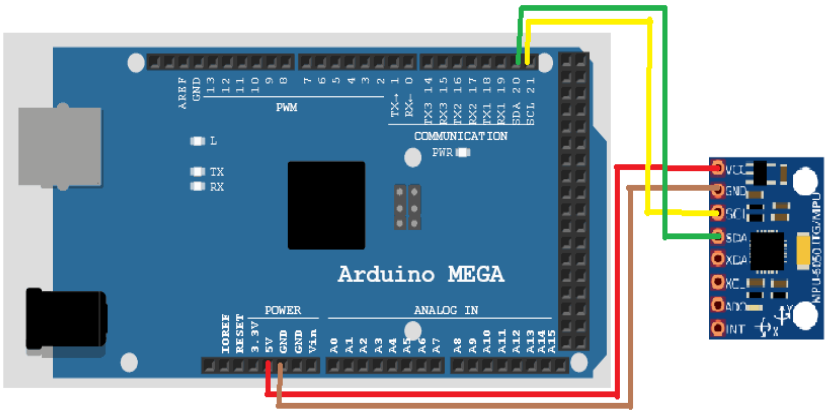
Figura 30 : Conexión física sensor LIDAR



Fuente: (Brampton, 2022)

- El módulo MPU6050 obligatoriamente usa los pines SCL y SDA por el protocolo I2C que este maneja de intentar realizar la conexión mediante un puerto serial será necesario utilizar algún método para adaptar el protocolo mencionado, lo que no es factible, la alimentación se la realiza a 5 voltios, como se muestra en la figura 31.

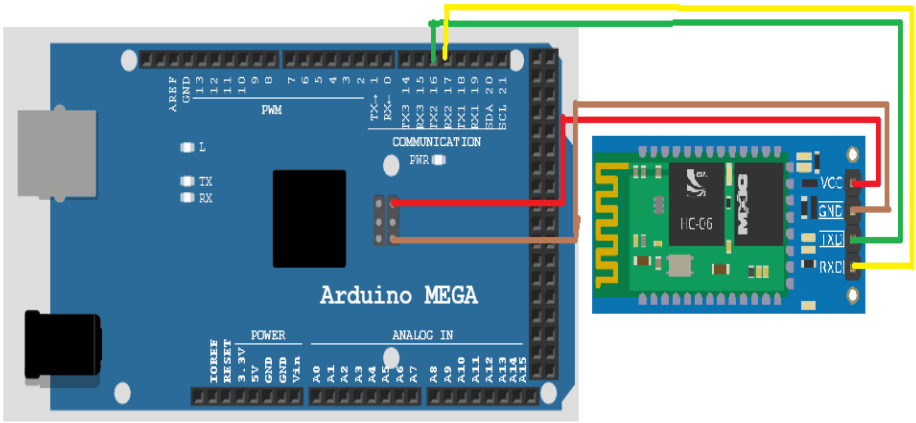
Figura 31 :Conexión física MPU6050.



Fuente: (DroneBot , 2015)

- El módulo HC06 ante la ocupación de todos los pines de alimentación regulares se recurre a realizar la conexión mediante los pines conocidos como TX y RX, tal como se muestra en la figura 32, cuyo objetivo principal es el de realizar la programación directa del microcontrolador por ello dispone de los pines de alimentación, el módulo HC06 al consumir una cantidad insignificante de corriente es totalmente seguro realizar la conexión hacia esos pines, la comunicación se realiza mediante serial 2 es decir TX2 y RX2.

Figura 32 :Alimentación HC06 mediante ICSP

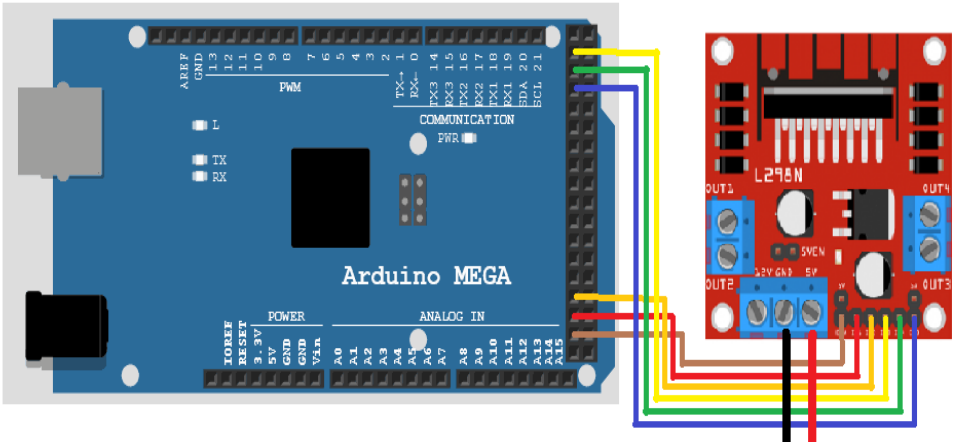


Fuente: (Orenda, 2016)

- Por último, el módulo de control de las llantas deberá de estar alimentada por una fuente diferente a la de Arduino, esto para evitar bajas de voltaje de manera inesperada que puedan llegar a afectar la toma de mediciones por parte de los

sensores además de cuidar la integridad de la placa, las conexiones se realizan tal cual se muestra en la figura 33.

Figura 33 : Conexión física modulo L298N.



Fuente: (Denysyuk, 2018)

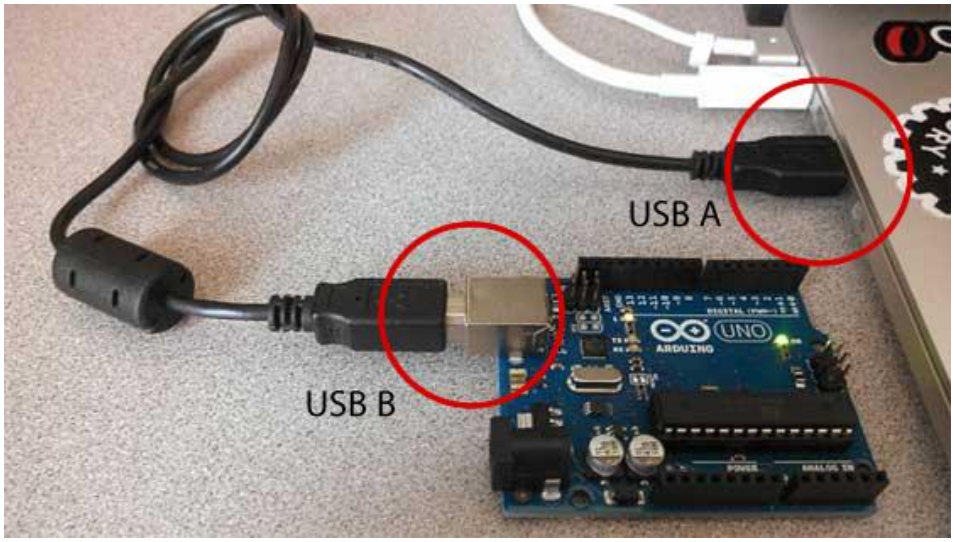
3.4 Comunicación Sensor

La comunicación de este tipo de sensor está basada principalmente en su programación previamente realizada, para generar el código hace uso de protocolos y bibliotecas dependiendo de su fabricación y uso interactivo, muchas de las formas para realizar la comunicación se basan en interfaces, a continuación, se explicará las interfaces más comunes en este tipo de sensores.

