**UNIVERSIDAD POLITÉCNICA SALESIANA**
**SEDE GUAYAQUIL**

**CARRERA DE INGENIERÍA ELECTRÓNICA**

**TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO ELECTRÓNICO**

**PROYECTO TÉCNICO:**

# "DISEÑO E IMPLEMENTACION DE SLAM Y CONTROL PID DE UN ROBOT AUTÓNOMO ROBOMASTER S1 UTILIZANDO PYTHON"

**AUTORES:**

JORGE ENRIQUE TUTIVEN REYES
DANIEL EUCLIDES VILLAGÓMEZ GALARZA

**TUTOR:**

ING. ORLANDO BARCIA MSc.

**GUAYAQUIL – ECUADOR**

**2021**

## Certificado de responsabilidad y autoría del trabajo de titulación

Nosotros, Jorge Enrique Tutiven Reyes y Daniel Euclides Villagómez Galarza, estudiantes de Ingeniería Electrónica de la Universidad Politécnica Salesiana certificamos que los conceptos desarrollados, análisis realizados y las conclusiones del presente proyecto de titulación son de exclusiva responsabilidad del autor y es propiedad intelectual de la Universidad Politécnica Salesiana.

Guayaquil, febrero del 2021

Jorge Enrique Tutiven Reyes
CI 0931021208

Daniel Euclides Villagómez Galarza
CI 0926607656

# Certificado de cesión de derechos de autores

Nosotros, Jorge Enrique Tutiven Reyes con documento de identificación Nº 0931021208 y Daniel Euclides Villagómez Galarza con documento de identificación Nº 0926607656, manifestamos nuestra voluntad de ceder a la Universidad Politécnica Salesiana los derechos patrimoniales en calidad de autor del proyecto de titulación titulado “DISEÑO E IMPLEMENTACIÓN DE SLAM Y CONTROL PID DE UN ROBOT AUTÓNOMO ROBOMASTER S1 UTILIZANDO PYTHON”, mismo que ha sido desarrollado para optar por el título de Ingeniero Electrónico con mención en Sistemas Industriales, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia, suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

Guayaquil, febrero del 2021

| | |
|---|---|
| _____ | _____ |
| Jorge Enrique Tutiven Reyes | Daniel Euclides Villagómez Galarza |
| CI 0931021208 | CI 0926607656 |

## Certificado de dirección del trabajo de titulación

Por medio de la presente doy a conocer que el Proyecto de titulación “DISEÑO E IMPLEMENTACIÓN DE SLAM Y CONTROL PID DE UN ROBOT AUTÓNOMO ROBOMASTER S1 UTILIZANDO PYTHON” presentado por los señores Jorge Enrique Tutiven Reyes con documento de identificación Nº 0931021208 y Daniel Euclides Villagómez Galarza con documento de identificación Nº 0926607656 para optar por el título de Ingeniero Electrónico con mención en Sistemas Industriales, se ajusta a las normas establecidas por la Universidad Politécnica Salesiana, por tanto, autorizo su presentación ante las autoridades pertinentes.

Guayaquil, febrero del 2021


_____

MSc Ing. Orlando Barcia
DIRECTOR DE PROYECTO DE TESIS

## Declaratoria de responsabilidad

Nosotros, Jorge Enrique Tutiven Reyes y Daniel Euclides Villagómez Galarza estudiantes de la Universidad Politécnica Salesiana de la Carrera de Ingeniería Electrónica mención Sistemas Industriales, declaramos que el trabajo de tesis aquí descrito ha sido desarrollada en su totalidad por nosotros, la cual es de nuestra autoría; no ha sido previamente presentada para ningún trabajo de grado; y hemos consultado las referencias bibliográficas que se detallan en este documento.

Guayaquil, febrero del 2021

| | |
|---|---|
| Jorge Enrique Tutiven Reyes | Daniel Euclides Villagómez Galarza |
| CI 0931021208 | CI 0926607656 |

## Dedicatoria

En primer lugar agradezco a Dios que me acompañó siempre con su bendición, a mis padres, los cuales siempre inculcaron en mi formación, principios, virtudes y valores, siendo su deseo en mí, verme convertir en un profesional, a mi esposa y a mi bendición Dylan Valentino Tutiven Arteaga, el cual amo con todo mi corazón, fue la inspiración y fuerza, habiendo estado a mi lado durante el desarrollo de mi tesis y en especial durante término de formación académica que es una etapa complicada, gracias a todos ellos por su apoyo incondicional ahora pude cristalizar este proyecto y anhelo de ser un profesional.

Jorge Enrique Tutiven Reyes

# Dedicatoria

Dedico este trabajo de titulación a Dios, que se encarga de siempre bendecirme, protegerme y sobre todo guiarme en cada paso que doy, permitiéndome disfrutar de cada logro y anhelo propuesto y a su vez de mi familia.

El presente trabajo de titulación es en memoria de mi abuelita Catuja Nicandra Ortega Cañola, ya que gracias a los valores infundado junto con el carácter y dedicación que siempre la describió, puedo decir que a pesar de ser mi abuelita eras mi segunda madre que para el desarrollo y formación académica tuve que lidiar con toda clase de obstáculos, y muchos de ellos los superé gracias a tus enseñanzas.

Daniel Euclides Villagómez Galarza

## Agradecimiento

Agradezco de todo corazón aquellas personas que estuvieron conmigo en esta etapa y fueron pieza fundamental para la culminación de esta tesis. Agradezco a Dios por brindarme las fuerzas día a día para seguir adelante, y aquellas mujeres bellas de mi vida, mi bella madre Carmen Reyes y mi bella esposa Milka Arteaga, gracias por su comprensión y paciencia.
A mis amigos que, con su compañía incondicional, palabras de aliento han sido importantes en la parte moral, y en especial a mi compañero de tesis Daniel Villagómez.

Jorge Enrique Tutiven Reyes

# Agradecimiento

A mis padres Reina Amelia Galarza Ortega y Gerardo George Villagómez Garcés, que son el pilar fundamental, sin su educación, sin sus sacrificios, sin sus consejos, sin su ayuda desinteresada yo no hubiese podido lograr todo lo que me he propuesto durante mi formación personal y académica.
A mis hermanos Gerardo Alberto Villagómez Galarza y Josué George Villagómez Galarza, que fueron mis mentores y quienes me guiaron junto a mis padres motivándome a no rendirme y siempre alcanzar mis metas.
A la Universidad, docentes, compañeros en general por todo lo anterior en conjunto con todos los copiosos conocimientos que me ha otorgado, en especial a mi compañero de proyecto de titulación Jorge Tutiven
A Carolina Andrade, que es la persona que ha sido sumamente importante con su apoyo incondicional inclusive en los momentos y situaciones más nefastas, no fue sencillo culminar con éxito este trabajo de titulación, sin embargo, siempre fuiste muy motivadora y esperanzadora, me ayudaste hasta donde te era posible, incluso más que eso.

Daniel Euclides Villagómez Galarza

## Resumen

Toda investigación en robótica tiene como objetivo común mejorar la autonomía de los robots. Para este fin trabajan en conjunto cada componente de hardware y software; partes fijas, partes móviles, sensores, actuadores, programas, herramientas, librerías, y siempre de la mano del programador que junta hábilmente todas las partes para lograr el funcionamiento deseado.

Este proyecto de titulación tiene como objetivo profundizar e investigar los robots programables aplicados al campo de la robótica móvil, para esto se implementó un robot autónomo programable “Robomaster S1”, combinado con placas electrónicas embebidas como Raspberry Pi, Arduino Nano, Pro Micro (una tarjeta Arduino Compatible), y dos herramientas: una física, el sensor YDLiDAR X4, para el mapeo en dos dimensiones del entorno que envuelve al robot; y una virtual, el framework ROS para conseguir la tele operación del robot desde una computadora conectada a su red.

El presente proyecto tiene la finalidad de analizar e interpretar el funcionamiento de hardware y software para la elaboración, desarrollo en programación Python de robots RoboMaster S1, que permitan obtener prototipos de robots con características de eficacia y fuerza para competir de manera satisfactoria en diferentes torneos, además de ayudar a los estudiantes a incrementar el interés en el desarrollo de programación Python, donde se combinan conocimientos de electrónica, programación a nivel medio y avanzado, complementando con diseños mecánicos, fomentando el interés en el diseño estructural.

El procesador es lo suficientemente potente para desempeñar diferentes tareas de manera simultánea, como procesar la inteligencia artificial,

interpretar las instrucciones de control e incluso realizar una transmisión de video a través de Wi-Fi, ya que integra una cámara de 5 megapíxeles, con una apertura f2.4 que permite enfoques a corta distancia y baja luminosidad, y un ángulo de visión de 120º que provee al robot un campo visual bastante amplio y mejora la experiencia del usuario que lo controla.

Palabras clave:

Robótica, ROS, LiDAR, Robomaster.

## Abstract

In every robotics research enhance the autonomy of robots is a common objective. For this purpose each and every component works together, hardware, software, fixed and moving parts, sensors, effectors, programs, tools, libraries, always hand in hand with the programmer who skillfully puts the parts together in order to obtain the expected operation.

The objective of the degree project is to deepen and investigate programmable robots applied to the field of mobile robotics, for this an autonomous programmable robot "RoboMaster S1" was implemented, combined with electronic boards like Raspberry Pi, Arduino Nano, Pro Micro (an Arduino based board), and a couple of tools, one physical, YDLiDar X4 sensor for mapping in two dimensions of the robot enviroment, and a virtual one, the framework ROS, for obtaining teleoperation of robot from a computer connected on its network.

The purpose of this project is to analyze and interpret the operation of hardware and software for the elaboration, development in Python programming of RoboMaster S1 robots, which allow obtaining prototypes of robots with characteristics of efficiency and strength to compete satisfactorily in different tournaments, in addition to helping students increase their interest in Python programming development, where they combine knowledge of electronics, intermediate and advanced level programming, complementing with mechanical designs, fostering interest in structural design.

The processor is powerful enough to perform different tasks at the same time, such as processing artificial intelligence, interpreting control instructions and even transmitting video over Wi-Fi, since it integrates a 5-megapixel camera,

an aperture f2.4 that allows close-range and low-light focusing, and a 120º angle of view that provides the robot with a fairly wide field of view and a better user experience for the operators.

Key words:

Robotics, ROS, LiDAR, Robomaster.

# Índice general

# Índice de figuras

# Índice de tablas

# Introducción

En pleno siglo XXI el desarrollo de la tecnología está presente en el día a día, la Ingeniería Electrónica ha incursionado en otras ciencias, como la ingeniería eléctrica y la mecánica, desarrollando así una rama importante conocida como la Robótica; su estudio ha llevado a conseguir avances que hace algunas décadas sólo eran posibles en nuestra imaginación.

En tal virtud se plantea a la Universidad Politécnica Salesiana SEDE GUAYAQUIL el trabajo de titulación: “DISEÑO E IMPLEMENTACIÓN DE SLAM Y CONTROL PID DE UN ROBOT AUTÓNOMO ROBOMASTER S1 UTILIZANDO PYTHON”.

El presente proyecto está enfocado a la migración y mejora de destrezas en programación en lenguaje Python, abriendo caminos a los estudiantes de la carrera de Electrónica y Automatización a fin de lograr de esta manera una motivación extra para que éstos diseñen sus propias programaciones y no dependan de tarjetas entrenadoras, consiguiendo con esto un desarrollo de ingeniería más meritorio y gratificante.

El proyecto complementa un análisis desarrollado con experimentos y pruebas, que crean un sistema didáctico de práctica para los estudiantes de las materias de programación, sistemas microprocesados, inteligencia artificial, de la misma forma que incentiva al club de robótica, de manera que los estudiantes aumenten sus destrezas, superen los obstáculos e incrementen sus expectativas respecto a la posibilidad de implementar cualquier tipo de proyecto que requiera el mismo o mayor nivel de complejidad.

# 1. EL PROBLEMA

## 1.1. Tema

“Diseño e implementación de SLAM y control PID de un robot autónomo Robomaster S1 utilizando Python”

## 1.2. Planteamiento del Problema

En la actualidad los estudiantes de la carrera de Electrónica y Automatización en la sede Guayaquil se encuentran en una continua evolución tecnológica de la mano de un pénsum académico que es constantemente actualizado. Incluir avances en el campo de la robótica en los planes de estudio de ingenierías ha llevado a innovar en el aprendizaje de manera continua, implementando nuevas tecnologías de software como lo es la programación en lenguaje Python. Por todo esto se vuelve notable la necesidad de crear espacios físicos y pedagógicos en el campus de la sede Guayaquil de la Universidad Politécnica Salesiana, dentro de los cuales el alumnado pueda desarrollar cómodamente estos proyectos, contribuyendo así a la implementación de un Laboratorio de Robótica Aplicada.

## 1.3. Antecedentes

Se indagó una serie de publicaciones científicas que brindaron soporte en el marco de estudio de nuestro proyecto de tesis, a continuación, se menciona las más relevantes para llevar a cabo la investigación a lo largo de este documento.

La tesis de grado publicada por [1] implementa un robot de 3 grados de libertad con fines educativos, utilizando la tecnología de impresión 3D; en el estudio se describe ampliamente los fundamentos matemáticos utilizados para comprender la cinemática del robot, y que serán muy útiles al momento de llevar a cabo el diseño, desarrollo y

comprensión del modelado matemático, y el análisis de los movimientos posibles en los grados de libertad del robot utilizado en nuestro proyecto.

El artículo científico de [2] publicado en la revista de la División de Ingenierías y Arquitectura la Universidad Santo Tomás en Bucaramanga, utiliza la tecnología de Mapeo y Localización Simultánea (SLAM por sus siglas en inglés) a través de un sensor LiDAR para la generación de mapas en dos dimensiones mediante escaneo omnidireccional y alcances de hasta 12 metros en un campo de visión de 360º, texto que ayudará a comprender cómo el robot utilizará este sensor para generar un mapa que lo ayude a evitar obstáculos durante la ejecución de tareas.

Finalmente, la tesis de [3] combina la tecnología SLAM con sensores LiDAR utilizando un conjunto de librerías de software y herramientas de código abierto, conocida como ROS (Sistema Operativo Robot) que facilita la programación de tareas como la navegación autónoma en robots móviles o la planificación de movimiento de robots manipuladores. De esta manera se brinda al robot una capacidad adaptativa para su correcto funcionamiento en diversos entornos y condiciones, sin necesidad de un operador que lo supervise constantemente.

## 1.4. Delimitación del Problema

### 1.4.1. Delimitación Académica

En este proyecto se integrarán conocimientos adquiridos en las materias: Sensores y Transductores, Redes de Computadoras III, Informática Industrial y Teoría de Control.

### 1.4.2. Delimitación Temporal

La implementación de este proyecto se realizó en un período de seis meses a partir de la aprobación del mismo, dentro del período académico 2020 - 2021.

### 1.4.3. Delimitación Espacial

Por disposiciones institucionales se acostumbra a que el proceso de elaboración del proyecto de tesis sea realizado en las instalaciones de la Universidad Politécnica Salesiana, sede Guayaquil, ubicada en la Avenida Domingo Comín y callejón Chambers. Sin embargo, por motivo de la Emergencia Sanitaria por la pandemia de Coronavirus, este proyecto se desarrolló desde nuestros hogares, para posteriormente ser entregado a la Carrera de Electrónica y Automatización de la sede.

## 1.5. Objetivos

### 1.5.1. Objetivo General

Diseñar e implementar un Banco de Prácticas Didácticas para un Robomaster S1 utilizando programación en lenguaje Python.

### 1.5.2. Objetivos Específicos

- Diseñar e implementar un algoritmo SLAM para localización y mapeo simultáneo utilizando programación en lenguaje Python.
- Diseñar e implementar un algoritmo de control PID de un robot autónomo con programación en lenguaje Python.
- Diseñar un banco de diez prácticas para un laboratorio de Robótica Aplicada utilizando lenguaje Python.

## 1.6. Beneficiarios de la Propuesta

Se considera dentro del grupo de beneficiarios a estudiantes y docentes de los semestres séptimo, octavo y noveno, de la carrera de Electrónica y Automatización de la Universidad Politécnica Salesiana, así como los integrantes del club de robótica que deseen utilizar este proyecto para nuevas implementaciones.

## 1.7. Justificación

Este proyecto de tesis surge ante la necesidad de implementar un laboratorio de robótica aplicada, así como repotenciar conocimientos respecto a programación, incursionando en el lenguaje Python, muy utilizado para el desarrollo de nuevas tecnologías en el ámbito internacional y contemporáneo, refrescando así las habilidades de los estudiantes y elevando su nivel académico, enriqueciendo la malla curricular y generando espacios de competencia que incentive la mejora constante de los mismos.

## 1.8. Variables e Indicadores

El Robot S1 de Robomaster tiene hasta cuarenta y seis componentes programables, seis de ellos aprovechan la inteligencia artificial. Los lenguajes de programación que pueden usarse con el robot son Python y Scratch 3.0, siendo este último una de las mejores alternativas para iniciarse en el mundo del código y los lenguajes de programación. Adicionalmente, los parachoques tienen sensores para detectar los impactos de proyectiles o de los rayos infrarrojos. Los datos generados a partir de los impactos se transmiten en tiempo real, una característica ideal para las competencias.

# 2. MARCO TEÓRICO

## 2.1. Robótica

### 2.1.1. Breve Historia del Robot

La palabra robot viene del vocablo checo “robota” que hace referencia a trabajo como obligación; el término fue acuñado por Karel Capek en una de sus obras de teatro publicada en 1920. La creación de autómatas data del 1300 a. C. con inventos en Egipto, como el reloj de agua o esculturas que emitían sonidos al recibir rayos del sol. Otros inventos como la urraca voladora y un caballo saltarín en China se dieron en el año 500 a. C. y tres siglos después, entre el 200 y el 206 a. C. se encontró un autómata acuático y una orquesta mecánica con monigotes automatizados.

Arquitas de Tarento en la antigua Grecia, construyó en el año 400 a. C. un autómata que funcionaba con agua o vapor, y simulaba el vuelo de un ave. Hacia el 300 a. C. Cresíbio inventa un órgano musical y un reloj que funcionan con flujo de agua. En el 200 a. C. Filón de Bizancio crea un autómata acuático y una catapulta automática. Estos inventos habrían sido los precursores para la creación de autómatas de movimientos repetitivos en la era actual.

En la segunda mitad del siglo primero Herón de Alejandría describiría en su libro “Autómata” diseño de varios inventos de esta clase, entre ellos artificios que realizaban movimientos impulsados por aspas de molino o circuitos primitivos de vapor de agua, se puede observar uno en la Figura 2-1. Otros inventos en la Roma antigua incluirían puertas automatizadas y manipuladores de llamas para eventos que eran espectáculos muy aclamados por las personas en esa época.

*Figura 2-1: Autómata accionado a vapor del Siglo I [4]*

En Oriente también se ven avances notables, como el buda sobre ruedas que se desplazaba solo en el año 335 d. C., las figuras antropomorfas que danzaban y cantaban del año 700 d. C., un mono alcancía que estiraba las manos y exclamaba “limosna”, un gato de madera que cazaba ratones en el 890 d. C., hasta el año 1050 de la era común donde el príncipe Bhoja publica su libro que trata sobre máquinas que realizaban acciones por cuenta propia, y las llamaba yantras.

A Al-Jazari le es atribuido la construcción de relojes de agua y otros autómatas musicales impulsados (también) por flujo de agua, así como máquinas dispensadoras de bebidas y otros autómatas que realizaban tareas importantes tanto para las labores como para el ocio. Son de gran importancia en esta época los relojes, tanto así que se crean dos de los autómatas más reconocidos en estos días, el gallo de Estrasburgo (funcional desde 1352 hasta 1789) y el Papamoscas, construido en el siglo XVI y que se mantiene funcional hasta la actualidad. Del año 1200 hasta el final del siglo se

reconocen al hombre de hierro de Alberto Magno y la cabeza parlante de Roger Bacon como los autómatas más conocidos.

*Figur*a 2-2: Gallo autómata del primer reloj año 1350 *[5]*

Leonardo da Vinci fue otro protagonista en la historia de la robótica, como inventor creó un león mecánico para el rey Luis XII de Francia, un caballero con armadura capaz de ponerse de pie, mover brazos y cabeza, abrir y cerrar la mandíbula, además diseñó una máquina de cálculo, máquina que sentaría las bases de la robótica computacional que dio paso a los inventos que existen hoy en día.

Salomón de Caux hizo su contribución con automatismos inspirados en la jardinería, como fuentes con decoraciones de aves que funcionaban con vapor y movían flujos de agua. Hans Bullmann luego fabricaría una pequeña orquesta de autómatas a inicios del siglo XVI. Luego en 1533 Johann Müller, construyó aves de metal y madera que volaban por sí solas, tan sólo una década antes que John Dee fabricara en Europa su escarabajo de madera que tenía la capacidad de volar.

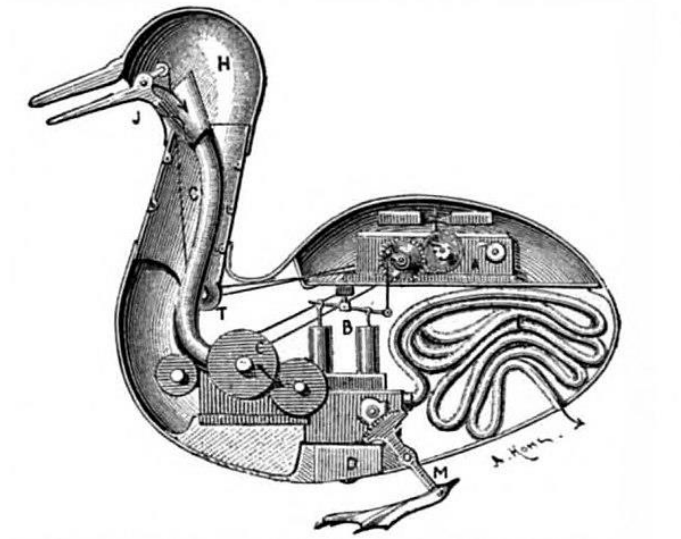
Alrededor del mismo siglo, Juanelo Turriano fabricaba el Hombre de Palo, un autómata con forma humana capaz de caminar y mover la cabeza. El coche tirado por caballos de Salomón de Camus vendría en los próximos años, éste disponía además de una dama que danzaba en el interior del carruaje. A finales de la primera mitad de este siglo el autómata antropomorfo femenino de Descartes fue construido en honor a su hija, y en la segunda mitad es inauguraría el teatro de autómatas en la ciudad de Osaka.

En el siglo XVII se empieza a utilizar el sistema binario para el cómputo y el cálculo, los autómatas tenían estructuras fijas hechas de hierro y partes formes de madera, además constaban de correas de cuero, y platinas de cobre, que en su conjunto formaban un sistema con engranajes, palancas y poleas, que facilitaban los movimientos accionados por fuerza directa.

Desde aquí en adelante vendrían los avances que impulsarían la robótica antigua hacia la contemporánea, con el cambio de percepción de los procesos internos y anatómicos del ser humano y la creación de modelos que comparaban el funcionamiento del cuerpo con la de una estructura mecánica, el desarrollo de algoritmos lógicos de Euler, Marquand y Venn, los modelos matemáticos con operativa mecánica de Goerge Boole son ejemplos del paso a la era moderna de las máquinas que tenían como base la computación.

A partir del año 1700 vendrían autómatas más avanzados, como un androide capaz de interpretar la flauta, moviendo los dedos hacia las diferentes posiciones del instrumento y siguiendo con los ojos las líneas melódicas en partituras musicales. Otro autómata mencionado es un pato mecánico capaz de comer de la mano de quienes lo “alimentaban” y realizar un proceso de digestión completa simulado. Otros autómatas construidos al final de este siglo eran maquetas con paisajes en

movimiento, que incluían humanoides capaces de escribir, dibujar, interpretar instrumentos musicales, proyectos que a veces tomaban años terminar.




**INTERIOR OF VAUCANSON'S AUTOMATIC DUCK.**

*A*, clockwork; *B*, pump; *C*, mill for grinding grain; *F*, intestinal tube; *J*, bill; *H*, head; *M*, feet.

*Figura 2-3: Representación del pato digestivo [6]*

En 1950 Asimov utiliza por primera vez el término "robótica" en su libro de relatos Yo Robot, proponiendo las tres leyes en las que se basa esta materia, y que aún en la actualidad son fundamentos teóricos vigentes, a pesar de no tratarse de una fuente científica. Asimov se refería además a un cuarto enunciado conocido como la ley cero, decía que un robot no debe ser construido para beneficio propio sino buscando el bien común. Las leyes de la robótica de Asimov se detallan en la Tabla 2.1.

*Tabla 2.1: Leyes de la robótica según Asimov*

**Leyes de la robótica según Asimov**

1. **Un robot no debe dañar a un ser humano ni, por su pasividad, dejar que un ser humano sufra daño.**
2. **Un robot debe obedecer las órdenes que le son dadas por un ser humano, excepto cuando estas órdenes están en oposición con la primera Ley.**
3. **Un robot debe proteger su propia existencia, hasta donde esta protección no esté en conflicto con la primera o segunda ley.**

*Fuente: [7]*

En 1938 se revoluciona la industria con la creación del primer manipulador que consistía en un brazo mecánico que manejaba un aspersor de pintura. En 1939 el nacimiento de Elektro (humanoide) y Sparko (perro robot), de la compañía Westinghouse, estos tenían movimientos limitados, pero significaban un gran avance para la tecnología en esa época. Los siguientes inventos se basan en la teoría de autómatas finitos, el álgebra de Boole y los avances de inteligencia artificial de aquellos años.

En 1947 se acuña el término “automatización” por asociados de la compañía de automóviles Ford y empieza un plan para sustituir la mano de obra humana por máquinas que eran capaces de construir automóviles. En 1950 se crea el test de Touring, y se confirma la existencia de problemas computacionales que sólo podrían ser resueltos por humanos afirmando así que una máquina nunca podría llegar a sustituir de manera completa.

En 1952 el MIT (Instituto Tecnológico de Massachusetts) termina de definir y patenta el lenguaje de tipo APT por medio del cual se programaban los robots. Tan solo un año después en Inglaterra se crea el primer autómata móvil de la historia, capaz de seguir una luz sin depender de órdenes de un operador. En 1956 mediante un sensor que consistía en magnetos se consigue retroalimentar los sistemas de manera que las máquinas podían reprogramarse solas, consiguiendo así por primera vez la definición que se conoce de robot. En 1954 se implementó un manipulador mecánico con sistema maestro-esclavo, en el cual un robot reproducía movimientos realizados por un operador, el cual fue de mucha utilidad al momento de manipular desechos radioactivos.

En 1957 se patenta el primer robot industrial controlado por finales de carrera y en 1959 se deja a un lado la lógica booleana para empezar a utilizar una nueva alternativa

que consistía en programas de decisiones binarias para el desarrollo de automatismos combinatorios, junto a esto la llegada del controlador digital modular, precursor de los controladores automáticos actuales.

En 1960 el MIT libera la patente del lenguaje de programación tipo APT para robots y Condec fabrica, a través de una subsidiaria llamada Unimation un robot manipulador automatizado que fue bautizado como Unimate (Figura 2-4), En 1963 en Europa se implementan robots en casi la totalidad de las líneas de ensamblaje de vehículos, a lo que Japón se sumaría cinco años después en 1968.

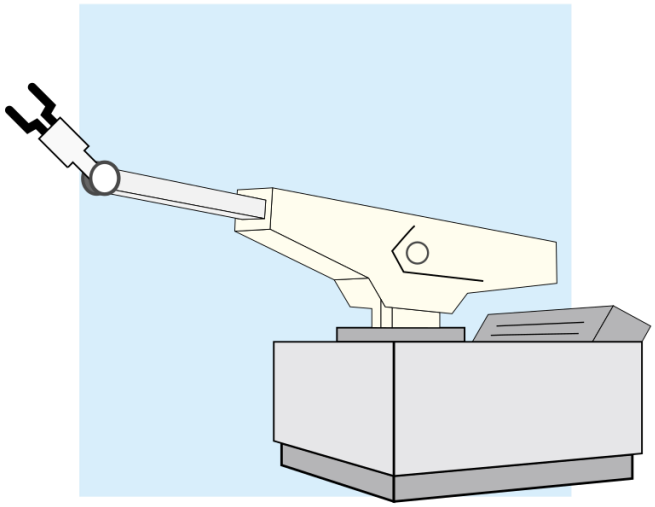


*Figura 2-4: Sketch de un Robot Unimate [8]*

El trabajo publicado en 1962 sobre sensores táctiles en un brazo robótico con 6 GDL marca el inicio de mejoras contundentes en la retroalimentación de los sistemas de control, lo cual brinda a los robots mejor adaptación a procesos. En 1968 el SRI (Instituto de Investigación de Stanford) aplica la visión artificial a través de una cámara colocada en el robot móvil Shakey. El Laboratorio de Inteligencia Artificial del SRI publica el mismo año el primer lenguaje de programación para robots basado en criterios de computación llamado WAVE.

En los siguientes años se crean brazos robóticos en Stanford y Boston, un camión a control remoto con funciones automáticas es entregado al ejército de Estados Unidos por General Electric Co., y el Mars-Rover en la NASA para la exploración del planeta Marte. Para finales de esta década se empieza a emplear el término mecatrónica para todo artefacto inteligente que integre las materias mecánica y electrónica.

En 1970 el SRI implementa el uso de motores eléctricos en los manipuladores automáticos y en la Universidad de Stanford se crea el primer robot seguidor de línea, llamado Stanford Cart. El mismo año Rusia explora la superficie lunar con el Lunokhod, robot provisto de sensores, actuadores y una tarjeta con funciones para telecomunicaciones. Luego en 1971 se descifró el uso de controladores bang-bang para movimientos curvos en brazos robóticos. En 1973 se integran funciones de comunicación de la mano de la empresa Modicon y el desarrollo del protocolo Modbus.

En 1974 el Instituto Tecnológico de Massachusetts trabaja en la aplicación de la ingeniería aeronáutica en la elaboración de piezas de alta precisión y un sistema de realimentación de fuerzas basado en inteligencia artificial; cuatro años después científicos del laboratorio Draper optimizan este sistema mediante técnicas sensoriales para el montaje de piezas.

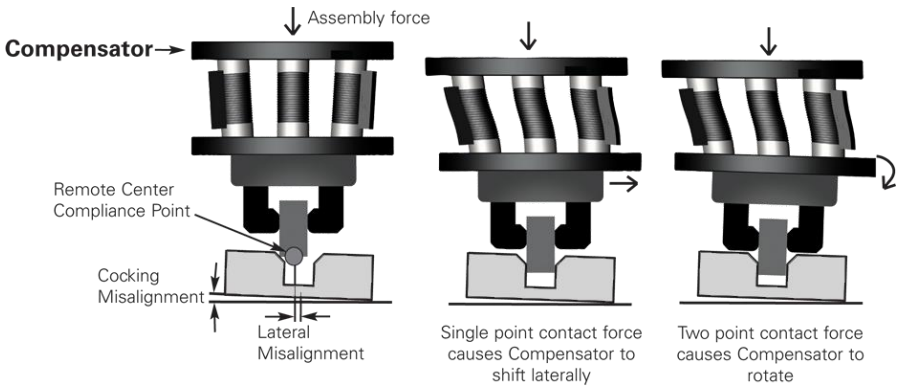



*Figura 2-5: Remote Center Compliance o RCC [9]*

Llegado 1975 los microprocesadores entran en juego, consiguiendo la producción de robots de menor tamaño y costo, provocando un auge en la creación de robots de los cuales se menciona algunos en la Tabla 2.2. Otro evento notable es el montaje de la línea de fabricación flexible en 1983, los robots 8 preprogramados en software gráficos en 1984 y el equipamiento de órganos sensibles en robots en el mismo año.

En 1985 se utilizaron robots para exploración submarina en el rescate de aviones accidentados y recuperación de los restos del Titanic. El mismo año se da el descubrimiento de las Bucky-balls, esferas de carbono nanométricas que impulsaron la implementación de robots a escala miniatura, de gran importancia en la medicina.

En 1989 investigadores del MIT proponen construcción de robots pequeños, económicos y en mayor cantidad, en lugar de grandes y demasiado costosos. Esta mentalidad hace que en solo seis años el parque de robots industriales alcance las 700K unidades. Desde los 90 en adelante la robótica ha tenido grandes avances para uso doméstico, logrando que los androides imiten el andar de una persona, o creando robots cuadrúpedos y artrópodos que imitan a mamíferos e insectos.

El apartado correspondiente a la historia del robot fue consultado de la recopilación redactada en los dos capítulos de las publicaciones citadas en [10] y [7].

*Tabla 2.2: Algunos modelos de robots con su fabricante, año y descripción*

| Año | Modelo | Fabricante | Descripción |
|---|---|---|---|
| **1976** | **Viking** | **NASA** | **Brazo manipulador capaz de tomar muestras de suelo en otros planetas** |
| **1978** | **PUMA** | **Unimation** | **Brazo robótico industrial para montaje de piezas** |
| **1981** | **SCARA** | **Yamanashi University** | **Brazo robótico de 4 GDL con posicionamiento horizontal** |
| **1982** | **RS-1** | **IBM** | **Robot de montaje industrial con tres dispositivos de deslizamiento** |

| | | | |
|---|---|---|---|
| **1989** | **Genghis** | **MIT** | **Robot tipo artrópodo de seis patas** |
| **1994** | **Dante II** | **Carnegie-Mellon University** | **Robot tipo artrópodo de ocho patas** |
| **1996** | **RoboTuna** | **MIT** | **Robot para el estudio del movimiento de especies marinas** |
| **1996** | **Gastrobot** | **N/A** | **Robot autopropulsado por carbono proveniente de materia orgánica** |
| **1997** | **Sojourner** | **NASA** | **Vehículo de seis ruedas controlado desde la Tierra** |
| **1997** | **P-3** | **Honda** | **Androide capaz de imitar la marcha de un ser humano** |
| **1998** | **Furby** | **Tiger-electronics** | **Mascota mecánica que se comunica por lenguaje hablado** |
| **1998** | **Mindstorms** | **LEGO** | **Parte del programa Robotic Invention Systems, permite ensamblaje sencillo** |
| **1998** | **Aíbo** | **Sony** | **Robot mascota canina** |
| | | **JST Corp** | **Robot de fabricante japonés que camina como humano** |
| **2000** | **Asimo** | **Honda** | **Nueva y mejorada versión de robot humanoide de 1.20 m y 43 Kg.** |

*Fuente: (elaborado por los autores)*

### 2.1.1. Robótica Educativa

Como ciencia multidisciplinaria, la robótica representa un pilar fuerte en la educación, incluso desde los niveles más básicos. Incluir esta práctica en el pensum estudiantil incentiva el desarrollo de habilidades creativas, productivas, mejoras en la percepción espacial y lógica del individuo, incentivando el aprendizaje de materias como la física, matemática e informática, y proporcionando un nivel de entendimiento avanzado y orientado a la solución de problemas mediante el uso de medios tecnológicos que tienen una amplia utilidad en la vida profesional [11].

Hoy en día existen plataformas con interfaces muy cómodas para la enseñanza y aprendizaje de la robótica, entre ellas Arduino, Lego Mindstorms y Android. Todas se pueden programar por bloques o bien en su lenguaje clásico, es posible la creación de aplicaciones de teléfono para control de los robots, que gracias a que son tecnologías de código abierto permiten explorar un sinnúmero de posibilidades [12]

### 2.1.2. Clasificación del Robot

De acuerdo al medio en que se desempeñan los robots pueden ser acuáticos, terrestres, aéreos y espaciales.

- Acuáticos

Son robots cuyo sellado debe ser totalmente hermético, y su construcción cambia de acuerdo a la profundidad a la que vayan a trabajar. Son impulsados comúnmente por turbinas y el sistema de dirección se maneja con paletas, aunque algunos realizan movimientos ondulatorios simulando el de algunos animales marinos [13].

- Terrestres

Su construcción debe ser más robusta, su desplazamiento puede darse por deslizamiento, ruedas, patas o cadenas. Cada tipo de desplazamiento es especializado para cierto tipo o tipos de terrenos en los que se vaya a desempeñar el robot [14]. El tipo de robot utilizado en este estudio es un robot terrestre que se desplaza usando cuatro ruedas omnidireccionales de clase mecanum, las cuales serán descritas en un apartado más adelante.

- Aéreos

Son conocidos como vehículos aéreos no tripulados (UAV por sus siglas en inglés), son útiles en la medida que permitan alcanzar lugares donde al ser humano se le dificulte o su presencia represente un riesgo. Han sido utilizados en misiones de exploración, manejo de desechos tóxicos, detección de explosivos, monitoreo de

cultivos o animales silvestres, entre otros. Los últimos avances representativos incluyen visión artificial, por lo que estos vehículos por lo general contienen cámaras con sensores adaptables a la finalidad con la que se construyen dichos robots. Generalmente son impulsados por hélices, pero también es posible encontrar robots insectos que ascienden al agitar sus alas.

- Espaciales

Son robots diseñados con el mayor grado de autonomía y adaptabilidad posible ya que por la distancia de miles de kilómetros la latencia está muy lejos de la deseable para el control óptimo de un manipulador, tienen sensores capaces de captar imágenes y recolectar muestras, para su desplazamiento constan de propulsores que utilizan diferentes tipos de combustibles.

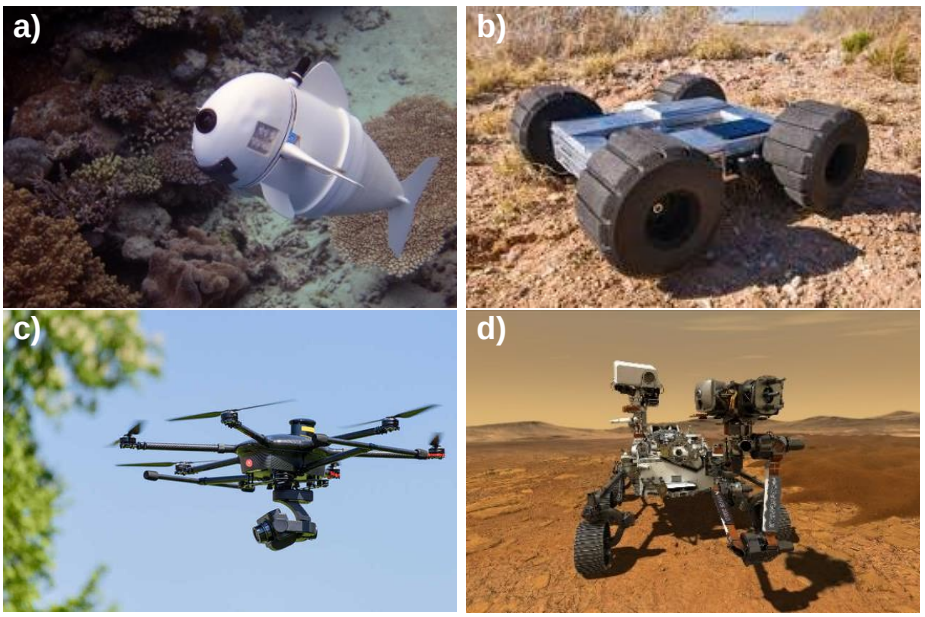



*Figura 2-6: Tipos de robots de acuerdo al medio en que se desempeñan - a) acuático [15], b) terestre [16], aéreo [17], y espacial [18].*

### 2.1.3. Arquitectura de un Robot Móvil

En la Figura 2-7 se observa un diagrama de bloques donde se representa la arquitectura base de un robot móvil, cada nivel será complejo en la medida de la necesidad de cada modelo de robot.


*Figura 2-7: Arquitectura básica de un robot móvil [19]*

El sistema mecánico se refiere al material del que está constituida la estructura física del robot, desde el esqueleto hasta las partes móviles, por lo general es de madera, plástico o metal, aunque en la actualidad muchos robots se desarrollan mediante la impresión 3D de sus piezas, el sistema mecánico debe asegurar formas y dimensiones que ayuden a la motricidad del sistema, por lo que tiene relación directa con los actuadores, y sus piezas deben ser resistentes a cierto nivel de compresión, pandeo, impacto, corrosión, entre otros factores.

El sistema electrónico son una diversidad de tarjetas electrónicas y varios integrados que intervienen en el funcionamiento del robot. Entre sus funciones primordiales se encuentran el procesamiento de datos (microprocesadores, controladores analógicos

y digitales), control de potencia (fuentes de alimentación, contactores, relés, drivers de motores) y transformación de variables recogidas del entorno en señales eléctricas medibles (sensores e instrumentación). Gran parte de estas funciones se pueden encontrar integradas en tarjetas electrónicas como Arduino y Raspberry Pi.

Las fuentes de energía pueden ser de gran variedad, tienen origen físico o químico, y pueden ser neumáticos, hidráulicos, eléctricos, o una combinación de estos dependiendo de sus componentes o necesidades. Con los avances en energías renovables hoy se busca volver los robots cada vez más autónomos y autosustentables, con sistemas que recuperan la energía disipada en forma de calor, o bien que aprovechen la proveniente de la radiación solar.

Por métodos de control se entiende a las técnicas matemáticas que procesan variables realimentadas dentro de algoritmos, para procesar señales y obtener respuestas que vuelvan a las máquinas más adaptables a su entorno, o bien más eficientes en el trabajo que realizan. De esta forma los robots se vuelven capaces de identificar y evitar obstáculos, seguir líneas dentro de una pista, atravesar laberintos, ensamblar artefactos, etc. Los avances tecnológicos en métodos de control tienen la finalidad de emular el pensamiento humano, por lo que se estudian en la rama conocida como inteligencia artificial, dentro de la cual se tratan métodos como las redes neuronales o los algoritmos genéticos.

Los medios de comunicación son los que permiten el ingreso y salida de información al robot, de manera que este puede recibir órdenes o enviar información de sus sensores ya sea a su usuario u otros robots. El medio más popular y conveniente de comunicación para robots móviles es el inalámbrico, se habla de tecnologías como WLAN o Bluetooth, que emplean modulación digital, y siempre va de la mano con

protocolos de encriptación para la codificación de la información entre los que se mencionan WPA, WEP, y otros.

Interfaz humano máquina se refiere a una interfaz gráfica para manipulación del robot, esta debe ser intuitiva y estética, debe facilitar la retroalimentación y acoplarse al gusto y facilidad de quien controla el robot, todo esto es conocido como ser amigable con el usuario. En la actualidad existen un sinnúmero de plataformas virtuales que permiten la creación de aplicaciones, para diversidad de sistemas operativos, mediante las cuales se pueden controlar robots.

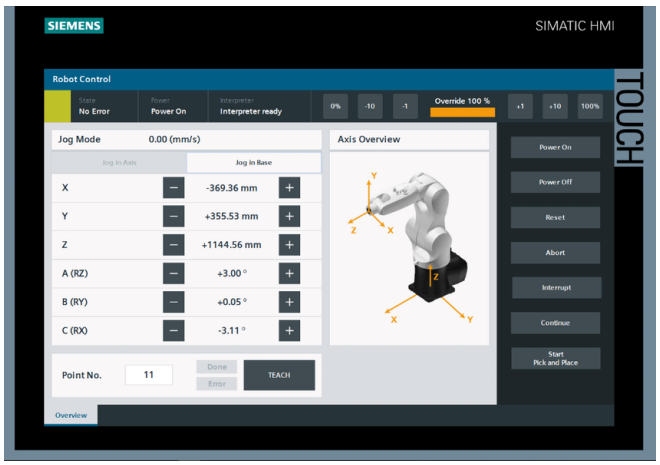



*Figura 2-8: Interfaz de control de robot Kuka en TIA Portal [20]*

### 2.1.4. Sensores e Instrumentos

Por sensores se refiere a todo dispositivo conectado al robot que le permite adquirir información o datos del medio externo o de su estado y funcionamiento interno. Esta información es medible en variables como peso, presión, sonido, tacto, y son transformadas en señales de voltaje y/o corriente que pueden ser interpretadas por otros integrados [21].

Desde el punto de vista de la robótica la clasificación de los sensores se estudia en tres parámetros, por el sitio donde están montados, por la proveniencia de la variable medida, y por su influencia en el medio para la medición [22].

Por la proveniencia de la variable medida los sensores pueden ser internos y externos:

- Sensores Internos o propioceptivos. - Miden variables directamente relacionadas al estado del robot.
- Sensores Externos. - Miden variables provenientes del entorno del robot.

Por el sitio donde están montados se clasifica en sensores locales y globales:

- Sensores Locales. - Son sensores montados en el robot ya sea dentro de él o con salida al entorno.
- Sensores Globales. - Son los que se encuentran en el entorno y envían la información al robot.