Interfaz serial: Es totalmente distinta la comunicación paralelo, la comunicación serial de manera resumida transmite y recibe los datos mediante un solo pin, en lugar de usar un pin para cada bit de los datos, se transmite de manera secuencial cada bit sincronizado de manera independiente, esto reduce drásticamente la cantidad de pines necesarios para realizar este tipo de comunicación, ya que usualmente solo se necesita uno o dos pines para funcionar, según el protocolo genera una consecuencia que da lugar a conectores mucho más pequeños como lo son el D 9 del RS 232 o el que se convertiría en el más usado en la actualidad el USB. Ahora para entender los protocolos, nos referimos al sistema utilizado para realizar la transmisión de datos y cada uno muestra diferentes configuraciones en comunicación y recopilación, existen diversos protocolos, pero los más usados son los siguientes: UART Y USART, interfaces de comunicación aplicadas en la electrónica.

Interfaz USB: este tipo de interfaz generalmente se utiliza para las comunicaciones y control de computadoras para información de datos, los conectores están basados para la comunicación de emisión y recepción, ya que facilita la transferencia de datos y el almacenamiento aun dispositivo previo conectado de manera física, generalmente existen diferentes tipos de USB: micro USB, USB tipo c, entre otros. Cada uno de estos conectores son diferentes dependiendo de cuál sea su propósito o dispositivo al que serán conectados, sin embargo, su interfaz para conectarse y recibir información será la misma. Este implemento como interfaz sirve cómo alimentación para dispositivos o placas electrónicas que serán energizadas y su valor oscilante de energía entregada está entre 500 miliamperios y 450 miliamperios. En resumen, es un dispositivo que almacena y recibe información para una placa o dispositivo electrónico, como se representa en la figura 34, a continuación, podemos observar la conexión mediante un cable físico del puerto A (computadora o dispositivo electrónico).

Figura:34 Comunicación placa externa



Fuente: (Llamas, 2016)

3.5 Diagramas de flujo

3.5.1 Intercambio de información

De forma general Arduino es el intermediario entre los sensores empleados y el programa de procesamiento final de los datos, recibe la información proveniente de los dos sensores utilizados, como primer paso realizamos un ligero procesamiento de los datos con el fin de

obtener los más relevantes, esto ya que ambos sensores brindan una cantidad de información considerable, no obstante, no toda es útil para el procesamiento a realizar.

El procesamiento llevado a cabo por Arduino tiene como objetivo extraer y filtrar los datos recibidos de mejor manera el espacio en la transferencia y procesamiento de datos hacia el programa principal Processing

Realizado todos los procesos y cálculos por parte de Processing, con los datos brindados por Arduino el cual después recibe la información sobre que actuadores poner en movimiento y de qué forma hacerlo para llevar a cabo la actividad para el cual fue diseñado. Este constante intercambio de información tal como se muestra en la Figura 35, se traduce a una comunicación bidireccional entre Arduino y Processing, lo que da como resultado un control y coordinación correcta de los actuadores en función del procesamiento de los datos.

Figura 35 :Comunicación bidireccional

Fuente: Autores

3.6 Programación de sensores

3.6.1 RPLIDAR A1M8

Su correcto funcionamiento implica la utilización de librerías externas estas son RPLIDAR.h el cual es proporcionado por el fabricante y LIDARMeasure.h. La librería RPLIDAR.h es la encargada de la comunicación con el sensor LIDAR atreves del puerto serial Serial1, a diferencia de otros sensores de menor coste este necesita recibir una señal de confirmación para iniciar la recolección de datos por ello provee de funciones y métodos para lograr interactuar con el sensor por otra parte la librería LIDARMeasure.h, se encarga del procesamiento y filtrado de datos para eliminar mediciones no deseadas dando como

resultado la estructura “LIDARMeasurements” la cual contiene la distancia y su respectivo ángulo.

Al implementar la librería otorgada por el fabricante, se obtendrá muchos datos erróneos como se puede observar en la Figura 36, donde se debería poder observar un cuadrado casi perfecto como se muestra en la Figura 37, ya que se le está ubicando en una caja de pruebas correspondientes, además el fabricante sugiere utilizar objetos de color blanco, ya que estos reflejan el haz de luz de mejor manera.

Figura 36 :Datos erróneos del mapeo LIDAR

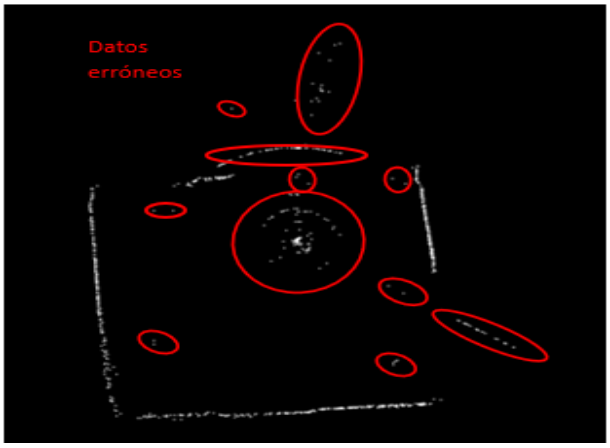
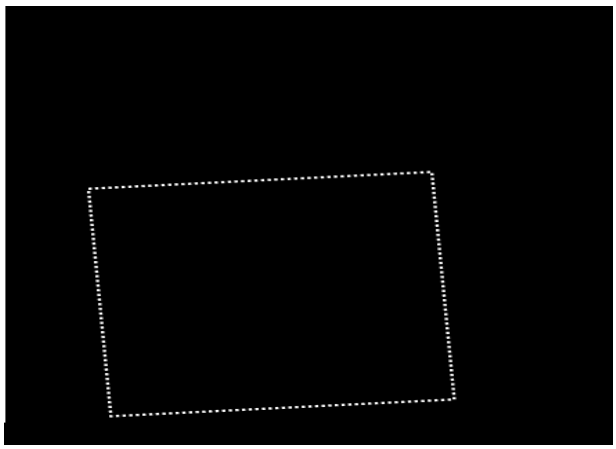


Figura 37 :Mapeado esperado del sensor



Fuente:Autores

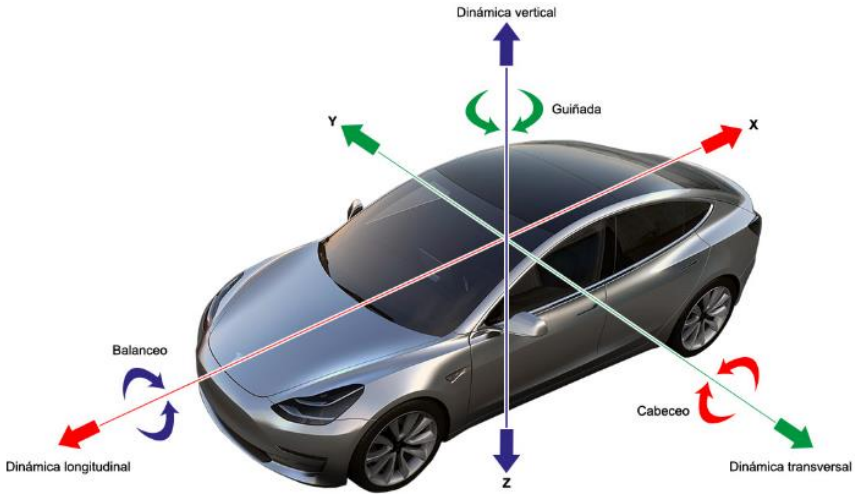
3.6.2 MPU6050

Al igual que el sensor LIDAR es necesario una librería, en este caso propio de Arduino para lograr controlar el módulo mediante el protocolo I2C, dicha librería tiene como nombre Wire.h sin embargo, se emplea conjuntamente la librería I2Cdev.h para simplificar y facilitar la interacción con el sensor, a breves rasgos este protocolo permite una comunicación bidireccional digital, lo que permite a Arduino ingresar a los registros del MPU5060 para configurarlo y obtener los datos que se encuentren en su interior además este módulo actúa como esclavo. Se utiliza la función “mpu.initialize()” para establecer la configuración inicial y estar listo para su uso, una vez configurado se utiliza la función “mpu.dmpGetCurrentFIFOPacket(fifoBuffer)” para obtener los datos del sensor y realizar

el procesamiento para obtener el ángulo de giro del sensor, con el cual se logrará tener con certeza el angulo, el cálculo para la obtención de los ángulos se lo hace mediante la aplicación de la biblioteca “MPU6050_6Axis_MotionApps20.h”.