Por la influencia en el medio para la medición se estudian sensores activos y pasivos:

- Sensores Activos. - Son sensores que estimulan el medio para su medición, como por ejemplo los sensores láser o sonares.
- Sensores Pasivos. – Son aquellos que toman su medición sin estimular el entorno, como el giroscopio o una cámara digital.

## 2.2. Arduino

Se define como una plataforma electrónica de código abierto que permite leer entradas y traducirlas a salidas que pueden ir desde encendido de luces led hasta el arranque de un motor eléctrico, para lo que se programa mediante una plataforma multisistema operativo, de tipo IDE basada en processing [23]. Desde su creación, ha sido utilizada en infinidad de proyectos, no únicamente por desarrolladores especializados en electrónica, lo cual ha llevado a la producción de una variedad de placas Arduino que se describirán más adelante.

*Tabla 2.3: Tipos de Datos del Lenguaje de Programación de Arduino*

| **Array** | **Char** | **Short** |
|---|---|---|
| Boolean | **Double** | **String** |
| Byte | **Float** | **unsignedChar** |
| long | **Int** | **unsignedInt** |

| | | |
|---|---|---|
| void | **word** | **unsignedLong** |

*Fuente: [23]*

El software Arduino IDE es la plataforma multi sistema operativo donde se codifica la placa electrónica, esta también facilita la compilación de los programas y su subida a la placa; está diseñada para trabajar con todas las placas producidas por Arduino, su código fuente está alojado en los servidores de GitHub que contienen instrucciones detalladas para implementar una infinidad de desarrollos desde cero. Arduino IDE puede utilizarse en Windows, MacOS y Linux, y todos los instaladores son descargables de la página web de Arduino, aunque también es posible, en la misma página, descargar un ejecutable .exe en la que es posible codificar.

### 2.2.1. Modelos de Placas Arduino

Existe una variedad extensa de placas Arduino creadas desde su lanzamiento en el 2006, muchas de ellas ya fueron retiradas del mercado debido a que han sido reemplazadas con placas más sencillas que satisfacen mejor las necesidades de los desarrolladores. Las placas que que se encuentran en el mercado hoy en día van desde el nivel básico hasta aquellas con funcionalidades avanzadas, ya sea en forma de tarjetas electrónicas clásicas, módulos, shields y kits de aprendizaje con proyectos variados para empezar en el mundo de la electrónica. La tabla 2.3 muestra algunos modelos de Arduino de nivel básico con sus principales características.


*Figura 2-9: Sitio de descargas del software IDE [23]*

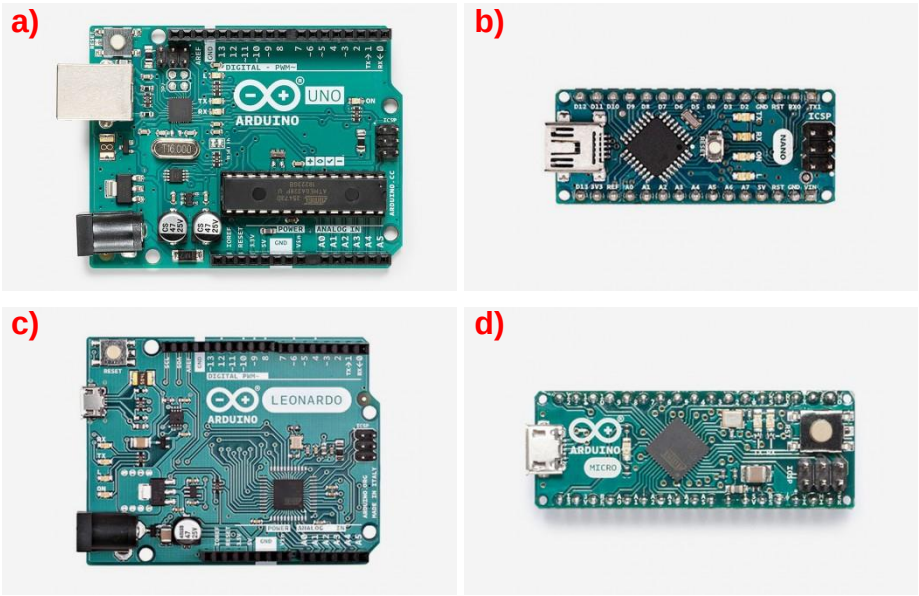



*Figura 2-10: Placas Arduino a) Uno, b) Nano, c) Leonardo y d) Micro [23]*

*Tabla 2.4 Modelos Arduino, con sus principales características*

| Modelo | Voltaje de Sistema | Velocidad de Reloj | Digital Inputs / Outputs | Analog Inputs | PWM | Microcontrolador |
|---|---|---|---|---|---|---|
| Uno | **5V** | **16MHz** | **14** | **6** | **6** | **ATMega328P** |
| Nano | **5V** | **16MHz** | **22** | **8** | **6** | **ATMega328** |
| Leonardo | **5V** | **16MHz** | **12** | **12** | **5** | **ATMega 32u4** |
| Micro | **5V** | **16MHz** | **20** | **12** | **7** | **ATMega 32u4** |

### 2.2.2. Placas Arduino Compatibles

Entendiendo que Arduino es una tarjeta electrónica de código abierto se vuelve posible encontrar los diagramas en su página web oficial, y no únicamente eso, sino utilizarlos para replicar la placa, o crear una con mejoras requeridas para una o varias necesidades específicas. Estas placas creadas a partir de Arduino son llamadas Arduino Compatibles, son fabricadas por terceros y al tener el chip ATMega como integrado principal se vuelve compatible con la totalidad de librerías y hardware relacionado a las placas Arduino originales [24].

Una de las descritas placas Arduino Compatibles es la Pro Micro (Dev-12640) de los fabricantes [25] que en su sitio web describe a la Pro Micro como una Arduino Pro Mini, pero con un ATMega32u4, con built-in USB que permite que funcione como un teclado. Del resto se conoce que tal como las Arduino comunes, funciona a 5V y 16MHz, con 12 Digital I/O, 5 de ellas PWM y un regulador de voltaje de hasta 12VDC.

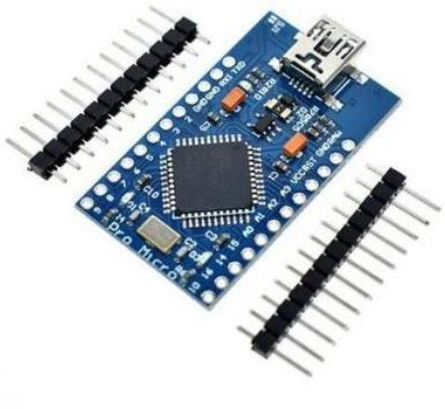


*Figura 2-11: Arduino Compatible - sparkfun Pro Micro 5V [25]*

## 2.3. Raspberry Pi

### 2.3.1. The Raspberry Pi Foundation

Con sede oficial en Reino Unido, la Fundación Raspberry Pi tiene la misión oficial de llevar la tecnología de las computadoras a las manos de todo el mundo: “Esto es para que más personas puedan aprovechar el poder de la informática y las tecnologías digitales para trabajar, para resolver problemas que les importan y para expresarse de manera creativa” [26].

La fundación Rasberry Pi cuenta con clubes y eventos en los que jóvenes pueden organizarse para el intercambio y aprendizaje de habilidades informáticas, de la mano de instituciones educativas y otras organizaciones, les es posible brindar educación en tecnología a cualquiera que lo necesite. Esto mediante el suministro de software de uso libre y computadoras de alto rendimiento a bajo costo.

### 2.3.2. Modelos de Raspberry Pi

Entre los productos destacan una variedad de placas a bajo costo para desarrollos computacionales, algunas computadoras integradas y placas microcontroladoras. A continuación, se presenta la Tabla 2.4 con los modelos más relevantes y sus características

- Raspberry Pi 4 B

La Raspberry Pi 4 Modelo B fue lanzada en el 2019 para reemplazar a la 3 Modelo B, tiene un CPU Broadcom BCM 2711 de 64 bits, con 4 núcleos a 1.5 GHz de velocidad de procesamiento, tiene WLAN y Bluetooth built-in, conexión Ethernet y GPIO extendido de 40 pines, 4 puertos USB 2, salida estéreo y puerto de vídeo compuesto, un puerto HDMI común, puerto de cámara DSI, puerto de pantalla LCD DSI, puerto de micro SD para insertar un disco de arranque y fuente de alimentación conmutada de 2.5 A vía micro USB [27]. La fundación Raspberry Pi se comprometió

a dar soporte y producción a esta tarjeta hasta el año 2029.

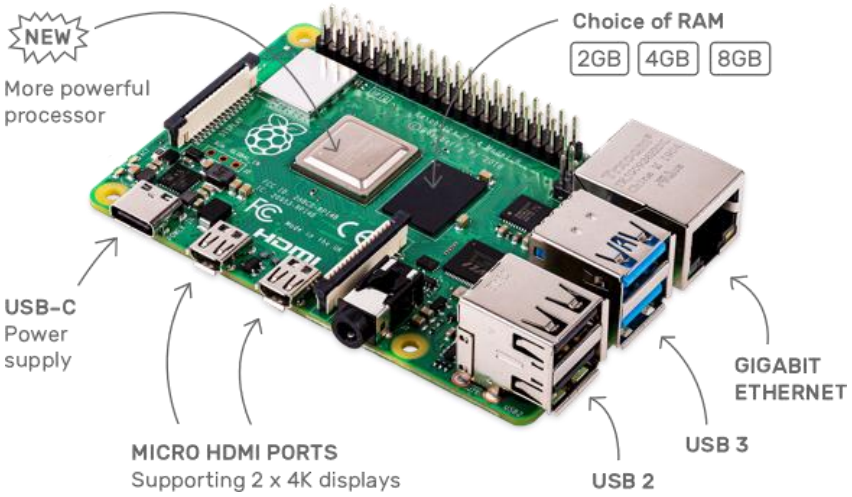



*Figura 2-12: Raspberry Pi 4 Modelo B [27]*

*Tabla 2.5: Tabla comparativa de Modelos Raspberry Pi*

| | Raspberry Pi B | Raspberry Pi 2 | Raspberry Pi 3 | Raspberry Pi 3 B+ | Raspberry Pi 3 A+ | Raspberry Pi 4 B | Raspberry Pi 4 B 8GB |
|---|---|---|---|---|---|---|---|
| **Año** | 2012 | 2015 | 2016 | 2018 | 2018 | 2019 | 2020 |
| **SOC** | Broadcom BCM2835 | Broadcom BCM2836 | Broadcom BCM2837 | Broadcom BCM2837B0 | Broadcom BCM2837B0 | Broadcom BCM2711 | Broadcom BCM2711 |
| **CPU Clock** | 700 MHz | 900 MHz | 1.2 GHz | 1.4 GHz | 1.4 GHz | 1.5 GHz | 1.5 GHz |
| **RAM** | 512 MB | 1 GB | 1 GB DDR2 | 1 GB | 512 MB DDR2 | 1 GB , 2 GB, 4 GB LPDDR4 | 8 GB LPDDR4 |
| **USB** | 2x USB 2.0 | 4x USB2.0 | 4x USB2.0 | 4x USB2.0 | 1x USB2.0 | 2x USB3.0 + 2x USB2.0 + USB-C OTG | 2x USB3.0 + 2x USB2.0 + USB-C OTG |
| **ETH** | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **HDMI** | 1 | 1 | 1 | 1 | 1 | 2 micro | 2 micro |
| **Analog Audio** | 3.5 mm jack | 3.5 mm jack | 3.5 mm jack | 3.5 mm jack | 3.5 mm jack | 3.5 mm jack | 3.5 mm jack |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **LCD Panel** | **Sí** | **Sí** | **Sí** | **Sí** | **Sí** | **Sí** | **Sí** |
| **Cámara** | **Sí** | **Sí** | **Sí** | **Sí** | **Sí** | **Sí** | **Sí** |
| **Serial** | **USB-C** | **USB-C** | **microUSB, GPIO** | **microUSB, GPIO** | **microUSB, GPIO** | **microUSB, GPIO** | **microUSB, GPIO** |
| **Wi-Fi** | **No** | **No** | **802.11n** | **2.4GHz and 5GHz 802.11 b/g/n/ac** | **2.4GHz and 5GHz 802.11 b/g/n/ac** | **2.4GHz and 5GHz 802.11 b/g/n/ac** | **2.4GHz and 5GHz 802.11 b/g/n/ac** |
| **Bluetooth** | **No** | **No** | **4.1 LE** | **4.2 BLE** | **4.2 BLE** | **5.0** | **5.0** |

*Fuente: [28]*

### 2.3.3. Linux OS

Linux es un sistema operativo basado en Unix, ha sido desarrollado para casi todo tipo de computadora existente y su flexibilidad y estabilidad ha permitido el desarrollo de distribuciones de software que brindan varias posibilidades dentro del sistema operativo. Algunas características de este sistema operativo es que permite la ejecución de varias tareas simultáneamente, permite creación de usuarios que comparten recursos asignados por el admin de acuerdo a su jerarquía, brinda soporte a consolas virtuales, reconoce todos los sistemas de archivos estándar, puede conectarse en red, reconoce una amplia variedad de hardware, es open source con un sistema de librerías compartidas que permite que cualquier cualquier ordenador conectado acceda a ellas [29].

La distribución de Linux más importante para nuestro proyecto es Debian, cuya versión elaborada específicamente para Raspberry es conocida como Raspbian. Al ser de distribución libre, estos sistemas operativos se encuentran subidos en las páginas web oficiales de Raspberry Pi, y es posible crear discos de arranque clonando imágenes dentro de memorias micro SD utilizando softwares especializados en este tipo de tareas. De esta forma con sólo insertar la Micro SD en la Raspberry Pi es posible empezar a desarrollar utilizando el sistema operativo [30].

### 2.3.4. Hardware Raspberry Pi

En la página de [26] se encuentra completa la información sobre hardware de las placas y el hardware asociado a Raspberry Pi disponible, a continuación se encuentra una breve descripción de los más importantes, y en anexos se citó el diagrama esquemático de la placa.

- Módulo de Cámara

Existen tres versiones de cámaras lanzadas desde el 2012 por Rasberry Pi, la inicial V1, de 5 megapíxeles, con sensor ov5647 y radio focal 2.9; la segunda versión o V2, tiene 8 megapíxeles, sensor Sony IMX219 y radio focal 2.0; la última versión lanzada en el 2020 es la HQ Camera, de 12.3 megapíxeles, sensor Sony IMX 477, y dependiendo del lente su radio focal es variable. El software de Raspberry permite a estas cámaras funcionar con imágenes en formato JPEG, GIF, BMP, PNG, YUV420 y RGB888, diferentes filtros con efectos y niveles de exposición, balance de blancos automático, entre otras características [26].

- Display Táctil

Se conecta a la Raspberry a través del conector DSI, y dependiendo del soporte de software que se tenga disponible puede permitir además la visualización en una pantalla adicional a través de su puerto HDMI. La instalación física se da por dos

cables de energía negro y rojo, que se conectan a tierra (GND) y a 5V respectivamente, y un bus de datos. [26].

- Mouse y Teclado

La conexión del teclado y mouse se puede dar a través de los puertos USB. Puede ser tanto de cables como inalámbricos, tanto los de tamaño normal como los mini teclados con trackpad integrados.

### 2.3.5. Raspbian OS

Es el sistema operativo más popular legítimo y publicado por la Fundación Raspberry Pi, su forma es bastante similar a la interfaz de Windows y MacOS, con íconos, menús desplegables y el uso de un cursor por lo que si se está familiarizado con cualquiera de estos sistemas operativos utilizar esta plataforma se vuelve muy sencillo [30].


*Figura 2-13: escritorio de raspbian con el menú de inicio desplegado [30]*

## 2.4. Python

Python es un lenguaje simplificado de formato específico basado en texto con un entorno tradicional y flexibilizado, de nivel de dificultad medio a alto, que permite elaboración de programas desde aquellos usados para cálculo hasta juegos de alta complejidad. Python es un intérprete de programación capaz de ejecutar sentencias

conforme estas son escritas, a diferencia de los programas compiladores que necesitan crear un fichero para luego poder traducirlo al lenguaje que entiende la computadora [31].

Python puede ser capaz de recibir sentencias sencillas de escribir, que utilicen palabras o frases fáciles de memorizar, estas palabras o frases formarán parte de aprender un nuevo lenguaje, y por tanto se entiende que tendrán sintaxis y semántica para ser comprendido en su totalidad [32].

La ventana de comandos consiste en dos columnas, a la izquierda se encuentran las frases de Python con las evaluaciones de las mismas, y a la derecha aparecerá la expresión traducida. Algunas frases y comandos de Python se muestran en la Tabla 2.6 con su respectiva sintaxis:

*Tabla 2.6: Algunos comandos de Python*

| Comando | Acción | Sintaxis |
|---|---|---|
| print | **Imprime o muestra en la pantalla** | **>>> print "hola mundo"** |
| x = 3 | **Asignación de variables. Da a una letra el valor que se desee asignar o calcular.** | **>>> x = 3** |
| x == 3 | **Comprobación booleana. Comprueba lo escrito y devuelve true o false** | **>>> x == 3**<br>**True**<br>**>>> x == 6**<br>**False** |
| If then else | **Se evalua una condición, si es verdadera ejecuta la primera instrucción, si es falsa la segunda.** | **if condición-boolena:**<br>**instrucciones 1**<br>**else**<br>**instrucciones 2** |
| while | **Repite varias veces un conjunto de instrucciones siempre que se cumpla una condición.** | **while condición-booleana:**<br>**instrucciones** |

Se debe cuidar mucho la sintaxis en Python ya que una letra faltante o una palabra mal escrita harán que el software no entienda las instrucciones que se dan. Pero los errores al programar en Python no se dan únicamente faltas al escribir (errores de sintaxis), sino también por error en orden de las sentencias (errores lógicos), o error en la claridad de las instrucciones (errores de semántica). Se debe cuidar estos tres puntos ya que Python siempre tomará lo escrito al pie de la letra [31].

*Tabla 2.7: Computadoras comerciales compatibles con ROS*

| Tarjeta | Características |
|---|---|
| Beagleboard  | **Procesador AM 335x 1 Ghz ARM**<br>**RAM 512MB DDR3**<br>**4GB 8-bit memoria flash**<br>**Acelerador de gráficos 3D**<br>**Acelerador NEON floating-point**<br>**Microcontroladores 2x PRU 32-bit** |
| Odroid C2  | **Procesador Amlogic ARM 1.5 GHz**<br>**RAM 2Gb DDR3 SD**<br>**Slot para tarjetas Micro SD**<br>**4 puertos USB 2.0**<br>**Receptor Infrarojo** |
| Raspberry Pi 3  | **Procesador Broadcom BCM2837 64bit 1.2 GHz**<br>**WLAN y Bluetooth on board**<br>**4 puertos USB 2.0**<br>**Puerto Micro SD** |

| | |
|---|---|
| NVIDIA Jetson TK1  | **Procesador ARM Cortex A15**<br>**16 GB almacenamiento**<br>**2 GB RAM**<br>**Lector tarjeta SD**<br>**Puerto USB 2.0**<br>**Puerto USB 3.0**<br>**Puerto HDMI** |

*Fuente: [3]*

## 2.1. Sistema Operativo Robot (ROS)

A pesar de incluir sus siglas el nombre de sistema operativo, ROS es descrito más como un framework para elaboración de softwares para robots, que funciona en varias plataformas, tiene herramientas y librerías que simplifican la tarea de crear robots avanzados y robustos, todo esto en un ambiente donde varios equipos pueden colaborar con sus conocimientos sobre temas en los que se especializan [34].

## 2.2. LASER IMAGING DETECTION AND RAGING (LIDAR)

Es un sistema de medición y detección de objetos por medio de láser, su funcionamiento es emitir rayos de luz láser infrarrojo, y posee de un lente receptor infrarroja, este dispositivo gira 360° cubriendo su entorno. [33]

## 2.3. Héctor SIMULTANEOUS LOCALIZATION AND MAPPING (SLAM)

Es un paquete de librerías que puede ser usado con visión artificial como con sensores láser, y brinda al robot la posibilidad de mapear su entorno en dos dimensiones. Tiene compatibilidad con la mayoría de sensores LiDAR, algunos de ellos detallados en la Tabla 2.8. Hector SLAM trabaja planteando estimaciones de acuerdo a la frecuencia de giro de los sensores que pueden ser desde 10 hasta los 750 Hz, es un controlador que funciona en lazo abierto y a pesar de no ser retroalimentado ha sido utilizado con éxito en la automatización de robots y vehículos terrestres no tripulados [34].

### 2.3.1. Soporte de Hardware

Como ya se mencionó, los paquetes de Hector SLAM funcionan con ROS, que al ser parte de las librerías de Arduino, es compatible con toda tarjeta que utilice los procesadores ATMega y su hardware relacionado, entre ellos los sensores LiDAR, incluido el utilizado para el desarrollo de nuestro proyecto. Los sensores soportados están enlistados en la página oficial de ROS [34], y entre ellos se encuentran para mapeo 1D, Mapeo 2D, mapeo 3D, speech recognition (reconocimiento de voz), cámaras, sensores de tacto, captura de rutas, estimación de ubicación, entre muchos otros. En la Tabla 2.8 se encuentra algunos de los sensores 2D que brindan soporte a los paquetes de mapeo mencionados.

*Tabla 2.8: Sensores 2D compatibles con ROS*

| Sensor | Características |
|---|---|
| **YDLiDAR X4**  | **Rango de 360 grados**<br>**5000 muestras/s**<br>**Alcance: 0,12 – 10 m**<br>**Velocidad máxima: 600 RPM** |
| **RPLIDAR A2M8 360 °**  | **Rango de 360 grados**<br>**4000 muestras/s a 10Hz**<br>**Alcance: 6 m**<br>**Velocidad máxima: 600 RPM** |
| **Xaxxon OpenLIDAR**  | **Rango de 360 grados**<br>**Frecuencia 750Hz**<br>**Alcance: 40 m**<br>**Velocidad máxima: 250 RPM** |

*Fuente: [34]*

## 2.4. DJI Robomaster S1

El S1 es un robot de cuatro ruedas que tiene la apariencia de un tanque, DJI crea el modelo S1 inspirado en sus torneos competitivos, pero con finalidad educativa. El S1 permite una sencilla incursión en el mundo de la programación a través de una serie de tutoriales de proyectos DJI que se encuentran en línea, muchos de ellos en lenguaje Python [35]. El robot funciona con aplicación celular y con control remoto, aunque para el desarrollo de este proyecto se busca controlarlo con caracteres enviados por teclado.