Continuando con el apartado, se utiliza la orientación en forma de ángulos de Euler los cuales incluyen los ángulos de cabeceo, balanceo y guiñada, en el presente proyecto se utiliza el angulo de cabeceo sobre todo por el codigo utilizado, ya que el ángulo mencionado es el que describía la orientacion del vehículo a pesar de que realmente dicho ángulo debería ser el de guiñada, como se observa en la figura 38. Los ángulos entregados por el sensor están expresadas en radianes, por lo que se deben de multiplicar por ciento ochenta y dividirlo por pi, adicional se realiza una tranformación a número entero mediante la función “static_cast<int>”, esto para optimizar el procesamiento de datos en Processing considerando que los decimales no afectan la graficación de la orientación del vehículo.

Figura 38 :Ángulos de Euler en el vehículo.



Fuente: (Millán Valbuena, 2020)

3.7 Recursos técnicos y economía

Este tipo de proyecto tiene una gran aplicación en diferentes ámbitos por sus diferentes aplicaciones en el mapeo y detección de objetos, generalmente la aplicación de estos sensores tiene como finalidad el mismo protocolo, el uso del sensor LIDAR para la recolección de datos en tiempo real y por consiguiente el envío y la recepción de datos a un

dispositivo electrónico, en muchas ocasiones una computadora. La realización de este proyecto es similar en sus recursos técnicos en comparación con otros, al hacer uso de tecnologías y sensores existentes en el comercio, esto facilita al ensamble y adaptación correspondiente para su trabajo de sondeo y mapeo de lugares específicos. Por otro lado, en los recursos técnicos nos basamos esencialmente en el uso y aplicación que se le dará, al comparar que este tipo de vehículo no es convencional como un auto a control remoto.

A continuando con los recursos técnicos y su variación de aplicaciones en nuestra actualidad este tipo de sensores ha sido empleado tanto en aspiradoras robot, vigilancia autónoma y detección de movimientos, para el mapeo de territorios lugares queda en un segundo plano secundaria, ya que esto es generado mediante la obtención de datos proporcionados por el sensor. Además, los recursos técnicos y costos realizados en la implementación de ese proyecto son representados en la tabla 1 a continuación

Tabla 1: Componentes del proyecto técnico

Tipo	**Descripción**	**Costo**
Equipos	Sensor LIDAR A1M8	$ 110.00
	Arduino Mega	$ 20.80
	Módulo bluetooth HC 06	$ 6.70
	Giroscopio MPU6050	$ 3.50
	Kit carro robot 4 ruedas	$ 15.00
	Pila 1600 mAh	$ 3.00
	Módulo L298n	$ 4.70
	Módulo de carga	$ 3.00
	Jumpers	$ 3.58
	Switch	$ 0.50

Tipo	Descripción	Costo
	Baterías 9 V	$ 4.80
	Conectores baterías 9 V	$ 0.50
Materiales	Cartón para los obstáculos.	$ 2.00
	Masilla epóxica	$ 1.49
	Cinta doble faz	$ 1.80
	Estaño	$ 0.80
	Termo encogible	$ 0.40
	Cinta aislante	$ 0.65
	Pegamento instantáneo	$ 1.00
	Amarras plásticas	$ 0.15
	1 metro cable para parlante	$ 0.40
	TOTAL	$ 184.77

Fuente: Autores

En la Tabla 1, se puede constatar el presupuesto necesario para lograr replicar el presente proyecto, además sus componentes son de fácil adquisición excepto el sensor LIDAR ya que este no se encuentra con facilidad en el mercado ecuatoriano, sin embargo, es posible reemplazarlo con cualquier otro siempre y cuando brinde un gran número de lecturas por segundo o revolución, además habrá que realizar los cambios necesarios en el código para lograr tener los resultados óptimos. Un punto para tener en cuenta es la variación de precios que se puede tener dependiendo del lugar donde se adquieran los componentes, pero dicha variación no es significante.

En resumen, con la tabla anterior mencionando sus aplicaciones en el campo laboral, en este tipo de prototipo puede eliminar algunos componentes, como por ejemplo se puede eliminar el chasis del vehículo siempre y cuando el sondeo y mapeo de datos esta referido a un punto designado sin moverlo, además de poder eliminar el puente H con las debidas restricciones, si el motor no utiliza demasiada fuerza para su movilización la placa Arduino será suficiente para poder realizar el movimiento del vehículo, en cuanto al desarrollo y diseño del chasis, el mismo puede ser cambiado por uno más pequeño o más compacto que utilice sólo 2 motores y 2 ruedas. Finalmente, si se necesita agilizar o reducir costos en la fabricación, se tiene que buscar un punto de equilibrio donde no sea afectado la adquisición de datos y el movimiento del vehículo al mapear el área o campo.

CAPÍTULO IV

4 RESULTADOS

4.1 Lectura de datos

La lectura de datos obtenidos por el sensor LIDAR fue determinada mediante el testeo y mapeo de la zona correspondiente, además de tomar en cuenta el entorno al que fue sometido, se toma consideraciones tanto de los objetos y el terreno utilizado para las pruebas. A continuación, en la figura 39, podemos observar la vista de los objetos tomados a considerar en el testeo, tanto de la vista superior y de la vista posterior en el campo utilizado para su prueba. El reconocimiento obtenido por el sensor LIDAR, al tener un punto fijo en su lectura de datos logró determinar los objetos a los cuales puede reconocer y además determinar la distancia a la cual están fijados.

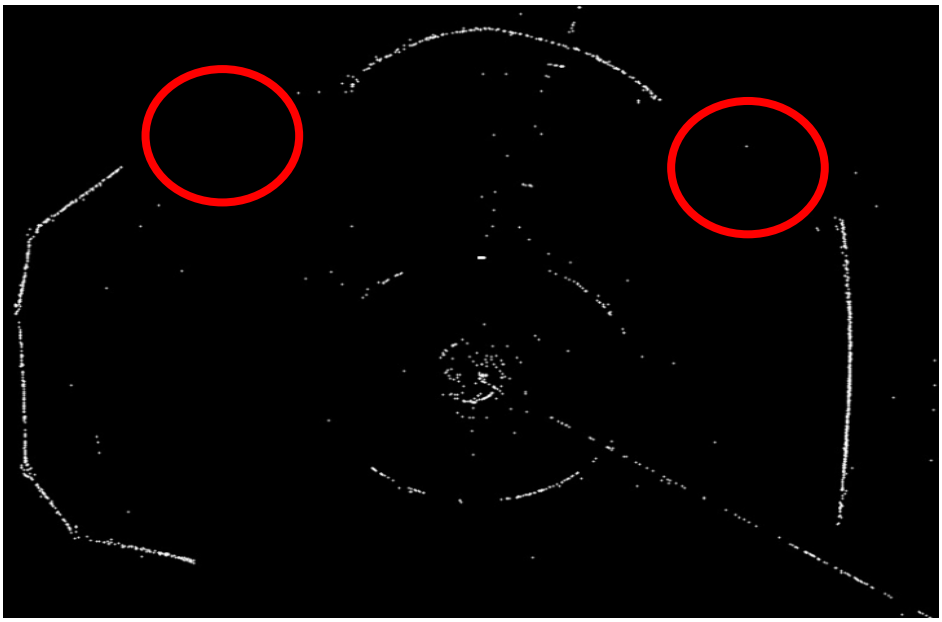
Figura 39 : Prueba de objetos testeados

Fuente: Autores

Para esta primera prueba se consideró una distancia mínima de un rango establecido entre los 15 centímetros a 18 centímetros, para poder determinar objetos a los cuales pueda reconocer con mayor facilidad y aquellos que no puede identificar, el mapeo generado en la primera prueba fue establecido con una previa programación en Arduino y Processing y fijándolo con un eje cero de rotación el automóvil a escala ensamblado, es decir, los datos recopilados por el haz de luz cómo eje central de datos podía desplazarse sin ningún error y los datos recolectados no serían erróneos, la única modificación que se observaría es la distancia de los objetos captados. Seguidamente los datos recolectados fueron graficados en la interfaz Arduino en dónde se observa la distancia del objeto y los alrededores captados,

en la figura 40 a continuación, se observará el primer testeo de datos en tiempo real captados por el sensor LIDAR, representados en dos dimensiones, tanto para el eje x como en el eje y, se podrá conocer los objetos planteados como obstáculos.

Figura 40 :Representación de datos



Fuente: Autores

En esta representación podemos notar una gráfica en el plano, donde sólo pudo captar a 3 objetos de los 5 destinados para la primera prueba de mapeo, al considerar este tipo de error, se denotó qué objetos transparentes o de baja coloración no serán captados con facilidad, al ser transparentes y no contar con un color fijo el láser reflejado hacia estos objetos los traspasa y no rebota emitiendo los datos hacia el sensor LIDAR.

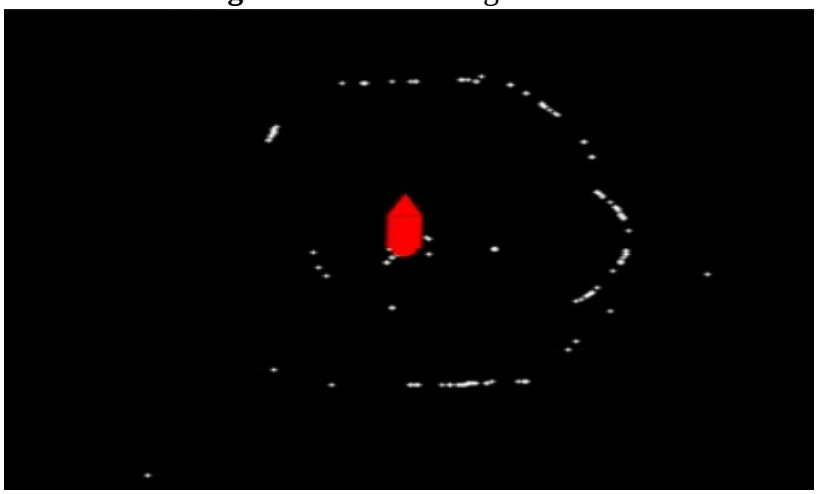
A continuación, se implementará el cambio de los objetos, por otros de mayor color y al determinar las especificaciones del fabricante, se tomó la consideración de hacer uso de objetos en color blanco, al saber que este sensor tiene una sensibilidad de reflejo con este tipo de color, se procedió al cambio de las botellas plásticas, por latas de pintura en color blanco y seguidamente se procedió a generar un nuevo mapeo y recolección de datos, con este tipo de cambio realizado en el entorno de pruebas para el sensor se muestra un cambio en la figura 41.

Figura 41: Reconocimiento de datos

Fuente: Autores

Al tomar en consideración las especificaciones de fabricación del sensor LIDAR se pudo obtener resultados más favorecedores al hacer el cambio y uso de objetos en color blanco, en consecuencia, se pudo obtener un esquema en donde cada uno de los objetos planteados fue reconocido sin ningún problema. Al poder determinar el espacio de los implementos, denotamos irregularidades al momento de generar un mapeo en dos dimensiones mostrado en la figura 42, se ubicó objetos irregulares que no concuerdan con los planificados para esta primera prueba, lo cual dio como resultado datos erróneos a considerar para el siguiente avance de pruebas. Además, para generar una mejor orientación de datos, se aplicó una flecha guía como punto de referencia para el vehículo ensamblado, así se determinará con mayor facilidad los datos obtenidos.

Figura 42 : Datos irregulares

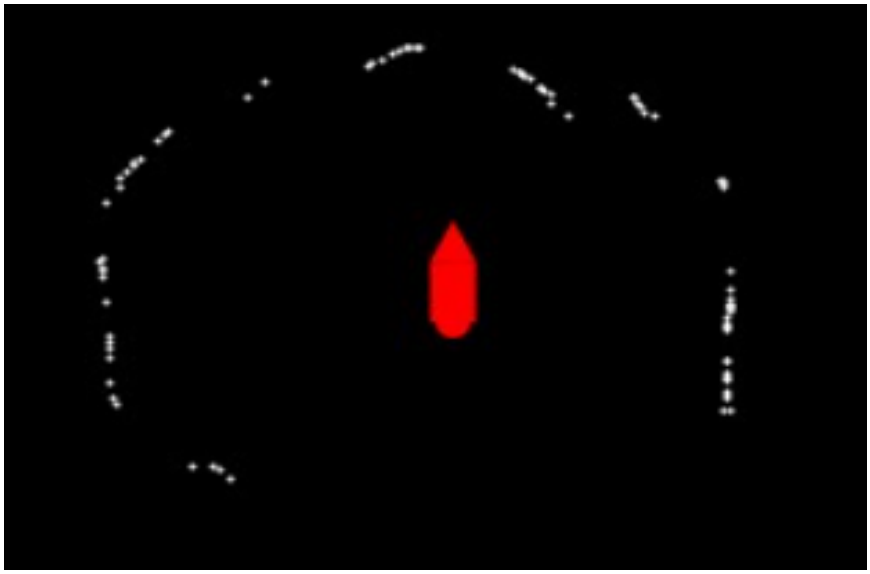


Fuente: Autores

Para la corrección de los datos erróneos de la figura 42, se aplicó los siguientes filtros de librerías basados en rangos de medición en Arduino, entre 151 milímetros y 9000 milímetros esto para obtener datos en concreto, ya que en Processing solamente muestra mediciones hasta un máximo de 1500 mm, el valor de 151 milímetros fue colocado en base al fabricante y menciona que la medición mínima es de 150 milímetros hasta un máximo de 12000 milímetros. Además, se pudo identificar puntos adyacentes que grafican puntos al azar causado por varios motivos como, por ejemplo: la luz del ambiente, misma que tiene una captación errónea del haz de luz, por ello al ser puntos aislados ponen la condición de que el punto puede ser válido, si existen más 50 puntos a su alrededor, a esto se pudo identificar que la distancia de umbral para considerar un punto adyacente idóneo es de 150 milímetros.

Al tomar en consideración las correcciones y aplicación de filtros en la programación, se pudo obtener un barrido de datos más concretos y exacto con los objetos planteados en la prueba de campo, la orientación y la nueva representación del plano están presentadas en la figura 43, la misma tiene datos más exactos de los objetos captados dentro de la zona de detección del sensor LIDAR.