### 2.4.1. Diagrama del S1

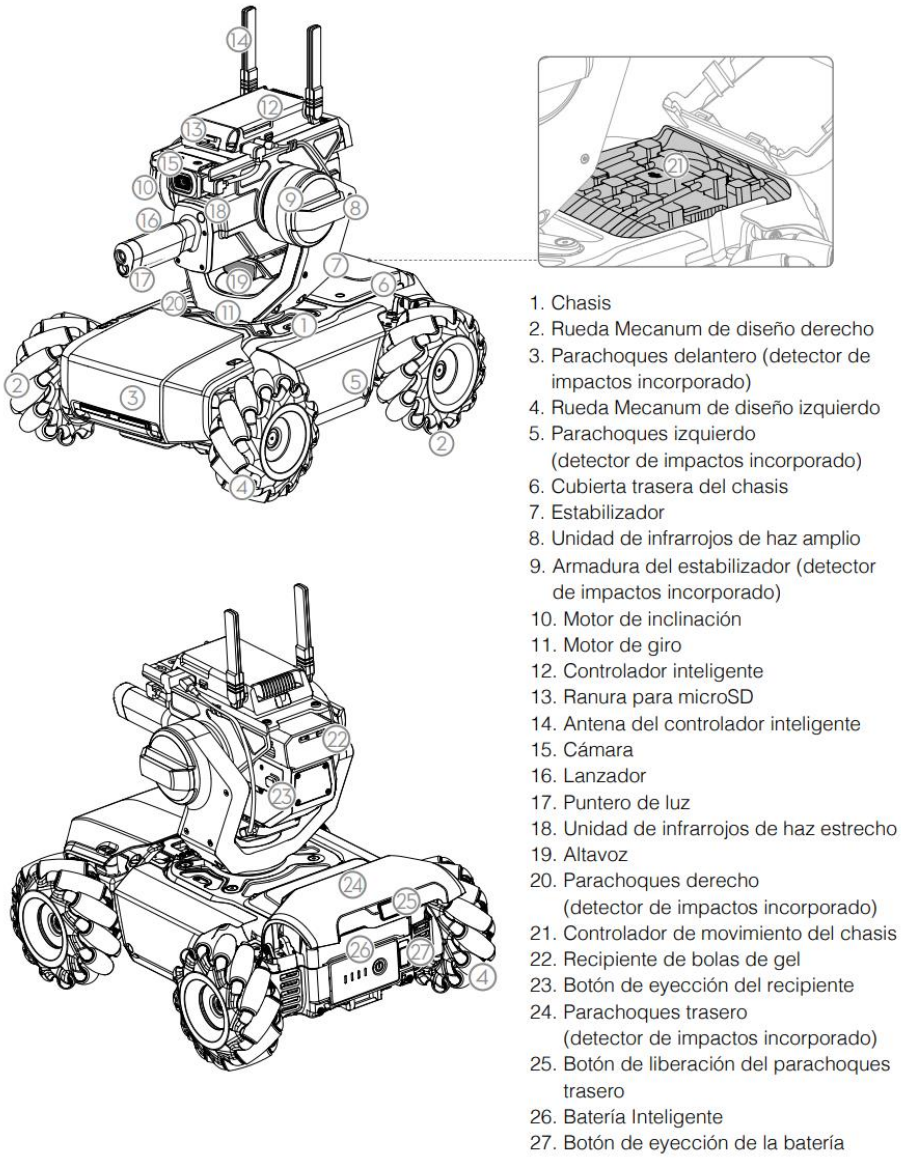



*Figura 2-14: Diagrama del Robot S1 con sus partes [35]*

### 2.4.2. Batería tipo Smart

La batería del S1 posee varias funciones de gestión de energía, entre las que se mencionan la descarga automática cuando su nivel es menor al 70% y ha pasado más de diez días inactivo, para preservar la vida útil del componente; equilibrado del voltaje de las células en cada ciclo de carga; desconexión automática del paso de energía cuando la batería ha sido completamente cargada; interrupción de carga para valores de temperatura mayores a 45º C, cuando hay subidas de corriente o al detectarse un

cortocircuito; alarma en la aplicación por celda en mal estado; el modo suspensión se activa cuando la batería ha sido encendida sin conectar al robot o cuando la carga es inferior al 5%, para evitar que la batería se descargue más allá del límite permitido por el fabricante. La batería tiene una capacidad de 2400 mAh y voltaje de 10.8 V, consta de botón de encendido y cuatro leds que se encienden o apagan consecutivamente dependiendo del nivel de carga de la batería, además de emitir un código traducido a señales que pueden ser consultadas en el manual de usuario del DJI Robomaster S1. Para cargarla se recomienda insertar primero la batería en el módulo de carga y luego conectar este a la toma de corriente.

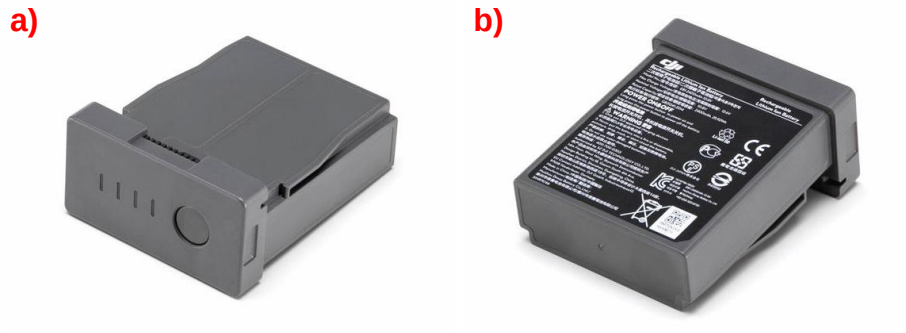



*Figura 2-15: Batería inteligente [35]*

### 2.4.3. Chasis Omnidireccional

La plataforma que permite la movilidad de este robot está basada en las ruedas omnidireccionales mecanum, que permiten movimientos hacia adelante y atrás, hacia los lados, giro, rotación y las posibles combinaciones que serán descritas más adelante.

El controlador se encuentra incorporado en el chasis omnidireccional, permite la combinación de movimientos de ruedas mecanum para permitir al robot movimiento en todas direcciones, además de servir de puente entre el estabilizador, la batería, los parachoques y motores. La configuración de las ruedas mecanum con motores sin

escobillas que alcanzan una velocidad de 1000 rpm, permiten movimiento libre en todas las direcciones, para el funcionamiento del sistema se requieren dos tipos: la rueda mecanum de rosca a la izquierda y la rueda mecanum de rosca a la derecha. Son comúnmente llamadas de diseño izquierdo (Figura 2-16 a) y de diseño derecho (Figura 2-16 b), respectivamente.

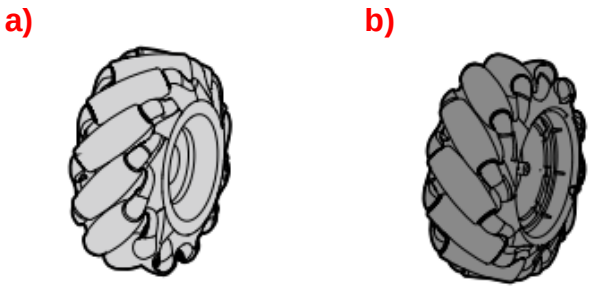



*Figura 2-16: Ruedas mecanum de a) diseño izquierdo y b) diseño derecho [35]*

El movimiento omnidireccional se consigue al colocar un juego de ruedas de cada tipo en las esquinas opuestas, como se puede apreciar en la Figura 2-17. Los movimientos se dan por el combinar de acción de los motores (hacia adelante y hacia atrás) como muestra en la Figura 2-18.

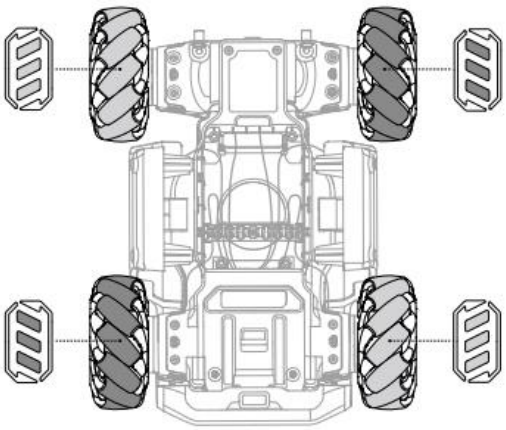


*Figura 2-17: Sistema omnidireccional con ruedas mecanum [35]*

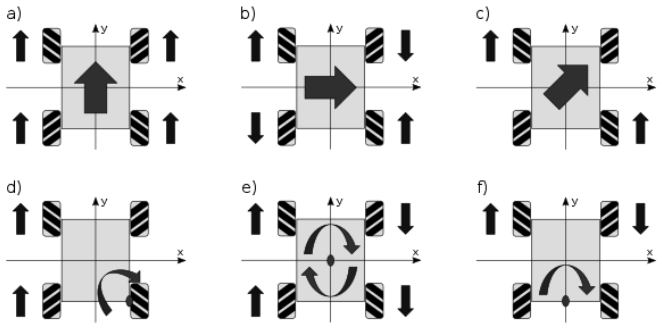



*Figura 2-18: Movimientos y su combinación de motores [36]*

### 2.4.4. Gimbal o Estabilizador

Tiene movimiento en dos ejes, contiene el lanzador y la cámara, y es capaz de permanecer estable sin importar las acciones que se encuentre realizando. Una muestra de esto es que el DJI Robomaster S1 muestra en la cámara un vídeo estable, aun cuando el cañón se encuentra lanzando balas.

Consta de dos motores, el de giro y de inclinación, por medio del controlador funcionan en conjunto para brindar estabilidad al cañón al momento de apuntar y disparar contra un objetivo. Externamente en el gimbal se encuentra la protección, esta a los lados posee cuatro leds a cada lado cuyo color es personalizable. En la Figura 2-19 se aprecian los ángulos de giro en ambos ejes.

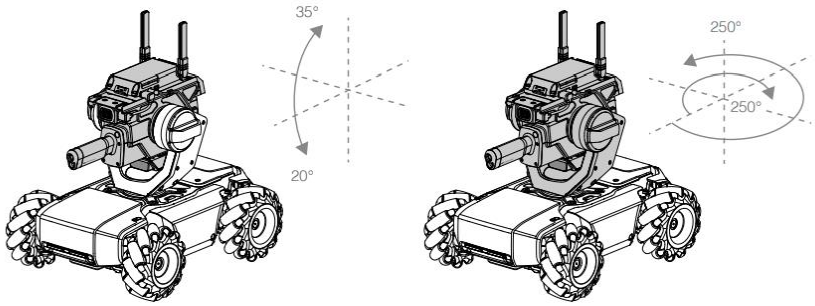



*Figura 2-19: Ejes de movimiento con su ángulo de giro [35]*

### 2.4.5. Controlador del Robomaster

Todos los sistemas del DJI Robomaster S1 están integrados a través del controlador inteligente del robot, permitiendo se añadan hasta seis módulos inteligentes que pueden ser: reconocimiento de línea, reconocimiento de persona, reconocimiento de palmadas, reconocimiento de gestos, y reconocimiento de otros robots S1.

El controlador tiene el puerto de cámara, puerto de altavoz, antenas, puerto bus can, puerto Micro USB, switch de modo de conexión y el botón de conexión.

### 2.4.6. Armamento

El S1 dispone de un cañón ubicado sobre el estabilizador o gimbal, dispone de un láser de 6 m de alcance que tiene una efectividad que varía de 40º a 10º dependiendo de la distancia a la que se encuentre el objetivo, el cañón también puede alimentarse a través de una bandeja que contiene bolas de gel y es capaz de dispararlas a 26 m/s en una frecuencia máxima de 10 proyectiles por segundo.

Para preparar las bolas de gel se mide una tapa de la botella que las contiene en 1000 ml de agua purificada y a temperatura ambiente, durante cuatro horas. La forma de cargar el cañón se explica en el manual de usuario y al usar el cañón del S1 se debe tener mucha precaución de no apuntar a personas o animales.

### 2.4.7. Cámara

Dispone de un sensor de ¼ in y 5 megapíxeles, con un FOV de 120º para una experiencia de juego en primera persona. Se le da mantenimiento con un paño suave y limpiador para lentes, removiendo material ajeno como polvo o impurezas, evitando siempre rayar el vidrio.

### 2.4.8. Altavoz

Se conecta por un Jack de 2.5 mm y tiene potencia nominal de 2 W. El robot S1 tiene una variedad de sonidos activados por la excitación de los diferentes sensores, y brinda una experiencia más real al usuario.

### 2.4.9. Manejo por Aplicación Móvil

- Aplicación Robomaster

La aplicación para control del robot se puede utilizar en cualquier dispositivo inteligente con pantalla táctil, los sistemas operativos soportados son Android, iOS y Windows; además la plataforma permite jugar con otros usuarios, escribir e intercambiar programas y almacenar multimedia que ingresa a los sensores del S1.

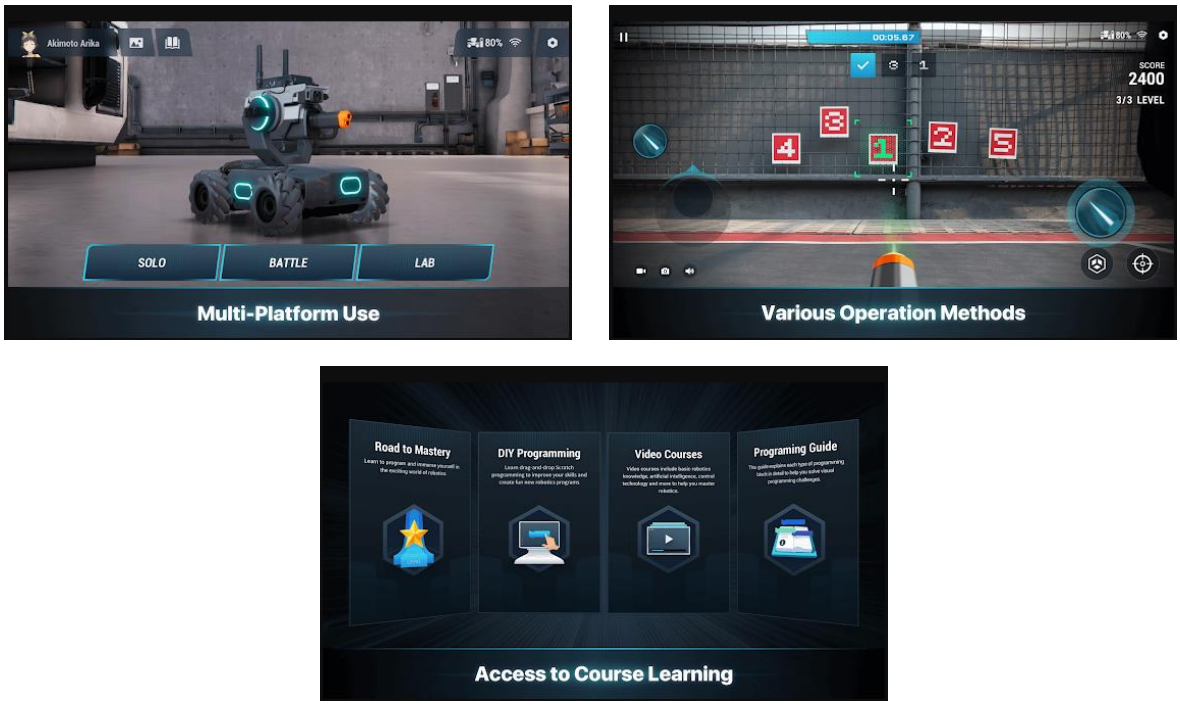



*Figura 2-20: Galería de Imágenes de la aplicación Robomaster [37]*

- Virtualizador Octopus

Esta aplicación se sobrepone a otras aplicaciones Android y virtualiza controles que son vinculados a caracteres de un teclado, mouse, gamepad y una gran variedad de

periféricos de otras marcas. Tiene un diseño amigable que lo hace compatible con la mayor parte de aplicaciones y juegos que existen para Android. Octopus es completamente personalizable, y además se acopla a las funciones nativas de Android como grabación de pantalla, calibración del sensor táctil y limpieza de RAM [37].

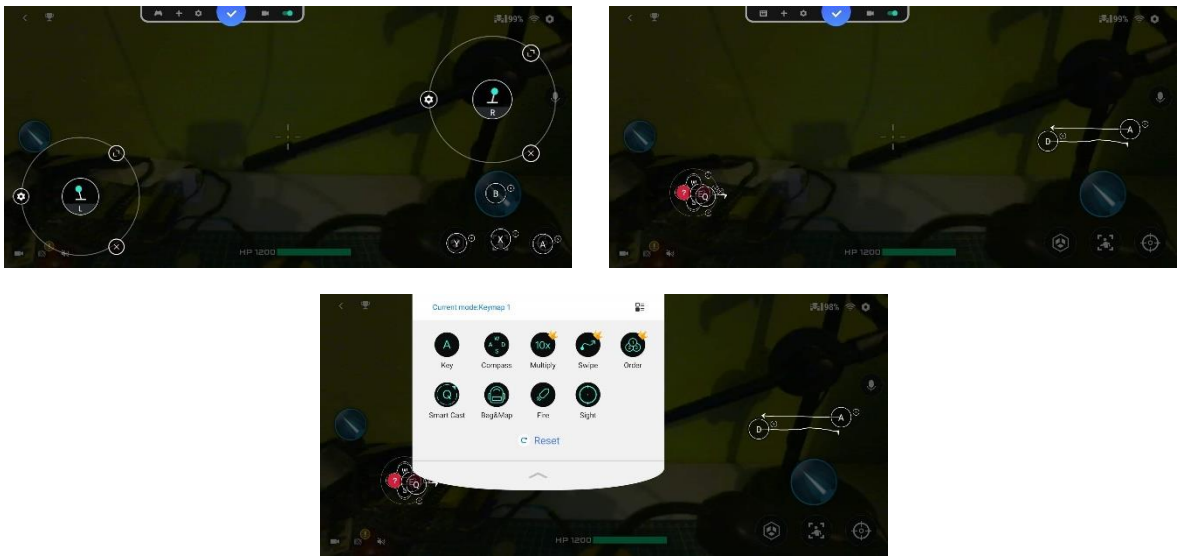


*Figura 2-21: Octopus funcionando superpuesta a la aplicación Robomaster*

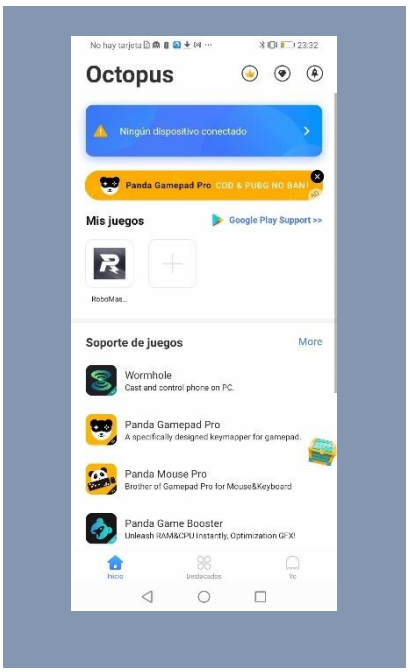



*Figura 2-22: Pantalla principal de Octopus App*

### 2.4.10. Mando a Distancia

El control remoto oficial o gamepad no viene incluido con la compra del robot, pero no es la única alternativa de mando para controlar a distancia al S1. Existen en el mercado opciones para este fin que pueden ser un poco más económicas y aun así guardan compatibilidad con el robot de DJI.


*Figura 2-23: Control remoto Robomaster Gamepad [38]*

*Figura 2-24: Controlador inalámbrico compatible con Robomaster S1 [39]*

### 2.4.11. Indicadores LED

El robot tiene dos juegos de LEDs, uno en el chasis y otro en el estabilizador. Los del estabilizador son ocho luces a cada lado, mientras que los del chasis uno en cada parachoques.

*Tabla 2.9: Descripción de LEDS del Robomaster S1*

| Indicador | Estado | LED del Estabilizador | LED del chasis |
|---|---|---|---|
| Carga de la batería | **Encendido** | **Parpadeo cian en antihorario** | **Cian fijo** |
| | **Apagado** | **Color personalizado se apaga** | **Color personalizado se apaga** |
| Conexión con la aplicación | **S1 no conectado** | **Pulsa en blanco** | **Pulsa en blanco** |
| | **S1 conectando** | **Parpadea Cian** | **Parpadea Cian** |
| | **S1 conectado** | **Color personalizado fijo** | **Color personalizado fijo** |
| Actualización del Firmware | **Actualizando** | **Barras blancas en secuencia** | **Blanco fijo** |
| | **Fallida** | **Rojo fijo** | **Rojo fijo** |
| | **Exitosa** | **Cian fijo** | **Cian fijo** |
| Modo un jugador | **Modo un jugador inicia** | **Color personalizado parpadea en antihorario, luego predeterminado fijo** | **Color personalizado fijo** |
| | **Modo seguimiento inicia** | **Color personalizado parpadea en antihorario** | **Color personalizado fijo** |
| Modo Batalla | **Modo batalla inicia** | **Color personalizado parpadea en antihorario, luego color de equipo fijo** | **Color personalizado fijo** |
| | **Impacto detectado** | **Parpadea rojo una vez** | **Parpadea rojo una vez** |
| | **Derrota** | **Color personalizado parpadea al azar, luego se apaga** | **Color personalizado parpadea, luego se apaga** |
| | **Reactivado** | **Color personalizado parpadea al azar, luego sólido** | **Color personalizado parpadea, luego sólido** |
| | **Victoria** | **Color personalizado fijo** | **Color personalizado fijo** |
| | **Barra de puntos de golpe restaurada** | **Color personalizado parpadea en antihorario, luego fijo** | **Color personalizado parpadea, luego sólido** |
| | **Bonus oculto usado** | **Color personalizdo parpadea tres veces en antihorario** | **Color personalizado fijo** |
| | **Impactado por bonus oculto** | **Color personalizado parpadea mientras bonus siga activo** | **Color personalizado parpadea mientras bonus siga activo** |

*Fuente: [35]*

# 3. MARCO METODOLÓGICO

En los siguientes títulos se hablará sobre el método utilizado, además de los tipos de investigación que serán de ayuda para llevar a cabo este proyecto.