Figura 43: Corrección de datos erróneos

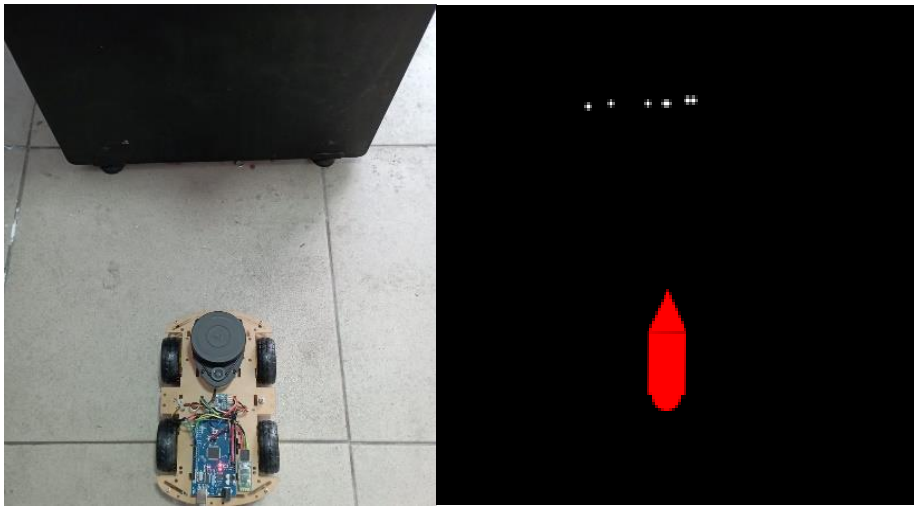


Fuente: Autores

4.2 Pruebas MPU

Las pruebas realizadas comprenden dos requerimientos el primero indicar cuál es la orientación del robot y el segundo brindar el dato necesario para que el mapa de puntos genere una dirección sin importar la posición del vehículo, siempre tendrá la misma orientación. Para ello se ubica el robot apuntando hacia un elemento fijo en este caso el panel de color negro tal como se muestra en la Figura 44.

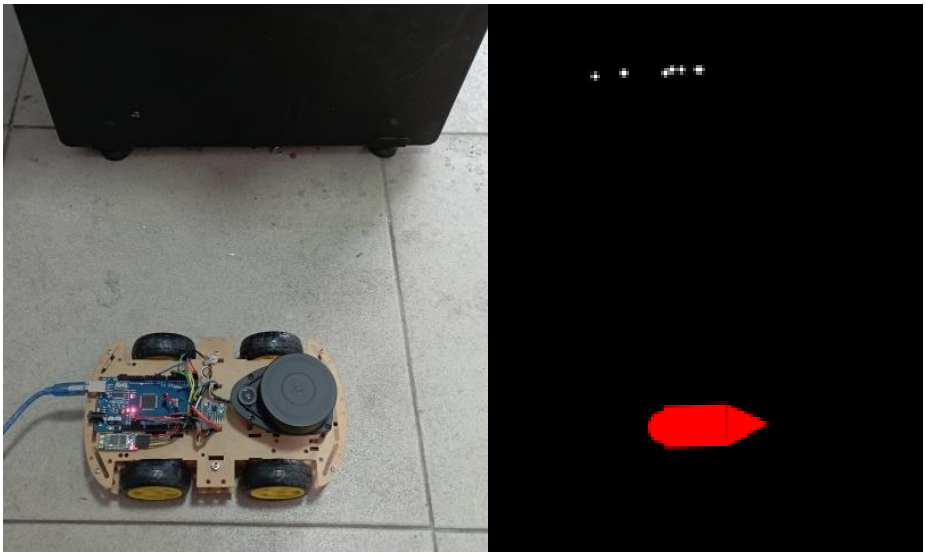
Figura 44 :Ambiente físico y representación en Processing 1



Fuentes: Autores.

En la Figura 44, se presenta que la orientación de la flecha es la misma que tiene el vehículo además tomar en cuenta en que orientación, se presentan los puntos del panel negro. A medida que el robot tome otra dirección se tendrá que ver mediante la flecha y el panel negro, en este caso la flecha no cambiará a ninguna posición, ya que tiene un eje centrado para el movimiento del vehículo en conjunto con una programación fijan la flecha, es decir, si el vehículo cambia de posición ya sea en la parte izquierda, derecha o al ejercer un giro de 360° el mismo seguirá compatible con la posición de la flecha.

Figura 45 :Ambiente físico y representación en Processing 2



Fuente: Autores

En la representación de la Figura 45 se observa el cambio de dirección por parte de la flecha obedeciendo al cambio de orientación del vehículo por lo que la prueba es superada con éxito al representar con exactitud lo que está sucediendo en el ambiente físico además la gráfica del panel negro.

CONCLUSIONES

La tecnología implementada en el prototipo a escala reflejando un nivel de autonomía nivel 2, al ejecutar un recorrido o trayectoria fija en un carril, se obtuvo un resultado de una asistencia de frenado por parte del sensor y dirigida a la conducción realizada, además de vehículo a escala se demostró que esta tecnología es de recursos en tiempo real, la misma que pude interrumpir el movimiento del vehículo fijando una alternativa de conducción más eficiente, al mismo tiempo de evitar colisiones o desfaces en la trayectoria establecida para la conducción del vehículo.

El uso de bibliotecas en código abierto generaron una escritura más concreta para la interacción y recepción de datos por parte del sensor LIDAR, dando un paso para aplicación de monitoreos y fijación de objetos en tiempo real de la conducción autónoma, es así, que se obtuvo resultados de almacenamiento de datos filtrados y sin información errónea, ya que los datos recopilados pasan el software JAVA generando un segundo filtrado de datos, lo cual demostró una mejora en el sistema y gestión de planos en 2D, perfeccionando el manejo de partículas o puntos para un futuro proyecto de autonomía y conducción.

Al hacer uso de la tecnología LIDAR en el prototipo de vehículo, se determinó que el mismo muestra resultados de recolección de datos y por consecuencia, la placa Arduino toma el trabajo de recetor y codificador del lenguaje programado por Processing, el comportamiento de este tipo de sensor fue fijado en una antena receptora y captadora de objetos, lo cual implica una conexión física y una capacidad de procesamiento de lectura estandarizada que genero resultados de un manejo de datos procesado por dispositivos y visualizados en una pantalla o interfaz de usuario establecida en la programación.

Para la parte práctica se tomó en consideración objetos fijados en la trayectoria, denotando sus diferentes dimensiones y colores, de esta manera se obtuvo los resultados de un mapeo erróneo y otros más exactos, al hacer uso de botellas plásticas el sensor tuvo un efecto transparente, al no poder detectar objetos con un grado de color bajo los mismos no fueron marcados en plano de datos, por otro lado, al hacer uso de objetos sólidos y de color blanco el comportamiento del testeo del área reconoció cada uno de los objetos marcados alrededor del sensor LIDAR y por consecuencia se generó una ruta alterna para la conducción autónoma del vehículo.

RECOMENDACIONES

Para el uso del sensor líder y su tecnología de pulsos de láser el cual mide distancias y genera mapeos en tiempo real con una alta precisión, se menciona algunas recomendaciones para poder tener un mayor resultado de datos como lo son:

- Se recomienda leer las especificaciones técnicas del fabricante, dependiendo de qué sensor LIDAR se ha utilizado, en muchos casos varía la resolución, la frecuencia de muestreo, el ángulo de visión y el alcance de distancia. Al leer este tipo de especificaciones podremos determinar el uso que se le dará en el campo o proyecto.
- Para poder tener una mejor calidad de imágenes, se recomienda calibrar adecuadamente el sensor sea manualmente o ingresándolo con un puerto directamente a la computadora, además se debe tomar en cuenta una calibración cada 15 días, ya que el eje central del sensor tiende a perder la comunicación al momento de encenderlo y ponerlo en movimiento.
- Se debe tomar en cuenta las condiciones en las que está expuesto el sensor, ya que no es apto para las lluvias, ni un excesivo polvo, al considerar este tipo de condiciones ambientales se podrá rendir con mayor facilidad el uso y mediciones más precisas, evitando un deterioro más rápido del sensor.
- Se debe establecer un punto fijo para tener una mayor comodidad y recepción de datos, es posible que la cámara necesite un ángulo de visión más específico por lo que se requiere un campo de pruebas totalmente amplio, por el contrario, otro tipo de sensor puede requerir un lugar más estrecho esto dependerá directamente de las especificaciones del manual del sensor.
- Tener el lente del sensor limpio y protegido con una cubierta o un protector glass, además de limpiar regularmente la banda de giro y evitar que sufra un desgaste prematuro por no limpiarlo adecuadamente en su período de tiempo, se los recomienda limpiar cada semana o 2 semanas dependiendo del ambiente en el que esté expuesto.
- Generar ciertas pruebas de vibraciones, como un testeo de inicio nos determinan si los datos recolectados sufrirán variaciones o pérdida de datos al momento de mapear el lugar, esto es de gran ayuda, ya que evitamos recolección de datos erróneos y podemos reajustar cada dato obtenido mediante una programación previa.