## 3.1. Método Analítico – Sintético

Por medio del método analítico – sintético se realizó desglose del objeto de estudio en sus partes más básicas, para después estudiar al objeto como un todo. Estudiar individualmente cada sección ayuda a especializarse dentro del conocimiento, y la conjunción que se realiza luego ayuda a entender el objeto como un todo y a la vez como la suma de cada parte. Es necesario explorar cómo se relacionan estas partes para saber cómo esta relación influirá en las demás.

Mediante esta modalidad del método científico se avanza hasta el final de este estudio para comprender al Robot Robomaster S1 parte por parte y como un todo.

## 3.2. Investigación Descriptiva y Aplicada

Se utilizó la investigación descriptiva pues se busca sistematizar el proceso por el cual se elaborará un banco de prácticas en la cual intervienen variables tanto cualitativas como cuantitativas, esto quiere decir considerar características medibles y los parámetros bajo los cuales se tomaron dichas medidas. También será útil la investigación aplicada, ya que nuestro objetivo luego de estudiar el robot es producir conocimiento y convertirlo en un producto orientado a satisfacer necesidades de un grupo beneficiario.

## 3.3. Implementación de la propuesta.

El proyecto se conforma de dos secciones, la primera del robot Robomaster S1, y de un controlador Raspberry PI 4 con tarjetas Arduino que realizara el mapeo SLAM por medio de un sensor LiDAR.

- Programación de Raspberry y tarjetas Arduino.
- Mapeo con LiDAR.
- Instalación de Héctor SLAM.
- Programa en Arduino Nano.
- Teleop_Twist_Keyboard y Ros

### 3.4. Arquitectura del Sistema

Para la conexión del sensor YDLiDAR X4 al robot S1 se utilizó una tarjeta Raspberry Pi 4 Modelo B. La Raspberry Pi a su vez proporciona red a una computadora, desde la que se enviarán los caracteres, que luego de ser recibidos por la Raspberry son enviados al Arduino Nano que calcula la lógica para el movimiento de los motores. Estos caracteres son enviados luego a la tarjeta Pro-Micro que los convierte en datos que la aplicación del smartphone convertirá en órdenes, enviadas al robot mediante la vinculación a su red Wi-Fi.

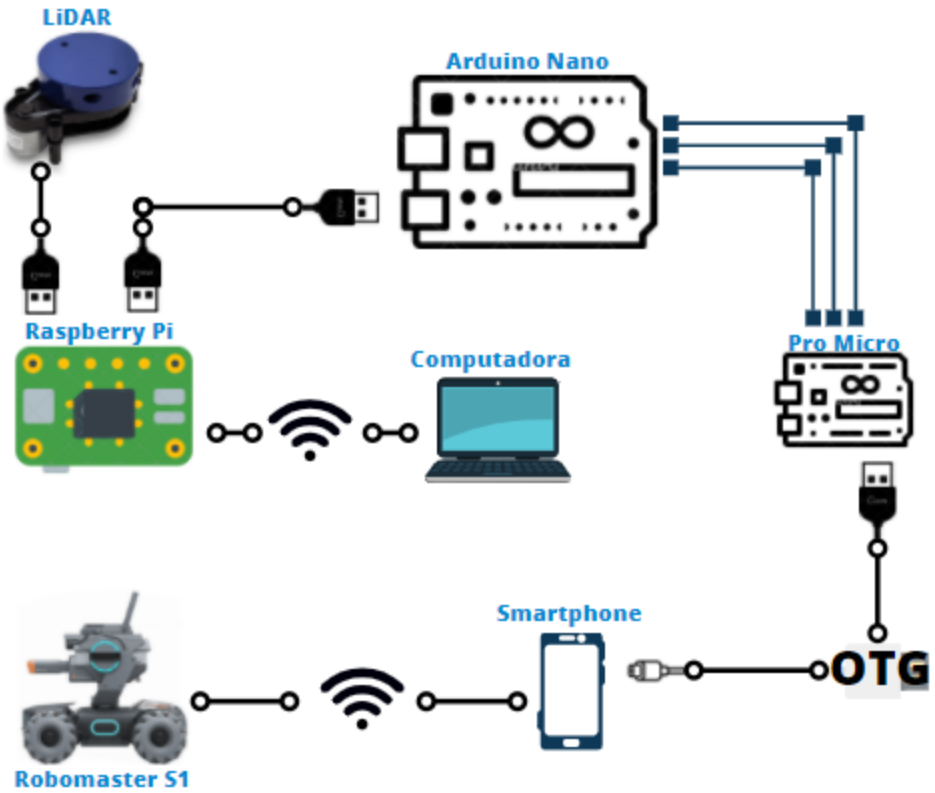



*Figura 3-1: Arquitectura del Sistema*

## 3.5. Programación de Raspberry y tarjetas Arduino

### 3.5.1. Instalando Raspbian en la Raspberry Pi

El software Pi Imager permite instalar el OS (sistema operativo) elegido en la tarjeta microSD previamente insertada y conectada a la computadora a través de un lector, sin embargo, para nuestra implementación se utilizó una versión del OS con desktop que se encontró para descarga en la página de Raspberry, bajo el nombre de *Raspbian Buster with desktop and recommended software*.

Para esto es necesario empezar por la descarga de Pi Imager de la página de Raspberry, seleccionando el que soporte nuestro sistema operativo, según sea Windows, MacOS o Linux. Cabe recalcar que si la tarjeta SD es nueva se puede proceder directamente a la instalación, pero si ha sido usada se debe previamente dar formato utilizando el mismo software Pi Imager,

Luego de instalado el software se procede a abrir y escoger la opción Use Custom, para luego seleccionar la imagen del Raspbian previamente descargada. Al dar click en Write empezará a imprimirse la imagen dentro de la memoria, lo cual puede tomar unos minutos. Al terminar la cuenta nuestra SD será un disco de arranque, que al colocarlo en el slot dentro de la Raspberry podrá iniciar el sistema operativo. Al iniciar pedirá que

### 3.5.2. Habilitando el protocolo SSH

El protocolo Secure Shell (SSH) se encuentra inicialmente inhabilitado y permite controlar la Raspberry desde una red no segura, sea esta una laptop o un smartphone, con únicamente la IP y la contraseña previamente configurada. Para habilitarlo se introduce la línea de código:

```
1. $sudo raspi-config
```

Y entre las opciones que aparecen se ingresa en 5 Interfacing Options. Dentro de este menú seleccionar P2 SSH y luego seleccionar Yes. El siguiente paso es permitir el ingreso con root, para lo que se escribe la siguiente línea de código:

```
1. $sudo nano /etc/ssh/sshd_config
```

En el documento que se abrió debe modificarse la línea donde dice #PermitRootLogin quitando el signo de numeral y escribiendo Yes al final de esta manera:

```
1. PermitRootLogin yes
```

Luego se guarda el archivo y se sale a la ventana de comandos. Entonces se debe consultar la IP de la Raspberry con el comando ifconfig. En nuestro caso la IP de la Raspberry es 192.168.0.4. Se reinicia la Raspberry.

Finalmente se debe instalar PuTTY, un software de código abierto para transferencia de archivos a través de SSH. El software PuTTY funciona además como consola serial y emulador terminal, permitiendo un manejo completo de la Raspberry.

### 3.5.3. Instalación de ROS Melodic

ROS o Sistema Operativo Robot por sus siglas en inglés, es un sistema meta-operado de código abierto que posee un conjunto de herramientas y librerías para facilitar la creación de robots, incluidos los paquetes para mapeo, navegación y localización que serán muy útiles en nuestro proyecto. Las primeras dos líneas de código que se ingresan son para realizar el montaje de los repositorios.

```
1. $sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -
   sc) main" > /etc/apt/sources.list.d/ros-latest.list'
2. $sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key
   C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

Lo siguiente es instalar las herramientas para programación, como son las dependencias del sistema (comando rosdep), diferentes librerías y paquetes necesarios para una correcta compilación (comando rosinstall_generator), además asegurar un workspace para sistemas de control de versiones múltples (comando wstool). Luego se inicializa rosdep, se crea catskin y se hace llamado a los paquetes básicos de funcionamiento conocidos como core packages. Las líneas que se ingresaron para esto fueron las siguientes:

```
1. $sudo apt-get update
2. $sudo apt-get install -y python-rosdep python-rosinstall-generator
   python-wstool python-rosinstall build-essential cmake
3. $sudo rosdep init
4. $rosdep update
5. $mkdir ~/ros_catkin_ws
6. $cd ~/ros_catkin_ws
7. $rosinstall_generator desktop --rosdistro melodic --deps --wet-only --tar >
   melodic-desktop-wet.rosinstall
8. $wstool init -j4 src melodic-desktop-wet.rosinstall
```

Lo siguiente es resolver dependencias y descargar librerías de OGRE para rviz.

```
1. $cd ~
2. $mkdir -p ~/ros_catkin_ws/external_src
3. $cd ~/ros_catkin_ws/external_src
4. $wget https://sourceforge.net/projects/assimp/files/assimp-
   3.1/assimp-3.1.1_no_test_models.zip/download -O assimp-
   3.1.1_no_test_models.zip
5. $unzip assimp-3.1.1_no_test_models.zip
6. $cd assimp-3.1.1/
7. $cmake .
8. $make
9. $sudo make install
10. $sudo apt-get install libogre-1.9-dev
11. $cd ~/ros_catkin_ws/
12. $rosdep install --from-paths src --ignore-src --rosdistro melodic
   -y
```

Ahora se debe escargar paquetes de catkin y cmake que permiten realizar el desarrollo en el workspace elegido. Por último, se reinicia la Raspberry y ya está Ros instalado.

```
1. $sudo ./src/catkin/bin/catkin_make_isolated --install -
   DCMAKE_BUILD_TYPE=Release --install-space /opt/ros/melodic -j2
2. $echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
3. $sudo reboot
```

Una vez iniciado es posible mediante código comprobar la instalación de Ros, así como los paquetes instalados, mediante el ingreso por separado de los dos siguientes códigos:

```
1. $roscore
```

Y luego:

```
1. $rospack list-names
```

Se cierra la sesión de roscore y se procede a crear el workspace para proseguir con nuestros objetivos. El workspace se inicializa mediante las siguientes líneas de código:

```
1. $mkdir -p ~/catkin_ws/src
2. $cd ~/catkin_ws/
3. $catkin_make
4. $echo "source $HOME/catkin_ws/devel/setup.bash" >> ~/.bashrc
```

### 3.5.4. Instalación de Arduino IDE y rosserial

Aprovechando la ventaja que ofrece Arduino al interconectar sensores y otros componentes electrónicos se procede a la instalación del IDE de Arduino.

Primero se revisa la arquitectura del sistema en la Raspberry, para lo que se usó el comando lshw (list hardware), el cual necesita ser instalado mediante esta línea de código:

```
1. $sudo apt-get install lshw
```

Ahora es posible ver la arquitectura del sistema ingresando este comando:

```
1. $lshw
```

El siguiente paso es descargar de la página de Arduino el instalador del IDE, es recomendable la versión portable, pero se debe verificar que esta sea compatible con la arquitectura de la Raspberry, consultada en el paso previo. Una vez instalado IDE Arduino mediante una ventana de comandos de la Raspberry se debe acceder a la carpeta raíz y ejecutar el comando install.sh así:

```
1. $sudo mv arduino-1.8.12 /opt/arduino-1.8.12
2. $cd ~
3. $cd /opt/arduino-1.8.12
4. $sudo ./install.sh
```

Mediante el comando apt-get se actualiza la base de datos de paquetes para confirmar que todos están disponibles, luego con el mismo comando se asegura que la versión de estos paquetes sea la más actual.

```
1. $sudo apt-get update && sudo apt-get upgrade
```

Se utiliza el comando clone para clonar rosserial del repositorio de GitHub:

```
1. $cd catkin_ws/src
2. $git clone https://github.com/ros-drivers/rosserial.git
```

Se generará mensajes necesarios para la comunicación:

```
1. $cd ~/catkin_ws/
2. $catkin_make
```

Se instalará las librerías de catkin en el directorio señalado:

```
1. $catkin_make install
```

Además, la librería de Ros en el entorno de Arduino:

```
1. ls
2. readme.txt ros_lib
```

Para comprobar que se realizaron los pasos correctamente se puede ejecutar el IDE de Arduino y abrir uno de los ejemplos en la pestaña Archivo / Ejemplos / ros_lib. Si no se presentan errores se prosigue a la instalación del IDE para lenguaje Python.

### 3.5.5. Instalación de PyCharm IDE

Descargar la última versión gratuita (community download) de la página de [40], descomprimir el archivo zip y luego ir al editor de código y se tipea las siguientes líneas:

```
1. $sudo mv pycharm-community-2019.3.3 /opt/pycharm-community-2019.3.3
2. $cd ~
3. $cd /opt/pycharm-community-2019.3.3/bin/
4. $sudo ./pycharm.sh
```

Al completar exitosamente la instalación surge una ventana para la configuración inicial del IDE donde podrán crear entradas de escritorio, agregar una contraseña y consultar la edición del software.

### 3.5.6. Configuración de VNC

Virtual Network Computing es un software que permite el acceso a la interfaz gráfica de un servidor a través de un computador que funcione como cliente. De esta manera es posible acceder remotamente a la Raspberry. Para esto se requiere primero instalar el software cliente en la máquina que se está usando, y acceder con las credenciales usuario y contraseña de la Raspberry. Lo primero es seleccionar una resolución de pantalla que corresponda a la máquina local, para lo cual se ingresó el código:

```
1. $sudo raspi-config
```

Y en opciones, se selecciona la resolución deseada.

Se reinicia la Raspberry.

Una vez reiniciada, se ingresa nuevamente el comando:

```
1. $sudo raspi-config
```

En las opciones navegando elegir 5 Interfacing Options / P3 VNC y se habilita el acceso remoto a la Raspberry. Luego se regresa al menú principal.

Se reinicia la Raspberry.

Ahora se procede a instalar el cliente VNC Viewer disponible para descarga de la página de RealVNC [41] de acuerdo al sistema operativo que se maneje.

Una vez instalado VNC Viewer en cliente y servidor, se ingresa en la máquina local al software, se coloca la IP de la Raspberry, se ingresa usuario y contraseña, y ahora es posible manejar la interfaz de la Raspberry en tiempo real de manera remota.

### 3.5.7. Convertir la Raspberry en un AccessPoint

Para funcionar como un router, la Raspberry necesita tener activado el paquete de software de Access point llamado hostapd, además de habilitarlo para que arranque al iniciar. Abrir una ventana de código para ingresar las siguientes líneas:

```
1. sudo apt install hostapd
```

Luego se habilita hostapd:

```
1. sudo systemctl unmask hostapd
2. sudo systemctl enable hostap
```

Se instala dnsmasq para proporcionar servicios de administración de red:

```
1. sudo apt install dnsmasq
```

Se instala paquetes para asegurar que las reglas que se cambiaron no se eliminen al reiniciar:

```
1. sudo DEBIAN_FRONTEND=noninteractive apt install -y netfilter-
   persistent iptables-persistent
```

Accediendo ahora al siguiente archivo es posible configurar la dirección IP de la Raspberry Pi.

```
1. sudo nano /etc/dhcpcd.conf
```

Ir al final del archivo y agregar las siguientes líneas. Luego de escribir guardar y cerrar.

```
1. interface wlan0
2. static ip_address=192.168.4.1/24
3.     nohook wpa_supplicant
```

Ahora se crea un archivo que al ejecutarse permitirá el tráfico de red entre la Raspberry y los dispositivos conectados a ella:

```
1. sudo nano /etc/sysctl.d/routed-ap.conf
```

Agregar el siguiente texto. Nuevamente se cierra guardando cambios.

```
1. net.ipv4.ip_forward=1
```

Escribir una regla de firewall enmascarada para permitir que los dispositivos que se conecten a la Raspberry Pi puedan acceder a la red global:

```
1. sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Guardar las reglas agregadas para que no se eliminen al reiniciar la Raspberry:

```
1. sudo netfilter-persistent sabe
```

Se procede a realizar un respaldo del archivo de configuración DNS original cambiándole el nombre, luego se crea uno con el nombre original de la siguiente forma:

```
1. sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
2. sudo nano /etc/dnsmasq.conf
```

Agregar el siguiente texto al archivo, se cierra guardando cambios.

```
1. interface=wlan0 # Listening interface
2. dhcp-range=192.168.4.2,192.168.4.20,255.255.255.0,24h
3. domain=wlan
4. address=/gw.wlan/192.168.4.1
```

Es poco probable que ocurra que la radio wi-fi de la Raspberry esté bloqueada, pero se debe asegurar su desbloqueo con el siguiente código:

```
1. sudo rfkill unblock wlan
```

Se crea el archivo de hostapd para la configuración del software del Access Point:

```
1. sudo nano /etc/hostapd/hostapd.conf
```

Agregar las siguientes líneas al archivo, luego guardar y cerrar:

```
1. country_code=EC
2. interface=wlan0
3. ssid=ROBOMASTER_RASP
4. hw_mode=g
5. channel=7
6. macaddr_acl=0
7. auth_algs=1
8. ignore_broadcast_ssid=0
9. wpa=2
10. wpa_passphrase=ROBOMASTER_CLAVE
11. wpa_key_mgmt=WPA-PSK
12. wpa_pairwise=TKIP
13. rsn_pairwise=CCMP
```

Ahora es posible reiniciar la Raspberry Pi y se podrá acceder a la red creada con las credenciales elegidas.

### 3.6. Mapeo con LiDAR

Se instalan las librerías de Ros que trabajan con LiDAR.

```
1. cd catkin_ws/src
2. git clone https://github.com/EAIBOT/ydlidar.git
3. cd ../
4. catkin_make
```

```
5. roscd ydlidar/startup // Make the shell script to an executable
6. sudo chmod 777 ./*
7. sudo sh initenv.sh
```

En el workspace de catkin, se ejecuta el siguiente código:

```
1. source devel/setup.bash
```

Y luego:

```
1. catkin_make
```

De esta manera es posible probar el sensor ingresando:

```
1. roslaunch ydlidar lidar_view.launch
```

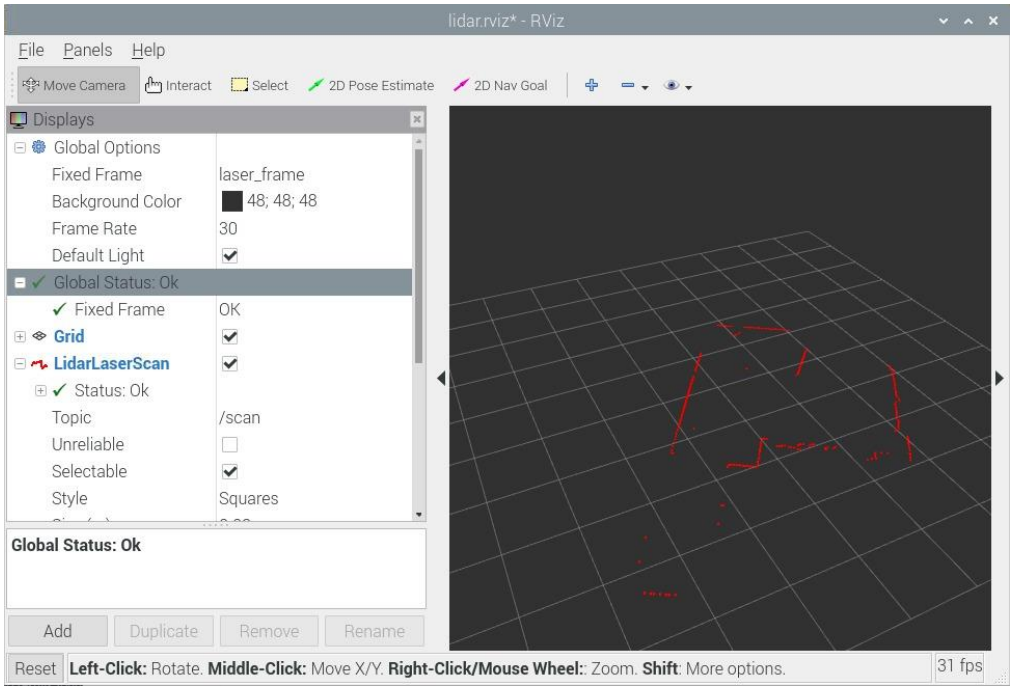



*Figura 3-2: Prueba de funcionamiento del sensor LiDAR*

### 3.7. Instalación de Hector SLAM

Los paquetes para la instalación de Hector SLAM se encuentran en el repositorio de GitHub, para instalarlos se ubica en la carpeta src y se corre el siguiente código:

```
1. git clone https://github.com/tu-darmstadt-ros-pkg/hector_slam.git.
```

Luego se cambia al workspace de Catkin y se procede a clonar el repositorio:

```
1. catkin_make
2. source ~/catkin_ws/devel/setup.bash
```

Para continuar una vez ya instalado Hector SLAM se ejecuta junto con el sensor YDLiDAR en terminales distintas.

```
1. roslaunch ydlidar_ros lidar.launch
```

```
1. roslaunch hector_slam_launch tutorial.launch
```

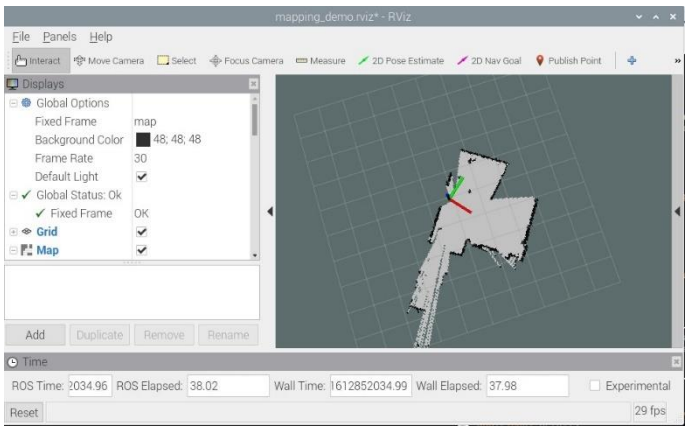



*Figura 3-3: Mapeo en 2D con sensor LiDAR usando Hector SLAM*

### 3.8. Programa en Arduino Nano

Con el código ingresado se pretende manejar los motores. Luego cuando se utilice teleop_twist_keyboard para enviar caracteres al Robomaster S1, este determinará las

condiciones para generar movimientos; la lógica utilizada fue descrita en la tabla 4.1. El código completo de la Arduino Nano será añadido en los anexos al final del trabajo.