REFERENCIAS BIBLIOGRÁFICAS.

Ahmed, H. O. (2020). FLS-Based Collision Avoidance Cyber Physical System for Warehouse Robots using FPGA. *Proceedings - 2019 6th International Conference on Dependable Systems and Their Applications, DSA 2019*, 262–268. https://doi.org/10.1109/DSA.2019.00040

Avella, G. (2021). La tecnología LIDAR. *SIstemas de Empotrados*, *0*(0), 63.

Carlos, A., Nube, L., Goyas Gutiérrez, M. A., Jhonny,) ;, & Vargas Cruz, D. (2013). *Escuela Superior Politécnica Del Litoral Facultad De Ingeniería En Electricidad Y Computación*.

Enríquez Herrador, R. (2009). *Guía de Usuario de Arduino*.

Ferazi, M., & Dahoud, A. (n.d.). Entorno de desarrollo integrado “IDE” para Arduino. *Al Zaytoona University*, *0*(0), 12.

Gil, J. J. (n.d.). *Controlador H ´ aptico Educacional basado en la Plataforma Arduino*.

Herranz, J. C. H. (2015). Una mirada al mundo Arduino. *Tecnología y Desarrollo*, *13*(0), 21. https://revistas.uax.es/index.php/tec_des/article/view/617

Planzer, R. (2005). La seguridad vial en la región de América Latina y el Caribe: situación actual y desafíos. *Recursos Naturales e Infraestructura*, *0*(0), 102.

Rica, C., Vial, S., Rica, E. C., Costarricense, C., Social, S., & Ccss, L. (2008). Accidentes de tránsito. *Acta Pediátrica Costarricense*, *20*(1), 2007–2009. http://www.scielo.sa.cr/scielo.php?script=sci_arttext&pid=S1409-00902008000100001&lng=en&nrm=iso&tlng=es

Roriz, R. (n.d.). Automotive LIDAR Technology A Survey (completo).en.es.pdf - Lumin | Lumin PDF Editor. *TRANSACCIONES IEEE EN SISTEMAS*, *0*(0), 16. Retrieved May 29, 2023, from https://app.luminpdf.com/es/viewer/646ba017bdcce102367e4e45

Roriz, R., Cabral, J., & Gomes, T. (2014). Tecnología automotriz LIDAR: una encuesta. *TECNOLOGÍA LIDAR AUTOMOTRIZ: ENCUESTA*, *1*(0), 1–2.

Ruan, J., Cui, H., Huang, Y., Li, T., Wu, C., & Zhang, K. (2023). A review of occluded objects detection in real complex scenarios for autonomous driving. *Green Energy and Intelligent Transportation*, *2*(3). https://doi.org/10.1016/j.geits.2023.100092

Ruiz Guti, M. (2007). Manual de Programación Arduino Arduino : Manual de Programación. *Arduino Notebook*, *1*, 3–70.

Widodo, N. S., Akbar, S. A., & Rahman, A. (2018). Robot Operating System (ROS) Compatible Low Cost Rotating Light Detection and Ranging (LIDAR) Design. *IOP Conference Series: Materials Science and Engineering*, *384*(1), 6–11. https://doi.org/10.1088/1757-899X/384/1/012058

ANEXOS.

ANEXO 1
1.1 PROGRAMACION ARDUINO

Una de las partes más importantes es la de inclusión de las librerías, estas se utilizan para que el programa en este caso ArduinoIDE tenga todas las herramientas para la correcta ejecución de las posteriores líneas de código, en este caso se tiene RPLIDAR.h la cual como se mencionó anteriormente es la encargada de proporcionar las herramientas para la conexión con el sensor, posteriormente se incluye la librería LIDARMeasure.h la cual es la primera línea de filtrado a pesar de ser muy ligero el proceso que realiza ya elimina ciertos puntos no necesarios. El giroscopio necesita de la librería I2Cdev.h para permitir la conexión mediante protocolo I2C al mismo tiempo necesita de la librería MPU6050_6Axis_MotionApps20.h para realizar la generación de ángulo de guiñada, cabeceo y alabeo. La última librería en utilizarse es Wire.h la misma trabaja en conjunto con los anteriores para el giroscopio la cual provee de los procesos que en dado caso pueda llegar a no tener la librería I2Cdev.h.

Posteriormente dentro del mismo bloque se declaran las variables y pines a utilizar, se los conocen como variables globales, se declaran variables para el sensor LIDAR y las necesarias para el cálculo del ángulo además de los necesarios para controlar los motores para saber la orientación del vehículo.

En la sección de void setup () se configura todo lo que son pines es decir establecer si serán de envío o recepción de información y señales, también se configura la velocidad de transmisión y recepción la cual se configura a 115200 baudios, todos los elementos a utilizar deben de estar a la misma velocidad caso contrario se tendrá errores de comunicación, cabe recalcar que este bloque de configuración solo se realiza una vez al iniciar el programa.

El void loop () es el que lleva las instrucciones que se repetirán indefinidamente en este caso, ya que se necesita que constantemente esté enviando y recibiendo información tanto de los sensores como de Processing, el primer if que se encuentra es el responsable de obtener los datos del giroscopio y procesarlo para solamente obtener el de cabeceo. En el siguiente if se obtienen los datos del sensor LIDAR caso contrario de ocurrir alguna eventualidad se generará el mensaje de “Error al obtener las medidas del LIDAR”, de darse el proceso

correctamente se da paso al siguiente if donde se filtra los datos en razón a un cierto rango de distancia, en el cual el valor mínimo debe de ser 151 mm ya que la medida mínima que reconoce correctamente el sensor es de 150 mm, el valor máximo puede ser de 9000 mm ya que con esos valores se obtienen los máximos posibles y con ello tener más información para procesar, como paso siguiente se imprimen los datos a través del puerto serial por lo que ya serán lo que va a tomar Processing.