*Tabla 3.1: Lógica de movimiento de motores mediante ingreso de caracteres*

| Valor de x lineal | Valor de z angular | Resultado |
|---|---|---|
| x > 0 | **z_rotation == 0** | **Mover adelante** |
| x == 0 | **z_rotation == 1** | **Mover a la derecha** |
| x == 0 | **z_rotation < 0** | **Mover a la izquierda** |
| x < 0 | **z_rotation == 0** | **Mover atrás** |
| x == 0 | **z_rotation == 0** | **Parar** |

Ahora a través de /cmd_vel se enviará valores en porcentajes de 0 a 100% que el controlador del motor entenderá como rango de velocidad de 0 a 255.

### 3.9. Teleop_Twist_Keyboard y Ros

Luego de programar la Arduino Nano se buscará controlar al S1 desde la computadora. Para esto primero se abrió la comunicación entre Arduino y el robot.

```
1. rosrun rosserial_python serial_node.py /dev/ttyACM0
```

Y para enviar caracteres con el Arduino se ingresa el código:

```
1. rosrun teleop_twist_keyboard teleop_twist_keyboard.py
```

En la Figura 4-3 se aprecia el comportamiento del código ingresado previamente, y cómo muestra el arreglo de valores del tema cmd_vel, comprobando así su funcionamiento.

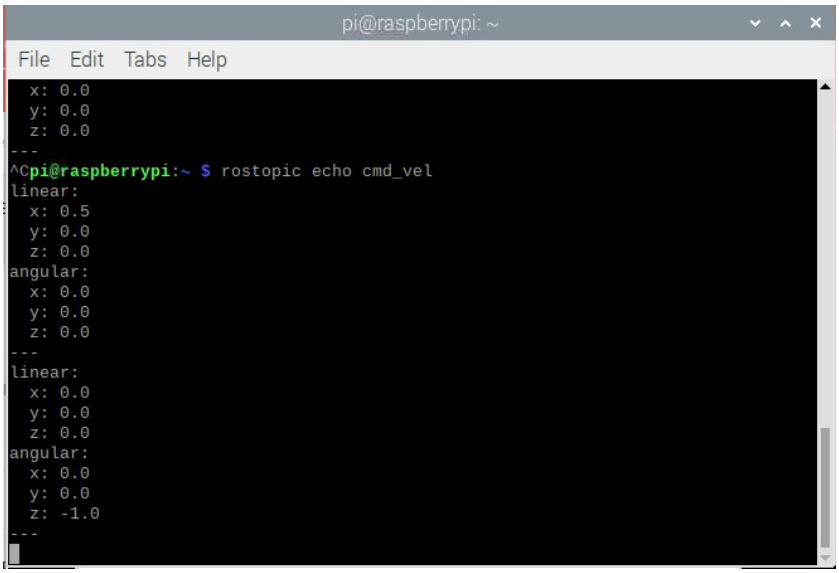



*Figura 3-4: Comprobación de la gestión de cmd_vel*

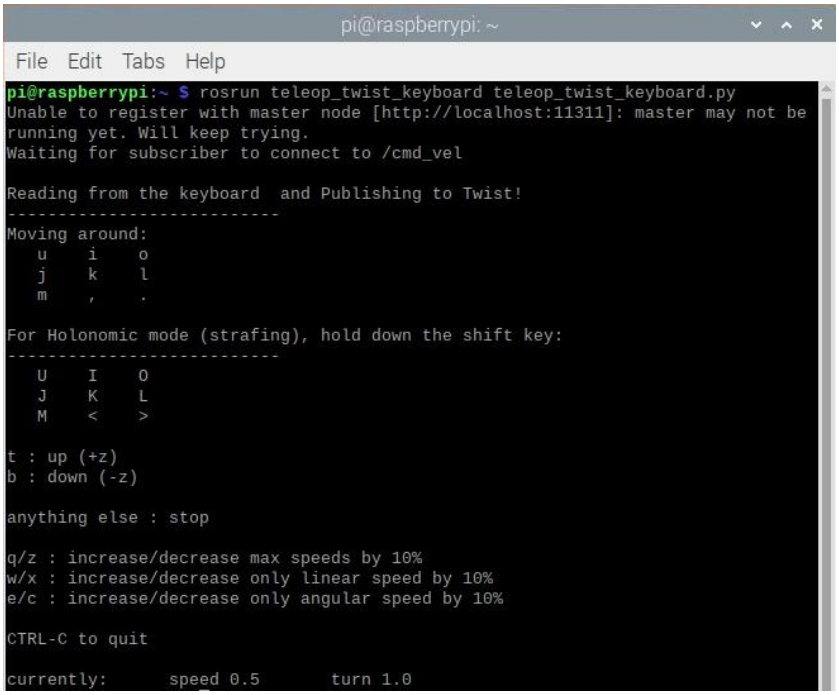



*Figura 3-5: Habilitación de envío de caracteres entre computadora y robot*

Ahora es necesario enviar los datos del tema cmd_vel al puerto serial, donde está conectada nuestra Arduino Nano.

```
1. rosrun rosserial_python serial_node.py /dev/ttyUSB0
```

Con esto ya es posible controlar el robot desde el teclado de una computadora remota.

*Tabla 3.2: Movimientos del robot accionados por teclado*

| Caracter del teclado | Acción del Robot |
|---|---|
| I (letra i) | **Movimiento hacia Adelante** |
| J (letra j) | **Movimiento hacia la Izquierda** |
| L (letra l) | **Movimiento hacia la Derecha** |
| , (signo de puntuación coma) | **Movimiento hacia Atrás** |

# RESULTADOS OBTENIDOS

## 4.1. Propuestas de Prácticas para el Laboratorio de Robótica Aplicada

El presente trabajo de titulación contiene adicional un manual de procedimientos de prácticas, a continuación se muestra una síntesis.

### 4.1.1. Práctica # 1

| | | |
|---|---|---|
| **FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES** | | |
| **CARRERA**: Ingeniería Electrónica y Automatización. | | **ASIGNATURA**: Robótica. |
| **NRO. PRÁCTICA**: | 1 | **TÍTULO PRÁCTICA**: Implementación de una práctica para reconocimiento de marcadores de visión. |

**OBJETIVO**:

- **OBJETIVO GENERAL.**

Implementación de una práctica para reconocimiento de marcadores de visión.

- **OBJETIVOS ESPECÍFICOS:**

- Instalar el software Robomaster de DJI en la computadora a utilizar.
- Vincular el Robomaster S1 mediante la aplicación a la computadora.
- Crear la programación en Python para el reconocimiento de marcadores de visión.

| | |
|---|---|
| **INSTRUCCIONES** | **1.** Descargar e instalar la aplicación de Robomaster de la página web DJI.<br>**2.** Ingresar con su cuenta DJI ingresando correo y contraseña.<br>**3.** Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Descargar e instalar la aplicación de Robomaster de la página web DJI.
2. Ingresar con su cuenta DJI ingresando correo y contraseña. Colocar también el código de verificación en el recuadro para que la aplicación permita el acceso.
3. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
4. Realizar la programación Python para poder reconocer los marcadores visuales.

**RESULTADO(S) OBTENIDO(S)**:

El estudiante puede conocer el modelo de Robot Robomaster S1 con el que se trabajará
El estudiante se familiarizará con el software Robomaster que se controla el robot.
Realizar un vídeo corto del funcionamiento del robot donde se aprecien los objetivos cumplidos.
Informe redactado en el formato de prácticas de laboratorio.

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de programar el Robomaster S1 para el reconocimiento de marcadores visuales.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: MSc. Orlando Barcia Ayala**

***Firma*:** _____

### 4.1.2. Práctica # 2

| | | |
|---|---|---|
| UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR | **FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES** | |
| **CARRERA**: Ingeniería Electrónica y Automatización. | | **ASIGNATURA**: Robótica. |
| **NRO. PRÁCTICA**: | 2 | **TÍTULO PRÁCTICA**: Implementación de un control PID en Robomaster S1. |

**OBJETIVO**:

- **OBJETIVO GENERAL.**

Implementar un sistema de control PID en Robomaster S1.

- **OBJETIVOS ESPECÍFICOS:**

- Instalar el software Robomaster de DJI en la computadora a utilizar.
- Vincular el Robomaster S1 mediante la aplicación a la computadora.
- Configurar el Robomaster para que siga una trayectoria elaborada con cinta de color en una pista.

| | |
|---|---|
| **INSTRUCCIONES** | 1. Descargar e instalar la aplicación de Robomaster de la página web DJI.<br>2. Ingresar con su cuenta DJI ingresando correo y contraseña.<br>3. Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Descargar e instalar la aplicación de Robomaster de la página web DJI.
2. Ingresar con su cuenta DJI ingresando correo y contraseña. Colocar también el código de verificación en el recuadro para que la aplicación permita el acceso.
3. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
4. Realizar la programación Python para la implementación de un control PID.

**RESULTADO(S) OBTENIDO(S)**:

El estudiante puede conocer el modelo de Robot Robomaster S1 con el que se trabajará.
El estudiante se familiarizará con el software Robomaster que se controla el robot.
Realizar un vídeo corto del funcionamiento del robot donde se aprecien los objetivos cumplidos.
Informe redactado en el formato de prácticas de laboratorio.

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de programar el Robomaster S1 para que funcione como seguidor de línea utilizando un control PID.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: __MSc. Orlando Barcia Ayala____**

***Firma*: _____**

### 4.1.3. Práctica # 3

| | | |
|---|---|---|
| UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR | **FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES** | |
| **CARRERA**: Ingeniería Electrónica y Automatización. | | **ASIGNATURA**: Robótica. |
| **NRO. PRÁCTICA**: | 3 | **TÍTULO PRÁCTICA**: Localización y mapeo simultáneo (SLAM). |

**OBJETIVO**:

**- OBJETIVO GENERAL.**

Implementación de una práctica utilizando algoritmo SLAM para localización y mapeo simultáneo.

**- OBJETIVOS ESPECÍFICOS:**

**-** Configurar la red de la computadora para la vinculación con el robot.
- Configurar el servidor VNC para manejar la RaspBerry de manera remota.
- Controlar el robot mediante el teclado con teleop_twist_keyboard.

| | |
|---|---|
| **INSTRUCCIONES** | 1. Descargar e instalar la aplicación de Robomaster de la página web DJI.<br>2. Ingresar con su cuenta DJI ingresando correo y contraseña.<br>3. Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Cambiar configuración del adaptador.
2. Conectarse a la Raspberry mediante servidor VNC.
3. Inicialización de ROS.
4. Prueba de arranque y captura de datos del sensor LiDAR.
5. Habilitar comunicación entre Robomaster S1 y Arduino.
6. Habilitar el envío de caracteres.
7. Mapeo mediante Hector SLAM.
8. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
9. Realizar la programación Python para la localización y mapeo simultáneo SLAM.

**RESULTADO(S) OBTENIDO(S)**:

Identificación de puntos en el entorno referenciados al mapeo de Hector SLAM.

Informe redactado en el formato de prácticas de laboratorio.

El estudiante se familiarizará con el entorno de VNC, ROS, LINUX

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de realizar el mapeo en 2D utilizando el Robomaster S1 y los paquetes de Hector SLAM.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: __MSc. Orlando Barcia Ayala____**

***Firma*: _____**

### 4.1.4. Práctica # 4

| | | |
|---|---|---|
| UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR | **FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES** | |
| **CARRERA**: Ingeniería Electrónica y Automatización. | | **ASIGNATURA**: Robótica. |
| **NRO. PRÁCTICA**: | 4 | **TÍTULO PRÁCTICA**: Identificar y rastrear una persona. |

**OBJETIVO**:

- **OBJETIVO GENERAL.**

Generar un código en el Robomaster S1 para el reconocimiento y seguimiento de una persona.

- **OBJETIVOS ESPECÍFICOS:**

- Instalar el software Robomaster de DJI en la computadora a utilizar.
- Vincular el Robomaster S1 mediante la aplicación a la computadora.
- Crear los programas y subirlos al Robomaster S1.

| | |
|---|---|
| **INSTRUCCIONES** | 1. Descargar e instalar la aplicación de Robomaster de la página web DJI.<br>2. Ingresar con su cuenta DJI ingresando correo y contraseña.<br>3. Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Descargar e instalar la aplicación de Robomaster de la página web DJI:
2. Ingresar con su cuenta DJI ingresando correo y contraseña. Colocar también el código de verificación en el recuadro para que la aplicación permita el acceso.
3. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
4. Realizar la programación Python para identificar y rastrear una persona.

**RESULTADO(S) OBTENIDO(S)**:

El estudiante puede conocer el modelo de Robot Robomaster S1 con el que se trabajará
El estudiante se familiarizará con el software Robomaster que se controla el robot.
Realizar un vídeo corto del funcionamiento del robot donde se aprecien los objetivos cumplidos.
Informe redactado en el formato de prácticas de laboratorio.

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de programar el Robomaster S1 para identificación y seguimiento de personas.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: __MSc. Orlando Barcia Ayala____**

***Firma*: _____**

### 4.1.5. Práctica # 5

| | | | |
|---|---|---|---|
| UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR | **FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES** | | |
| **CARRERA**: Ingeniería Electrónica y Automatización. | | **ASIGNATURA**: Robótica. | |
| **NRO. PRÁCTICA**: | 5 | **TÍTULO PRÁCTICA**: Respuestas únicas al reconocimiento de aplausos. | |

**OBJETIVO**:

- **OBJETIVO GENERAL.**

Generar un código en el Robomaster S1 mediante dos o tres aplausos.

- **OBJETIVOS ESPECÍFICOS:**

- Instalar el software Robomaster de DJI en la computadora a utilizar.
- Vincular el Robomaster S1 mediante la aplicación a la computadora.
- Crear los programas y subirlos al Robomaster S1.

| | |
|---|---|
| **INSTRUCCIONES** | 1. Descargar e instalar la aplicación de Robomaster de la página web DJI.<br>2. Ingresar con su cuenta DJI ingresando correo y contraseña.<br>3. Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Descargar e instalar la aplicación de Robomaster de la página web DJI:
2. Ingresar con su cuenta DJI ingresando correo y contraseña. Colocar también el código de verificación en el recuadro para que la aplicación permita el acceso.
3. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
4. Realizar la programación Python para respuestas únicas al reconocimiento de aplausos.

**RESULTADO(S) OBTENIDO(S)**:

El estudiante puede conocer el modelo de Robot Robomaster S1 con el que se trabajará
El estudiante se familiarizará con el software Robomaster que se controla el robot.
Realizar un vídeo corto del funcionamiento del robot donde se aprecien los objetivos cumplidos.
Informe redactado en el formato de prácticas de laboratorio.

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de programar el Robomaster S1 para respuestas únicas al reconocimiento de aplausos.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: __MSc. Orlando Barcia Ayala____**

***Firma*: _____**

### 4.1.6. Práctica # 6

| | | |
|---|---|---|
| UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR | **FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES** | |
| **CARRERA**: Ingeniería Electrónica y Automatización. | | **ASIGNATURA**: Robótica. |
| **NRO. PRÁCTICA**: | 6 | **TÍTULO PRÁCTICA**: Respuestas autodefinidas al reconocimiento de gestos físicos de una persona. |

**OBJETIVO**:

**- OBJETIVO GENERAL.**

Generar un código en el Robomaster S1 para respuestas autodefinidas al reconocimiento de gestos físicos de una persona.

**- OBJETIVOS ESPECÍFICOS:**

- Instalar el software Robomaster de DJI en la computadora a utilizar.
- Vincular el Robomaster S1 mediante la aplicación a la computadora.
- Crear los programas y subirlos al Robomaster S1.

| | |
|---|---|
| **INSTRUCCIONES** | 1. Descargar e instalar la aplicación de Robomaster de la página web DJI.<br>2. Ingresar con su cuenta DJI ingresando correo y contraseña.<br>3. Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Descargar e instalar la aplicación de Robomaster de la página web DJI:
2. Ingresar con su cuenta DJI ingresando correo y contraseña. Colocar también el código de verificación en el recuadro para que la aplicación permita el acceso.
3. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
4. Realizar la programación Python para respuestas autodefinidas al reconocimiento de gestos físicos de una persona.

**RESULTADO(S) OBTENIDO(S)**:

El estudiante puede conocer el modelo de Robot Robomaster S1 con el que se trabajará
El estudiante se familiarizará con el software Robomaster que se controla el robot.
Realizar un vídeo corto del funcionamiento del robot donde se aprecien los objetivos cumplidos.
Informe redactado en el formato de prácticas de laboratorio.

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de programar el Robomaster S1 para respuestas autodefinidas al reconocimiento de gestos físicos de una persona.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: __MSc. Orlando Barcia Ayala____**

***Firma*: _____**

### 4.1.7. Práctica # 7

| | | |
|---|---|---|
| UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR | **FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES** | |
| **CARRERA**: Ingeniería Electrónica y Automatización. | | **ASIGNATURA**: Robótica. |
| **NRO. PRÁCTICA**: | 7 | **TÍTULO PRÁCTICA**: Robot seguidor de línea. |

**OBJETIVO**:

- **OBJETIVO GENERAL.**

Generar un código en el Robomaster S1 para un robot seguidor de línea.

- **OBJETIVOS ESPECÍFICOS:**

- Instalar el software Robomaster de DJI en la computadora a utilizar.
- Vincular el Robomaster S1 mediante la aplicación a la computadora.
- Crear los programas y subirlos al Robomaster S1.

| | |
|---|---|
| **INSTRUCCIONES** | 1. Descargar e instalar la aplicación de Robomaster de la página web DJI.<br>2. Ingresar con su cuenta DJI ingresando correo y contraseña.<br>3. Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Descargar e instalar la aplicación de Robomaster de la página web DJI:
2. Ingresar con su cuenta DJI ingresando correo y contraseña. Colocar también el código de verificación en el recuadro para que la aplicación permita el acceso.
3. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
4. Realizar la programación Python para un robot seguidor de línea.

**RESULTADO(S) OBTENIDO(S)**:

El estudiante puede conocer el modelo de Robot Robomaster S1 con el que se trabajará
El estudiante se familiarizará con el software Robomaster que se controla el robot.
Realizar un vídeo corto del funcionamiento del robot donde se aprecien los objetivos cumplidos.
Informe redactado en el formato de prácticas de laboratorio.

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de programar el Robomaster S1 para un robot seguidor de línea.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: __MSc. Orlando Barcia Ayala____**

***Firma*: _____**

### 4.1.8. Práctica # 8

| | | |
|---|---|---|
| UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR | **FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES** | |
| **CARRERA**: Ingeniería Electrónica y Automatización | | **ASIGNATURA**: Robótica. |
| **NRO. PRÁCTICA**: | 8 | **TÍTULO PRÁCTICA**: Implementar una práctica para un robot explorador radio controlado. |

**OBJETIVO**:

- **OBJETIVO GENERAL.**

Implementar una práctica para un robot explorador radio controlado.

- **OBJETIVOS ESPECÍFICOS:**

- Instalar el software Robomaster de DJI en el dispositivo móvil a utilizar.
- Vincular el Robomaster S1 mediante la aplicación del dispositivo móvil.
- Crear la programación en Python para un robot explorador radio controlado.

| | |
|---|---|
| **INSTRUCCIONES** | 1. Descargar e instalar la aplicación de Robomaster de la página web DJI:<br>2. Ingresar con su cuenta DJI ingresando correo y contraseña.<br>3. Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Descargar e instalar la aplicación de Robomaster de la página web DJI:
2. Ingresar con su cuenta DJI ingresando correo y contraseña. Colocar también el código de verificación en el recuadro para que la aplicación permita el acceso.
3. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
4. Realizar la programación Python para un robot explorador radio controlado.

**RESULTADO(S) OBTENIDO(S)**:

El estudiante puede conocer el modelo de Robot Robomaster S1 con el que se trabajará
El estudiante se familiarizará con el software Robomaster que se controla el robot.
Realizar un vídeo corto del funcionamiento del robot donde se aprecien los objetivos cumplidos.
Informe redactado en el formato de prácticas de laboratorio.

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de programar el Robomaster S1 para un robot explorador radio controlado.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: __MSc. Orlando Barcia Ayala____**

***Firma*: _____**

### 4.1.9. Práctica # 9

**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES**

| | | |
|---|---|---|
| **CARRERA**: Ingeniería Electrónica y Automatización. | | **ASIGNATURA**: Robótica. |
| **NRO. PRÁCTICA**: | 9 | **TÍTULO PRÁCTICA**: Implementar una práctica para un robot de batalla utilizando el cañón de disparo incorporado con objetivo fijo. |

**OBJETIVO**:

- **OBJETIVO GENERAL.**

Implementar una práctica para un robot de batalla utilizando el cañón de disparo incorporado con objetivo fijo.

- **OBJETIVOS ESPECÍFICOS:**

- Instalar el software Robomaster de DJI en el dispositivo móvil Android a utilizar.
- Vincular el Robomaster S1 mediante la aplicación al dispositivo móvil Android.
- Crear la programación en Python para un robot de batalla utilizando el cañón de disparo incorporado con objetivo fijo.

| | |
|---|---|
| **INSTRUCCIONES** | 1. Descargar e instalar la aplicación de Robomaster de la página web DJI.<br>2. Ingresar con su cuenta DJI ingresando correo y contraseña.<br>3. Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Descargar e instalar la aplicación de Robomaster de la página web DJI.
2. Ingresar con su cuenta DJI ingresando correo y contraseña. Colocar también el código de verificación en el recuadro para que la aplicación permita el acceso.
3. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
4. Realizar la programación Python para un robot de batalla utilizando el cañón de disparo incorporado con objetivo fijo.

**RESULTADO(S) OBTENIDO(S)**:

El estudiante puede conocer el modelo de Robot Robomaster S1 con el que se trabajará
El estudiante se familiarizará con el software Robomaster que se controla el robot.
Realizar un vídeo corto del funcionamiento del robot donde se aprecien los objetivos cumplidos.
Informe redactado en el formato de prácticas de laboratorio.

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de programar el Robomaster S1 para un robot de batalla utilizando el cañón de disparo incorporado con objetivo fijo.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: __MSc. Orlando Barcia Ayala____**

***Firma*: _____**

### 4.1.10. Práctica # 10

| | | |
|---|---|---|
| UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR | **FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES** | |
| **CARRERA**: Ingeniería Electrónica y Automatización. | | **ASIGNATURA**: Robótica. |
| **NRO. PRÁCTICA**: | 10 | **TÍTULO PRÁCTICA**: Implementar una práctica incorporando la previa descrita de un robot radio controlado con el uso de cañón de disparo, detector de impacto, incorporando el estabilizador. |

**OBJETIVO**:

- **OBJETIVO GENERAL.**

- Implementar una práctica incorporando la previa descrita de un robot radio controlado con el uso de cañón de disparo, detector de impacto, incorporando el estabilizador

- **OBJETIVOS ESPECÍFICOS:**

- Instalar el software Robomaster de DJI en el dispositivo móvil Android a utilizar
- Vincular el Robomaster S1 mediante la aplicación al dispositivo móvil Android.
- Crear la programación en Python para un robot radio controlado con el uso del cañón de disparo, detector de impacto incorporando el estabilizador.

| | |
|---|---|
| **INSTRUCCIONES** | 1. Descargar e instalar la aplicación de Robomaster de la página web DJI.<br>2. Ingresar con su cuenta DJI ingresando correo y contraseña.<br>3. Conectar el robot con el programa Robomaster.<br>4. Realizar la programación Python. |

**ACTIVIDADES POR DESARROLLAR**

1. Descargar e instalar la aplicación de Robomaster de la página web DJI.
2. Ingresar con su cuenta DJI ingresando correo y contraseña. Colocar también el código de verificación en el recuadro para que la aplicación permita el acceso.
3. Conectar el robot dando clic en el botón ubicado en la parte superior derecha.
4. Realizar la programación Python para un robot radio controlado con el uso de cañón de disparo, detector de impacto, incorporando el estabilizador.

**RESULTADO(S) OBTENIDO(S)**:

El estudiante puede conocer el modelo de Robot Robomaster S1 con el que se trabajará
El estudiante se familiarizará con el software Robomaster que se controla el robot.
Realizar un vídeo corto del funcionamiento del robot donde se aprecien los objetivos cumplidos.
Informe redactado en el formato de prácticas de laboratorio.

**CONCLUSIONES**:

Los estudiantes estarán en capacidad de programar el Robomaster S1 para un robot radio controlado con el uso de cañón de disparo, detector de impacto, incorporando el estabilizador.

**RECOMENDACIONES**:

Verificar paso a paso la sintaxis de Python para evitar errores.

***Docente*: __MSc. Orlando Barcia Ayala____**

***Firma*: _____**

# ANALISIS DE RESULTADOS OBTENIDOS

## 5.1. Análisis y resultados práctica # 1

Práctica # 1: Implementación de práctica para reconocimiento de marcadores de visión.

En la práctica se trabajó con los marcadores visuales con el cual se logró implementar el reconocimiento de marcadores visuales por inteligencia artificial del robot Robomaster S1, el cual es una de las características del Robomaster S1.

La práctica comprende que el robot realizara las siguientes actividades al detectar los siguientes números.

Número 1 – se desplazará hacia adelante.

Número 2 – se desplazará hacia atrás.

Número 3 – se desplazará hacia izquierda.

Número 4 – se desplazará hacia derecha.

Número 5 – realizara un giro de 360°

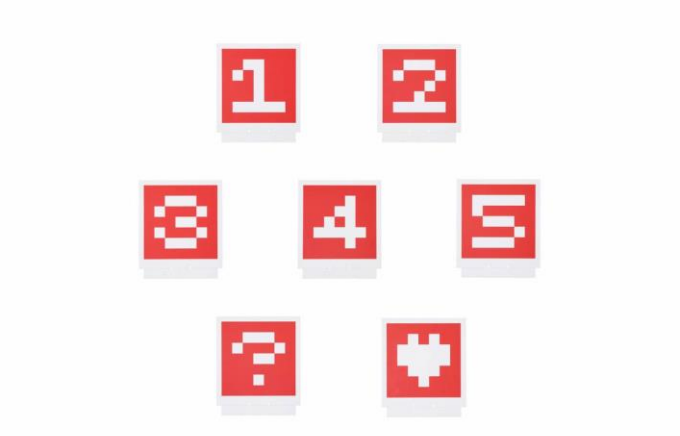


*Figura 0-1: Marcadores de Visión [35]*

Hay que tomar en cuenta que antes de realizar esta práctica se realizó la programación Python de reconocimiento de marcadores visuales.

**CONCLUSIÓN:**


*Figura 0-2: Pantalla inicial de aplicación*

Una vez introducidos al manejo del Software Robomaster manejado por computadora, cómo poder realizar la programación propia, utilizando algoritmos de programación Python ya establecidos para el Robomaster S1, también se logra experimentar el manejo del software en tiempo real logrando así un módulo didáctico de aprendizaje.


*Figura 0-3: Programación en Python*

Se obtiene mayor destreza con el robot Robomaster S1 utilizando los marcadores visuales.

### 5.2. Análisis y resultados práctica # 2

Práctica # 2: Implementación de una práctica para realizar un control PID.

En esta práctica se realizada, una vez familiarizado con el uso y manejo del software Robomaster S1 se verifica que el robot tenga esté conectado con la computadora para proceder a realizar la programación Python.

El proceso de programación se realizó por una secuencia de algoritmos Python ya establecidos para el Robomaster S1 y utilizando la característica de inteligencia artificial del robot se logra la identificación de la línea implementándose un autotuning en la programación para poder realizar el control PID.

*Figura 0-4: Robomaster como seguidor de línea*

## CONCLUSIÓN:

Una vez introducidos al manejo del Software Robomaster manejado por computadora, Se logró realizar la secuencia de algoritmo en programación Python para que pueda ejecutar el objetivo planteado.

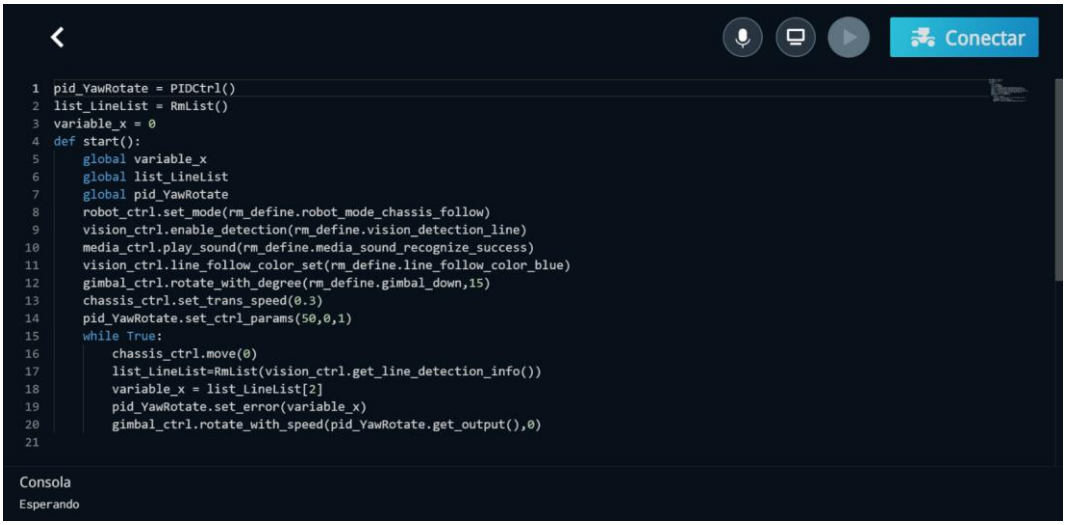



*Figura 0-5: Programación en Python*

Se obtiene mayor destreza con el robot Robomaster S1 realizando un control PID en un seguidor de línea.

### 5.3. Análisis y resultados práctica # 3

Práctica # 3: Implementación de una práctica utilizando algoritmo SLAM para localización y mapeo simultáneo.

En esta práctica realizada se trabajó con el robot Robomaster S1, con el cual se implementó la práctica para el movimiento del robot dentro de una habitación, la cual se realizó el mapeo.

El proceso de programación se realizó por una secuencia de algoritmos Python ya establecidos para el Robomaster S1 y utilizando la característica de inteligencia artificial del robot se identifica los cuatro marcadores visuales que se utilizaron para realizar el movimiento dentro de la habitación.

Adicional se implementó un controlador Raspberry para conectar el sensor LiDAR que realizó el mapeo por medio de Héctor SLAM.

El acceso a la Raspberry se conecta por medio de conexión WIFI

Se Configura la IPV4 de la computadora para que coincida con la del Robomaster S1.

Se Coloca los siguientes parámetros:

Dirección IP: 192.168.4.8

Máscara de subred: 255.255.255.0

Puerta de enlace predeterminada: 192.168.4.1

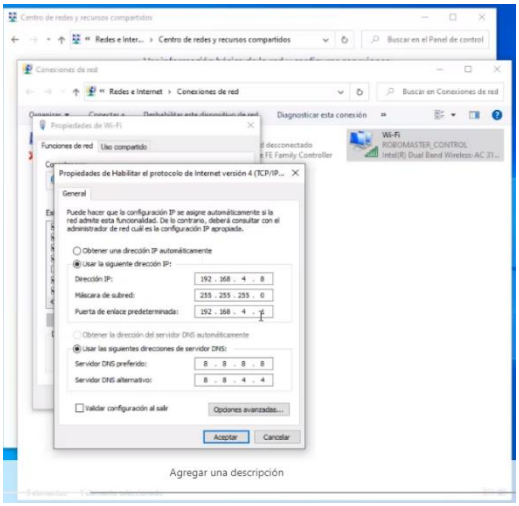



*Figura 0-6: Dirección de puerta de enlace*

Nombre de Red: ROBOMASTER_CONTROL
Contraseña: Clave123456

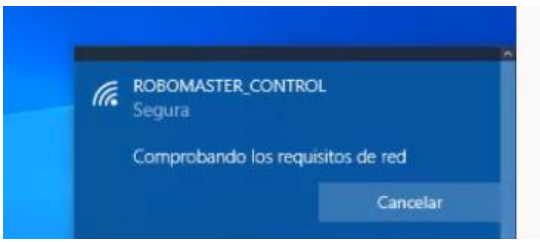



*Figura 0-7: Conexión Wi-Fi*

Se Conecta a la Raspberry mediante servidor VNC.

Doble click en la Raspberry de nombre Robomaster, se accede con las credenciales:
Usuario: pi
Contraseña: raspberry

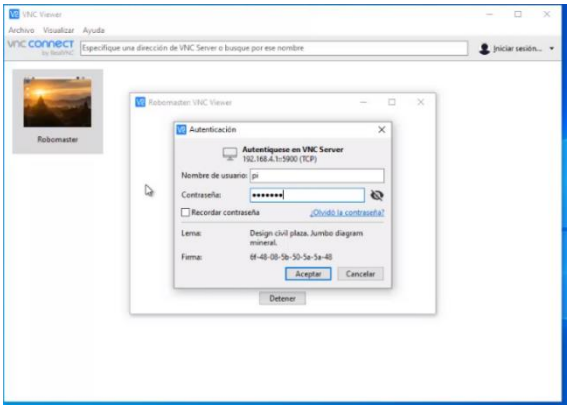



*Figura 0-8: Conexión a VNC Viewer*

Una vez se tenga acceso a la pantalla de la Raspberry se podrá abrir una o varias ventanas de comandos para proceder al siguiente paso.

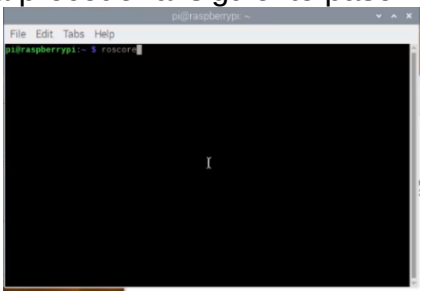



*Figura 0-9: Ventana de comandos*

Inicialización de ROS
Prueba de arranque y captura de datos del sensor LiDAR
Habilitar comunicación entre Robomaster S1 y Arduino
Habilitar el envío de caracteres
Arranque de LiDAR
Mapeo mediante Hector SLAM

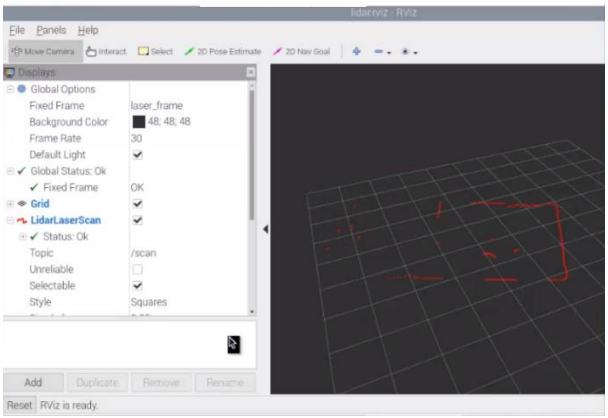



*Figura 0-10: Arranque de LiDAR*

## CONCLUSIÓN:

Una vez introducidos al manejo del Software Robomaster manejado por computadora, Se obtiene mayor destreza con el robot Robomaster S1 un movimiento autónomo dentro de una habitación

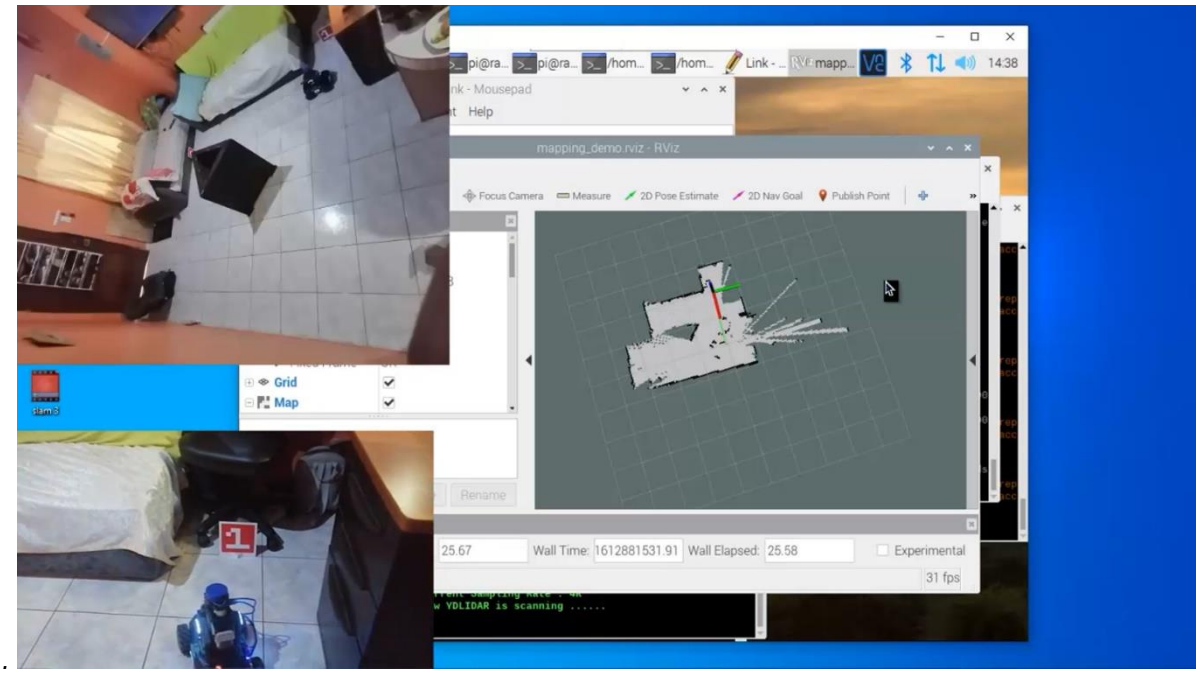



*Figura 0-11: Ejecución de prácita LiDAR*

Se logró realizar la secuencia de algoritmo en programación Python para que pueda ejecutar el objetivo planteado.

### 5.4. Análisis y resultados práctica # 4

Práctica # 4: Implementación de una práctica para identificar y rastrear una persona.

En esta práctica realizada se trabajó con el robot Robomaster S1, con el cual se implementó la práctica de identificar una persona para proceder a seguirla cuando la persona se encuentre en frente del mismo.

*Figura 0-12: Ubicación de sujeto para reconocimiento*

Hay que tomar en cuenta que el robot tenga esté conectado con la computadora para proceder a realizar la programación Python, y ejecutar el programa.

El proceso de programación se realizó utilizando la característica de inteligencia artificial del robot se logra la identificación de la persona.

## CONCLUSIÓN:

Una vez adaptados al uso del software Robomaster se debe tomar cuenta de utilizar los algoritmos de programación Python ya establecidos para el Robomaster S1.

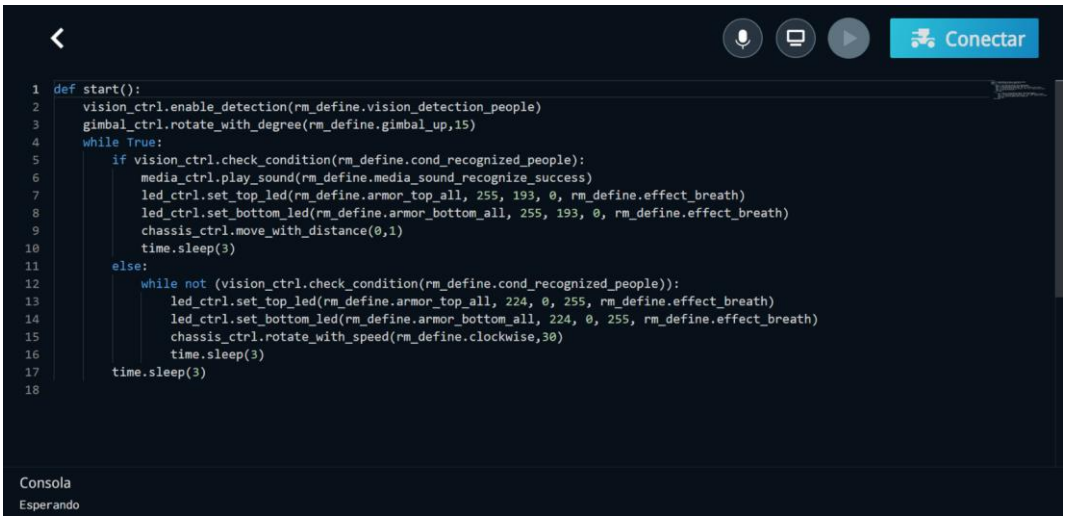



*Figura 0-13: Programación en Python*

Se obtiene mayor destreza con el robot Robomaster S1 realizando la identificación y seguimiento de una persona.

Se logró realizar la secuencia de algoritmo en programación Python para que pueda ejecutar el objetivo planteado.

### 5.5. Análisis y resultados práctica # 5

Práctica # 5: Implementación de una práctica para respuestas únicas al reconocimiento de aplausos.

En esta práctica realizada se trabajó con el robot Robomaster S1, con el cual se implementó la práctica para respuestas únicas al reconocimiento de aplausos, la cual al realizar dos aplausos el robot girará 360°, luego al proceder con tres aplausos el robot esperará dos segundos y apagará las luces leds con la cámara de grabación.

*Figura 0-14: Ubicación del sujeto para práctica aplausos*

Hay que tomar en cuenta que el robot tenga esté conectado con la computadora para proceder a realizar la programación Python, y ejecutar el programa.

**CONCLUSIÓN:**

Una vez adaptados al uso del software Robomaster se debe tomar cuenta de utilizar los algoritmos de programación Python ya establecidos para el Robomaster S1.

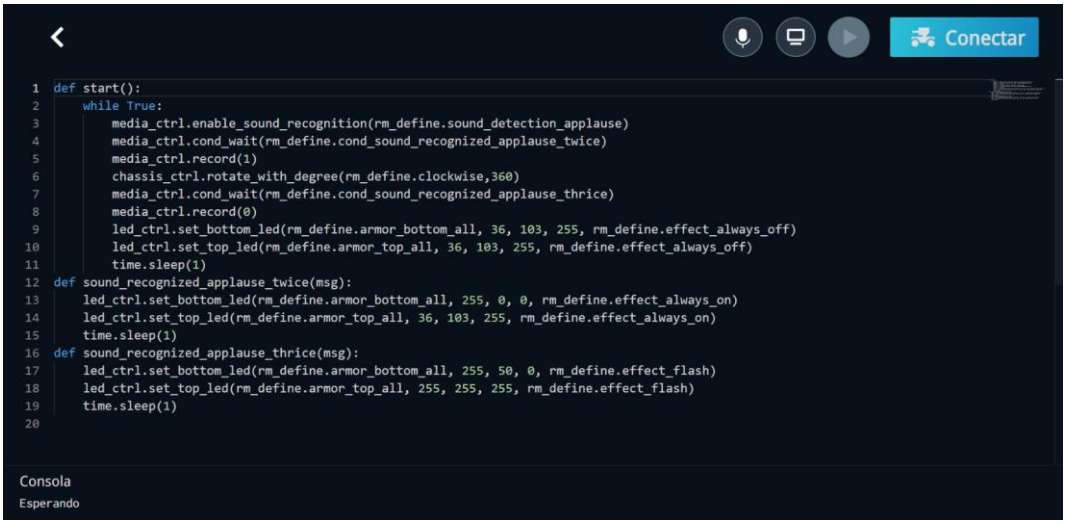



*Figura 0-15: Programación en Python*

Se obtiene mayor destreza con el robot Robomaster S1 realizando la identificación y seguimiento de una persona.

Se logró realizar la secuencia de algoritmo en programación Python para que pueda ejecutar el objetivo planteado.

### 5.6. Análisis y resultados práctica # 6

Práctica # 6: Implementación de una práctica para respuestas autodefinidas al reconocimiento de gestos físicos de una persona.

En esta práctica realizada se trabajó con el robot Robomaster S1, con el cual se implementó la práctica para respuestas autodefinidas al reconocimiento de gestos físicos de una persona.

La práctica comprende que el robot realizara las siguientes actividades al detectar las siguientes condiciones:

Al identificar la V realizada con los brazos – se desplazará hacia adelante.

Al identificar la V invertida realizada con los brazos – se desplazará hacia atrás.

*Figura 0-16: Ubicación del sujeto para prácita gestos*

Hay que tomar en cuenta que el robot tenga esté conectado con la computadora para proceder a realizar la programación Python, y ejecutar el programa.