El siguiente bloque es el encargado de recibir los comandos atreves del puerto serial para el movimiento de las ruedas es por ello que se crea un nuevo ciclo llamado void executeCommand el cual al recibir cierto comando desde Processing por el puerto serial mismo que puede ser w,s,a,d y sus variantes en mayúsculas, al recibir el comando llama a las instrucciones respectivas para realizar el movimiento las cuales son moveForward(), moveBackward(), turnRight() y turnLeft() respectivamente en caso de que no se reciba ningún comando relacionado es decir un espacio en blanco o cualquier otra letra se procede a ejecutar los comandos stopMotors().El código presentado está destinado principalmente a la programación de la placa Arduino mega, y su código respectivo está mostrado a continuación:

1.2Código_ Arduino

```
#include <RPLIDAR.h>

#include "LIDARMeasure.h"

#include "I2Cdev.h"

#include "MPU6050_6Axis_MotionApps20.h"

#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE

#include "Wire.h"

#endif

#define RPLIDAR_SERIAL_PORT Serial1  // Puerto serial utilizado para la comunicación con el LIDAR

#define BAUDRATE 115200  // Velocidad de baudios para la comunicación serial
```

```
RPLIDAR LIDAR;

Measure_t LIDARMeasurements;

MPU6050 mpu;

#define INTERRUPT_PIN 2

bool blinkState = false;

bool dmpReady = false;

uint8_t mpuIntStatus;

uint8_t devStatus;

uint16_t packetSize;

uint16_t fifoCount;

uint8_t fifoBuffer[64];

Quaternion q;

VectorInt16 aa;

VectorInt16 aaReal;

VectorInt16 aaWorld;

VectorFloat gravity;

float euler[3];

float ypr[3];

volatile bool mpuInterrupt = false;

void dmpDataReady() {

mpuInterrupt = true;}
```

```
int angleMPU = 0;

int IN1 = 50;  // IN1 a pin digital 50

int IN2 = 48;  // IN2 a pin digital 48

int ENA = 52;  // ENA a pin digital 52

int IN3 = 44;  // IN3 a pin digital 44

int IN4 = 42;  // IN4 a pin digital 42

int ENB = 46;  // ENB a pin digital 46

void setup() {

Serial.begin(BAUDRATE);

RPLIDAR_SERIAL_PORT.begin(BAUDRATE);

pinMode(RPLIDAR_MOTOR_CTL, OUTPUT);

digitalWrite(RPLIDAR_MOTOR_CTL, HIGH);  // Encender el motor del LIDAR

delay(1000);  // Esperar a que el LIDAR se inicialice

if (LIDAR.begin(RPLIDAR_SERIAL_PORT)) {

delay(500);  // Esperar 500 ms después de la inicialización para permitir que el LIDAR se estabilice }

#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE

Wire.begin();

Wire.setClock(400000);

#elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE

Fastwire::setup(400, true);

#endif
```

```
mpu.initialize();
pinMode(INTERRUPT_PIN, INPUT);
devStatus = mpu.dmpInitialize();
mpu.setXGyroOffset(220);
mpu.setYGyroOffset(76);
mpu.setZGyroOffset(-85);
mpu.setZAccelOffset(1788);
if (devStatus == 0) {
mpu.CalibrateAccel(6);
mpu.CalibrateGyro(6);
mpu.PrintActiveOffsets();
mpu.setDMPEnabled(true);
attachInterrupt(digitalPinToInterrupt(INTERRUPT_PIN), dmpDataReady, RISING);
mpuIntStatus = mpu.getIntStatus();
// Vaciar el búfer FIFO inicial
while (mpu.getFIFOCount() > 0) {
mpu.getFIFOBytes(fifoBuffer, packetSize);}
dmpReady = true;
packetSize = mpu.dmpGetFIFOPacketSize(); }
else {
Serial.print(F("DMP Initialization failed (code "));
```

```
Serial.print(devStatus);
Serial.println(F(")"));
  }
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENA, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(ENB, OUTPUT);
  Serial.begin(115200); // Establecer la comunicación serie a 115200 bps
}
void loop() {
  if (mpu.dmpGetCurrentFIFOPacket(fifoBuffer)) {
    mpu.dmpGetQuaternion(&q, fifoBuffer);
    mpu.dmpGetGravity(&gravity, &q);
    mpu.dmpGetYawPitchRoll(ypr, &q, &gravity);
    angleMPU = static_cast<int>(ypr[0] * 180 / M_PI);
  }
  if (LIDARMeasure(&LIDAR, &LIDARMeasurements, true)) {
    // Se obtuvieron las medidas correctamente
    // Verificar si la distancia está dentro del rango requerido (151 mm a 1500 mm)
```

```
    if (LIDARMeasurements.LIDARDistance >= 151 &&
LIDARMeasurements.LIDARDistance <= 9000) {
      Serial.print(LIDARMeasurements.LIDARDistance);
      Serial.print(",");
      Serial.print(LIDARMeasurements.LIDARAngle);
      Serial.print(",");
      Serial.println(angleMPU);
    }
  } else {
    Serial.println("Error al obtener las medidas del LIDAR");
  }
  // El código de control del motor se recibirá desde el puerto serie
  if (Serial.available() > 0) {
    char command = Serial.read();
    executeCommand(command);
  }
  delay(1);  // Retardo entre cada lectura del LIDAR
}
void executeCommand(char command) {
  switch (command) {
    case 'W':
      moveForward();
```

```
      break;
    case 'S':
      moveBackward();
      break;
    case 'A':
      turnLeft();
      break;
    case 'D':
      turnRight();
      break;
    default:
      stopMotors();
      break;
  }
}
void moveForward() {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(ENA, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
```

```
  digitalWrite(ENB, HIGH);
}
void moveBackward() {
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(ENA, HIGH);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  digitalWrite(ENB, HIGH);
}
void turnRight() {
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(ENA, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
  digitalWrite(ENB, HIGH);
}
void turnLeft() {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
```

```
  digitalWrite(ENA, HIGH);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  digitalWrite(ENB, HIGH);
}
void stopMotors() {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(ENA, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);
  digitalWrite(ENB, LOW);
}
```

ANEXO 2

2.1 PROGRAMACIÓN PROCESSING.

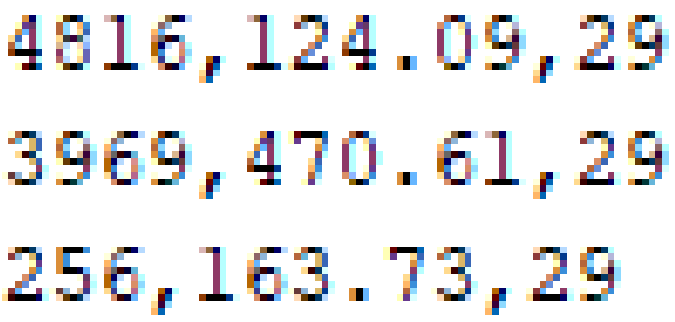
Al igual que en Arduino se necesita incluir una librería para lograr la comunicación serial la cual es import processing.serial, no se necesita agregar ninguna librería más para lograr el entorno gráfico dado que lo peculiar de este software es que está orientado a crear ventanas donde se puede mostrar múltiple información o gráficos dependiendo de la aplicación, al igual que Arduino se declaran todas las variables a utilizar.

En el apartado del void setup () se ubican líneas de configuración como lo es el puerto serial desde el cual se recibirá la información igualmente de específica los baudios a los que va a funcionar en este caso los mismos 115200 de Arduino, también se establece el tamaño de la ventana. El filtrado de datos se realiza con listas donde se recopilan los puntos recibidos y

puntos filtrados, cabe recalcar que este bloque solo se realiza una vez al dar inicio al programa.

El apartado void draw() es el encargado de realizar todos los procesos necesarios para el filtrado y presentación de la información de manera gráfica por tal razón se repite indefinidamente hasta que deje de recibir información por el puerto serial, el código el cual se encuentra comentado la funcionalidad de cada apartado se encuentra en el Anexo 2. Un punto importante a considerar es la manera en la que se recibe la información, ya que se encuentra en un formato específico el cual se presenta en la Figura 46.

Figura 46 Formato recepción de información.



Valores independientes, Fuentes: Autores

Como se observa en la Figura 46, cada fila es información independiente donde los valores se encuentran separados por una coma teniendo así como primer valor la distancia por ejemplo en la primera fila la distancia es 4816.124, el segundo valor es el ángulo de la distancia obtenida y por último el tercer valor es el ángulo de dirección del robot teniendo como unidades milímetros y grados en el caso de los valores de ángulo, la línea encargada de identificar cada valor de la fila recibida es if (values.length == 3) donde es necesario que lleguen los tres datos, en caso de necesitar más información esta tendrá que acoplarse al formato y modificar la línea mencionada para no tener inconvenientes.

Un detalle del sensor LIDAR es el punto de referencia desde el cual mide el ángulo de cada punto, ya que dicha referencia está en el sensor por lo cual al girar el robot también girara el mapa teniendo como resultado una imprecisa representación del mismo para corregirlo se implementa la línea de código rotate(radians(180)), además el ángulo de dirección del robot se presenta en un formato de 180 grados positivos y negativos por lo cual se es necesario

hacer una conversión para tener dichos datos en un rango de 360 grados siendo la línea encargada de este proceso la siguiente calculatedAngle = (this.mpuAngle + 360) % 360.

El procesamiento realizado para obtener los puntos finales para su presentación se lo realiza utilizando un método llamado puntos adyacentes, ya que observando la gráfica sin filtrar se reconoció un patrón donde los puntos erróneos se graficaban de manera aleatoria, pero de manera muy separada, el bloque encargado de este proceso es el boolean hasAdjacentPoint(PVector point), donde se establece un umbral para considerar un punto válido es decir si un punto no tiene cerca otro punto dentro del umbral establecido este será considerado como un dato erróneo y eliminado de la lista de datos a presentar, además con motivo de ahorrar espacio se borra toda la lista de los datos ya presentados para evitar saturar el procesamiento el cual se lo realiza mediante el proceso bajo el nombre de void filterValidPoints(int currentTime).

Por último se tiene el control de los motores mediante comando en este caso las letras del teclado teniendo w,a,s,d para ir hacia delante, derecha, atrás e izquierda respectivamente las cuales se encuentran en el ciclo void keyPressed() además si no se detecta que se presiona alguna tecla se ejecuta el ciclo void keyReleased() el cual envía un espacio en blanco, ya que en Arduino si llega a ingresar un comando diferente a los esperados simplemente los motores no se mueven.

2.2 Código Processing

```
import processing.serial.*;

Serial serialPort;

float maxDistance = 1500; // Valor máximo de distancia en milímetros (1.5 metros)

float referencePointSize;

ArrayList<PVector> LIDARPoints;

ArrayList<PVector> validPoints; // Lista de puntos válidos del LIDAR

int startTime;

int mpuAngle = 0;
```

```
float calculatedAngle = 0; // Variable para el ángulo calculado

boolean firstDataReceived = false;

boolean motorsActive = false; // Variable para controlar si los motores están activos

void setup() {

size(1000, 800); // Ajustar el tamaño de la ventana

background(0);

// Establecer la conexión con el puerto serial

serialPort = new Serial(this, "COM5", 115200);

// Tamaño del punto de referencia (rojo)

referencePointSize = 10;

// Inicializar las listas de puntos del LIDAR y puntos válidos

LIDARPoints = new ArrayList<PVector>();

validPoints = new ArrayList<PVector>();

// Iniciar el tiempo de inicio

startTime = millis();

}

void draw() {

// Limpiar la pantalla en cada iteración

background(0);

 // Centrar el punto de origen en el centro de la pantalla

 translate(width/2, height/2);
```

```
 // Escalar los puntos en relación con la distancia máxima
float scale = min(width, height) / (2 * maxDistance);
// Leer los datos del puerto serial
while (serialPort.available() > 0) {
String data = serialPort.readStringUntil('\n');
if (data != null) {
data = data.trim();
String[] values = data.split(",");
if (values.length == 3) {
float distance = float(values[0]);
float angle = radians(float(values[1]));
float mpuAngle = float(values[2]);
// Calcular el ángulo adicional (sin desplazamiento)
float additionalAngle = (mpuAngle + 90) % 360;
 // Calcular las coordenadas polares
float x = distance * cos(angle + radians(additionalAngle));
float y = distance * sin(angle + radians(additionalAngle));
 // Escalar las coordenadas
 x *= scale;
 y *= scale;
// Agregar el punto a la lista con el tiempo actual
```

```
LIDARPoints.add(new PVector(x, y, millis() - startTime));
// Actualizar el ángulo del MPU
this.mpuAngle = int(mpuAngle);
// Calcular el ángulo calculado
calculatedAngle = (this.mpuAngle + 360) % 360;
}}}
// Obtener el tiempo transcurrido
int currentTime = millis() - startTime;
// Filtrar y almacenar los puntos válidos del LIDAR
filterValidPoints(currentTime);
 // Mostrar los puntos válidos del LIDAR almacenados durante 1 segundo
stroke(255);
strokeWeight(2);
for (PVector point : validPoints) {
if (point.z <= currentTime && point.z >= currentTime - 1000) {
pushMatrix();
rotate(radians(180)); // Rotar los puntos en 180 grados
point(point.x, point.y);
popMatrix();
}}
// Dibujar el punto de referencia (rojo)
```

```
noStroke();
fill(255, 0, 0);
ellipse(0, 0, referencePointSize, referencePointSize);
// Dibujar la flecha girada según el ángulo del MPU
pushMatrix();
rotate(radians(mpuAngle));
drawArrow();
popMatrix();
// Limpiar los puntos después de 1 segundo
if (currentTime > 7000) {
LIDARPoints.clear();
validPoints.clear(); // Limpiar la lista de puntos válidos también
startTime = millis();
  }
// Imprimir el ángulo calculado en la consola
println("Ángulo calculado: " + calculatedAngle);
}
void filterValidPoints(int currentTime) {
  // Limpiar la lista de puntos válidos
  validPoints.clear();
  // Recorrer los puntos del LIDAR
```

```
 for (int i = 0; i < LIDARPoints.size(); i++) {
 PVector currentPoint = LIDARPoints.get(i);
// Verificar si el punto tiene un punto adyacente
 if (hasAdjacentPoint(currentPoint)) {
 validPoints.add(currentPoint); // Agregar el punto a la lista de puntos válidos
 }}}
boolean hasAdjacentPoint(PVector point) {
float thresholdDistance = 15; // Distancia de umbral para considerar un punto adyacente
// Recorrer los puntos del LIDAR
for (PVector otherPoint : LIDARPoints) {
// Excluir el punto actual y verificar la distancia con otros puntos
if (otherPoint != point && PVector.dist(point, otherPoint) <= thresholdDistance) {
return true; // Se encontró un punto adyacente
}}
return false; // No se encontró ningún punto adyacente
}
void drawArrow() {
float arrowSize = 100/3;
float headSize = arrowSize * 0.4;
float headOffset = arrowSize * 0.2;
 // Cuerpo de la flecha
```

```
rect(-headSize / 2, -arrowSize + headSize, headSize, arrowSize - headSize);
 // Cabeza de la flecha
triangle(-headSize / 2, -arrowSize + headSize, headSize / 2, -arrowSize + headSize, 0, -arrowSize);}
void keyPressed() {
if (!motorsActive) {  // Si los motores no están activos
if (key == 'w' || key == 'W') {
serialPort.write('W');  // Enviar comando 'W' para mover hacia adelante
motorsActive = true;  // Activar los motores
} else if (key == 's' || key == 'S') {
serialPort.write('S');  // Enviar comando 'S' para mover hacia atrás
motorsActive = true;  // Activar los motores
} else if (key == 'a' || key == 'A') {
serialPort.write('A');  // Enviar comando 'A' para girar a la izquierda
motorsActive = true;  // Activar los motores
}else if (key == 'd' || key == 'D') {
serialPort.write('D');  // Enviar comando 'D' para girar a la derecha
motorsActive = true;  // Activar los motores
}}}
void keyReleased()
```