El proceso de programación se realizó utilizando la característica de inteligencia artificial del robot se logra la identificación de los gestos.

**CONCLUSIÓN:**

Una vez adaptados al uso del software Robomaster se debe tomar cuenta de utilizar los algoritmos de programación Python ya establecidos para el Robomaster S1.


*Figura 0-17: Programación en Python*

Se obtiene mayor destreza con el robot Robomaster S1 realizando la identificación de gestos de una persona realizada con los brazos.

Se logró realizar la secuencia de algoritmo en programación Python para que pueda ejecutar el objetivo planteado.

## 5.7. Análisis y resultados práctica # 7

Práctica # 7: Implementación de una práctica para un robot seguidor de línea.

En esta práctica realizada se trabajó con el robot Robomaster S1, con el cual se implementó la práctica para un robot seguidor de línea.

La práctica comprende que el robot realizara las siguientes actividades:

Al colocar el programa color azul – El robot seguirá la línea azul.

Al colocar el programa color rojo – El robot seguirá la línea roja.

Al colocar el programa color verde – El robot seguirá la línea verde.

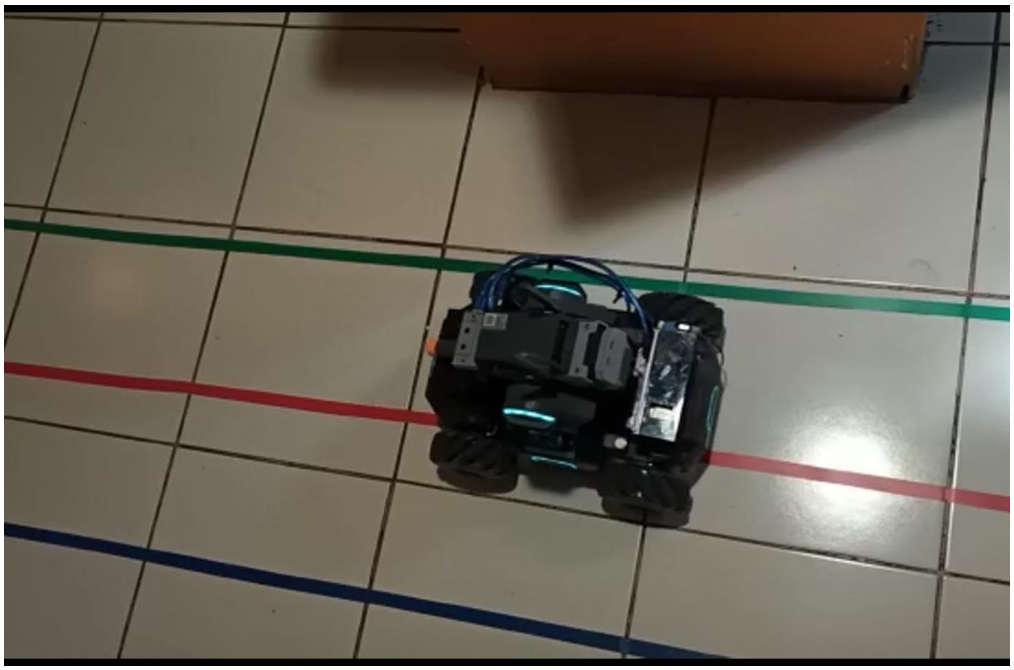


*Figura 0-18: Robomaster como seguidor de línea*

Hay que tomar en cuenta que el robot tenga esté conectado con la computadora para proceder a realizar la programación Python, y ejecutar el programa.

El proceso de programación se realizó utilizando la característica de inteligencia artificial del robot se logra la identificación los colores de la línea.

**CONCLUSIÓN:**

Una vez adaptados al uso del software Robomaster se debe tomar cuenta de utilizar los algoritmos de programación Python ya establecidos para el Robomaster S1.


*Figura 0-19: Programación en Python*

Se obtiene mayor destreza con el robot Robomaster S1 realizando la identificación y seguimiento del color de las líneas propuestas.

Se logró realizar la secuencia de algoritmo en programación Python para que pueda ejecutar el objetivo planteado.

### 5.8. Análisis y resultados práctica # 8

Práctica # 8: Implementación de una práctica para un robot explorador radio controlado.

En esta práctica realizada se trabajó con el robot Robomaster S1, con el cual se implementó la práctica para un robot radio explorador radio controlado

La práctica comprende de las siguientes actividades:

- Conectar un control remoto para controlar los movimientos del robot en la aplicación.
- Implementar la utilización de la app Shooting Plus V3 para utilizar acoplador en el uso del control remoto.

- Uso de la app Robomaster en un celular Android.

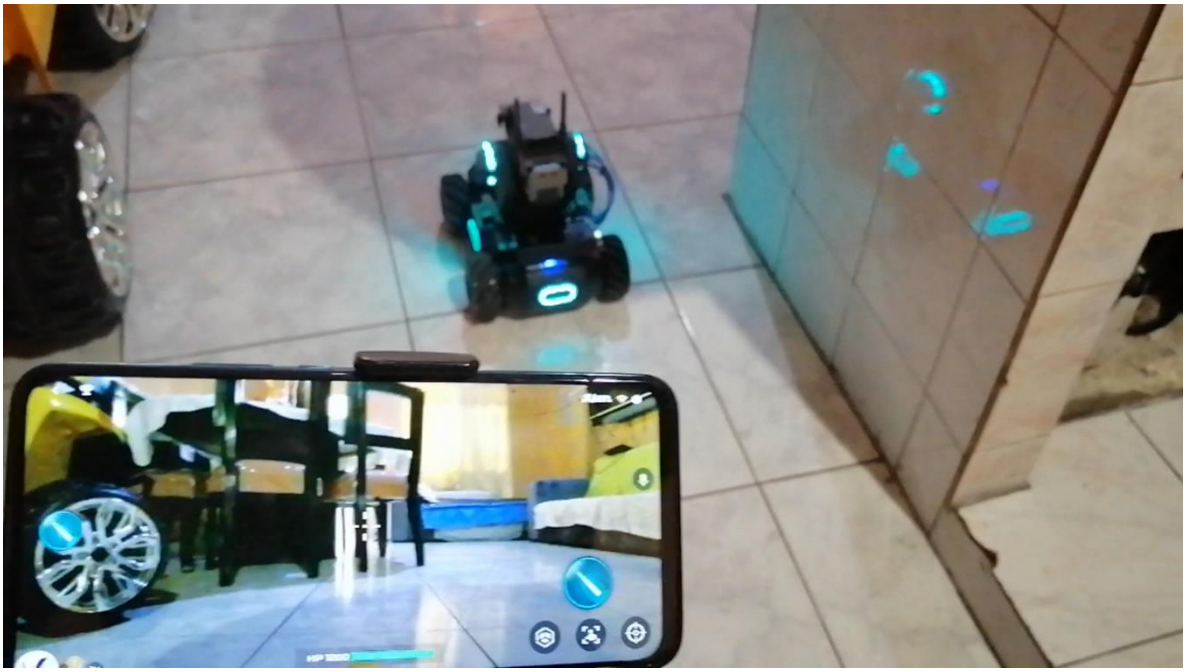


*Figura 0-20: Robomaster en modo de exploración*

**CONCLUSIÓN:**

Se logra comprender el uso de la app Robomaster en un celular Android, y la utilización del control remoto por conexión bluetooth mediante la app Shooting Plus V3.

El robot Robomaster S1 brinda muchas posibilidades para adquirir destrezas en la parte robótica.

Se logró realizar el objetivo planteado.

### 5.9. Análisis y resultados práctica # 9

Práctica # 9: Implementación de una práctica para un robot de batalla utilizando el cañón de disparo incorporado con objetivo fijo.

En esta práctica realizada se trabajó con el robot Robomaster S1, con el cual se implementó la práctica para un robot de batalla utilizando el cañón de disparo incorporado con objetivo fijo.

La práctica comprende de las siguientes actividades:

- Identificar los objetivos, que para este caso se programaron los marcadores visuales del 1 al 5.
- Cuando se identifique el objetivo el robot hará uso de cañón de disparo para lanzar dos bolitas de gel.
- Solo el disparo se ejecutará cuando el objetivo se encuentre fijo.


*Figura 0-21: Ubicación del sujet, práctica con cañón*

## CONCLUSIÓN:

Una vez adaptados al uso del software Robomaster se debe tomar cuenta de utilizar los algoritmos de programación Python ya establecidos para el Robomaster S1.

Se obtiene mayor destreza con el robot Robomaster S1 utilizando el cañón de disparo incorporado con objetivo fijo.

Se logró realizar la secuencia de algoritmo en programación Python para que pueda ejecutar el objetivo planteado.

### 5.10. Análisis y resultados práctica # 10

Práctica # 10: Implementación de una práctica incorporando la previa descrita de un robot radio controlado con el uso del cañón de disparo, detector de impacto, incorporando estabilizador.

En esta práctica realizada se trabajó con el robot Robomaster S1, con el cual se implementó la práctica incorporando la previa descrita de un robot radio controlado con el uso del cañón de disparo, detector de impacto incorporando estabilizador.

La práctica comprende de las siguientes actividades:

- Controlar el robot radio controlado desde el celular Android por medio de la App Robomaster.
- Cuando el robot detecte algún impacto en sus sensores ubicados en el chasis o estabilizador del cañón, girará inmediatamente hacia la posición y disparará cinco bolitas de gel
- Cuando el robot realice un giro a X posición utilizara el estabilizador con ° del chasis para mantener la orientación del cañón hacia el norte.

*Figura 0-22: Ubicación del sujeto para práctica de sensor de impacto*

## CONCLUSIÓN:

Una vez adaptados al uso del software Robomaster se debe tomar cuenta de utilizar los algoritmos de programación Python ya establecidos para el Robomaster S1.

Se obtiene mayor destreza con el robot Robomaster S1 con la práctica incorporando la previa descrita de un robot radio controlado con el uso del cañón de disparo, detector de impacto incorporando estabilizador

Se logró realizar la secuencia de algoritmo en programación Python para que pueda ejecutar el objetivo planteado.

# Conclusiones

1. Se consiguió implementar un banco de prácticas ordenado, pedagógico, sistemático, teórico y práctico, que permite el trabajo individual y en equipo, que mejora las destrezas e incentiva las habilidades investigativas de los estudiantes.
2. Todas las mejoras que se consiguió implementar utilizaron el lenguaje Python, dando apertura a la posibilidad de desarrollar nuevas destrezas.
3. Se consiguió realizar la programación del sistema operativo ROS en Raspberry y las librerías necesarias para la comunicación con las tarjetas arduino y realizar el mapeo 2D en Héctor SLAM.
4. Se consiguió brindar autonomía al Robomaster S1 mediante la implementación de un controlador Raspberry que utilizaba librerías de ROS y técnicas de mapeo en dos dimensiones mediante un sensor LiDAR con los paquetes de Héctor SLAM.
5. Se logró implementar el desarrollo de las prácticas cumpliendo con los parámetros establecidos.
6. Dentro de las pruebas y desarrollo de las prácticas se logra el funcionamiento óptimo de las características propias del Robomaster S1.
7. Estas prácticas fueron desarrolladas por medio del software Robomaster en PC y de la App Robomaster en un celular Android con éxito.
8. Las prácticas se desarrollaron de manera independiente y con una secuencia en algoritmos Python ya establecidos para el uso del Robomaster.
9. Se utilizó la aplicación Shooting Plus V3 que sirvió de enlace para en el control remoto de conexión bluetooth al dispositivo móvil logrando así el control del Robot Robomaster S1.
10. Las guías de prácticas fueron elaboradas simultáneamente con cada una de las prácticas, logrando así la explicación explícita para que se puedan desarrollar sin mayor inconveniente por otra persona.

# Recomendaciones

1. Energizar el robot Robomaster S1 verificando que el estabilizador del cañón no se encuentre obstruido.
2. Verificar la conexión entre el robot Robomaster S1 al PC o dispositivo móvil antes de realizar alguna práctica.
3. Verificar la configuración del adaptador para el enlace Wifi a la Raspberry en la práctica del SLAM.
4. Realizar la programación Python de acuerdo a los algoritmos ya establecidos en el uso del Robomaster S1.
5. Verificar las líneas de programación y variables dentro de la codificación Python, así mismo que en la compilación no se encuentren errores.
6. Se pueden añadir sensores al controlador Raspberry ya que posee entradas analógicas y digitales, y desarrollar mayores destrezas.

# Referencia Bibliográfica


[1] A. García, Diseño, Construcción y Control de un Robot Manipulador de 3 grados de libertad de bajo coste para el desarrollo de un Manipulador Móvil, Valencia, 2016.

[2] E. Álvarez, «Generación de Mapa Global 2D y SLAM usando LiDAR y una Estéreo Cámara para el seguimiento de movimiento de un robot móvil,» *ITECKNE,* vol. 16, nº 2, pp. 144-156, diciembre 2019.

[3] E. Aldas, Plataforma Robótica Móvil para Experimentos de Mapeo y Localizacón Simultánea (SLAM) en base a Sensores de Rango, Riobamba, 2017.

[4] E. Knight, «Knight's American Mechanical Dictionary,» 1876.

[5] C. b. Ji-Elle, «Gallo autómata del primer reloj (hacia 1350),» 2010.

[6] A. Konby, «Scientific American Volume 80 Number 03,» 1899.

[7] F. Sánchez-Martin, «Historia de la robótica: de Arquitas de Tarento al Robot da Vinci. (Parte II),» *Actas Urológicas Españolas,* vol. 31, nº 3, pp. 185-196, marzo 2007.

[8] EBatlleP, «Sketch of a Unimate robot,» 2019.

[9] ATI, «RCC Remote Compliance Compensator,» 2021.

[10] F. Sánchez-Martin, «Historia de la robótica: de Arquitas de Tarento al rrobot da Vinci (Parte I),» *Actas Urológicas Españolas,* vol. 61, nº 2, pp. 69-76, 2007.

[1] K. Pittí y I. Moreno, «La robótica educativa, una herramienta para la enseñanza-aprendizaje de las ciencias y las tecnologías,» *Revista Teoría de la Educación: Educación y Cultura en la Sociedad de la Información,* vol. 2, nº 13, pp. 74-90, enero 2012.

[2] F. Ramos de la Flor, Robótica Educativa: prácticas y actividades, RA-MA Editorial, 2016.

[3] M. Meza, Diseño y construcciónd e un robot acuático, Bogotá, 2015.

[4] B. Aguirre, Diseño e implementación de un sistema multirobot descentralizado para realizar trabajo colaborativo con aplicaciones logísticas, flexibles y escalables, Sangolquí, 2019.

[5] NYTimes, 2018. [En línea]. Available: https://www.nytimes.com/es/2018/03/26/espanol/peces-robot-oceanos.html.

[6] BostonDynamics, 2009. [En línea]. Available: http://www.peatom.info/universidad/122057/robots-que-saltan-obstaculos-de-8-metros/.

[7] TimeToast, «Línea de Tiempo de Robots,» 2020. [En línea]. Available: https://www.timetoast.com/timelines/la-robotica-e34bc095-5b49-4c8f-8a4d-7ece76e7ba2c.

[8] NASA, «Perseverance Rover,» 2021. [En línea]. Available: https://www.space.com/perseverance-rover-mars-2020-mission.

[19] G. Argudo, Diseño y construcción de un robot móvil teleoperado para la asistencia en operaciones de alto riesgo del cuerpo de bomberos, Cuenca, 2012.

[20] Z. Dechao, «Research and Development of High Precision Robot Automatic Docking System Based on mxAutomation,» de *Journal of Physics Conference Series*, 2019.

[21] Appin Knowledge Solutions, Robotics, Hingham, 2007.

[22] T. Bräunl, EMBEDDED ROBOTICS, Perth: Springer, 2006.

[23] Arduino, «arduino,» 2021. [En línea]. Available: https://www.arduino.cc/en/Guide/Introduction.

[24] MCI Electronics, «MCI Electronics - Arduino Chile,» 2021. [En línea]. Available: https://arduino.cl/.

[25] sparkfun, «SparkFun Electronics,» 2021. [En línea]. Available: https://www.sparkfun.com/products/12640.

[26] Raspberry Pi, «about,» 2021. [En línea]. Available: https://www.raspberrypi.org/about/.

[27] Raspberry Pi, «Raspberry Pi 4 modelo B,» 2020. [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/.

[28] SocialCompare, «Raspberry Pi models comparison,» 2021. [En línea]. Available: https://socialcompare.com/en/comparison/raspberrypi-models-comparison.

[29] S. Allende, Sistema Operativo Linux Teoría y Práctica, Córdoba: EduTecne, 2019.

[30] G. Halfacree, The official Raspberry Pi beginner's guide How to use your new computer, Raspberry Pi Press, 2018.

[31] C. Severance, «Python para todos,» 2020.

[32] ANII, *Breve manual de programación en Python,* Montevideo, 2013.

[33] motorpasion, «Motorpasion,» 2017. [En línea]. Available: https://www.motorpasion.com/tecnologia/que-es-un-lidar-y-como-funciona-el-sistema-de-medicion-y-deteccion-de-objetos-mediante-laser.

[34] ROS, «ROS Org,» 2021. [En línea]. Available: https://www.ros.org/.

[35] DJI, «DJI - Robomaster S1,» 2021. [En línea]. Available: https://www.dji.com/robomaster-s1.

[36] Mrmw, «Mecanum Wheel control principle,» 2020. [En línea]. Available: https://category.yahboom.net/collections/micro-bit/products/omni-bit.

[37] DJI, «Sitio de descargas,» 2021. [En línea]. Available: https://www.dji.com/downloads/djiapp/robomaster.

[38] J. Goldman, «DJI's new $500 RC robot features a camera, 31 sensors and a mini cannon,» 2019. [En línea]. Available: https://www.cnet.com/news/djis-new-500-rc-robot-features-a-camera-31-sensors-and-a-mini-cannon/.

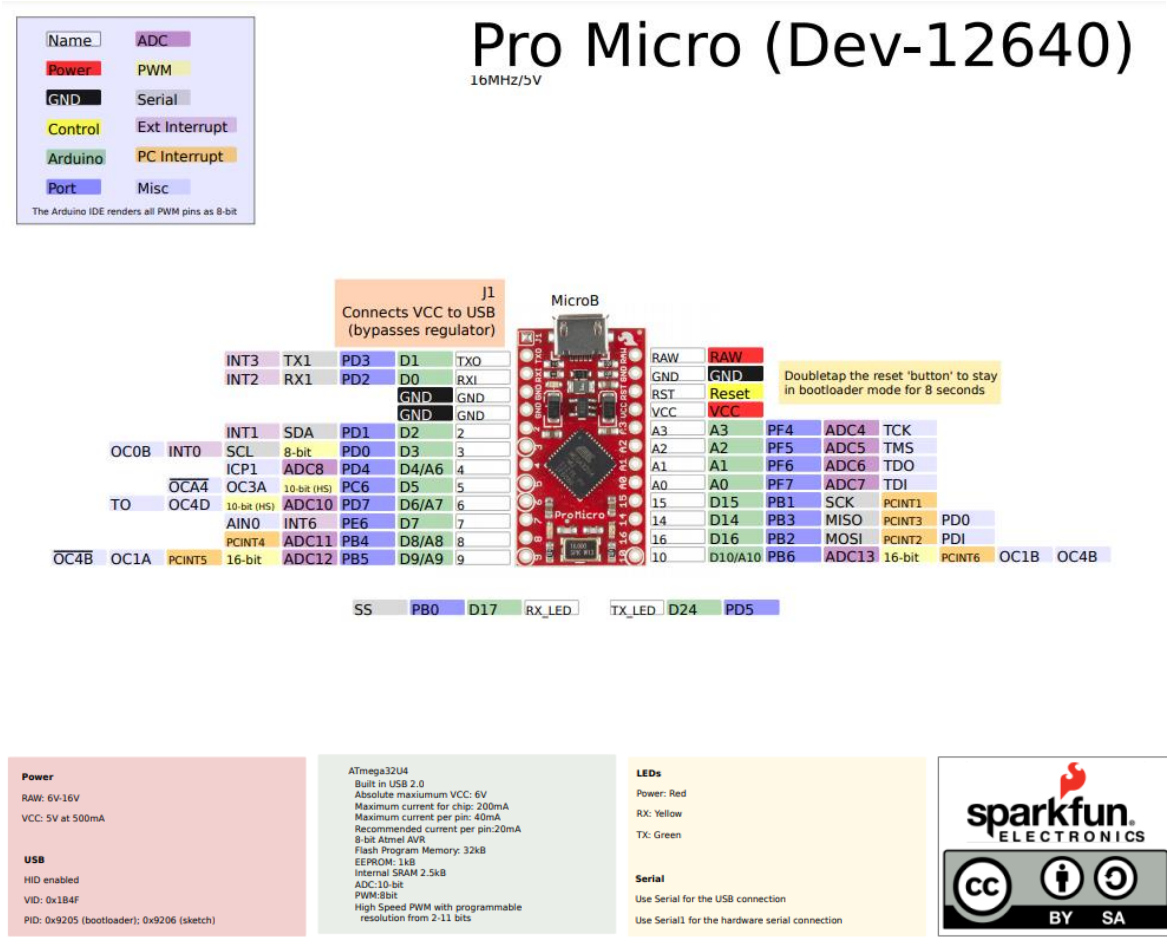
[39] Amazon, «Controlador inalámbrico para Robomaster S1,» 2021. [En línea]. Available: https://www.amazon.com/gp/product/B07XQ9LMP1/ref=ppx_yo_dt_b_asin_title_o02_s00?ie=UTF8&psc=1.

[40] JetBrains, «Download Pycharm,» 2020. [En línea]. Available: https://www.jetbrains.com/pycharm/download/#section=linux.

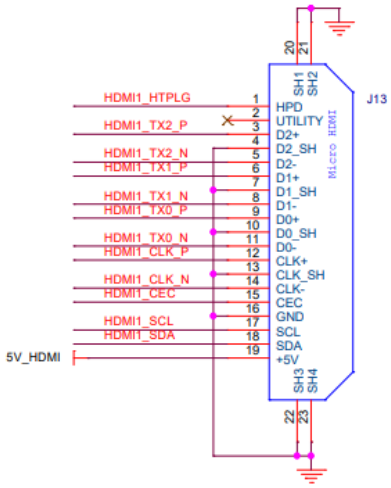
[41] RealVNC, «Descargue VNC Viewer,» 2020. [En línea]. Available: https://www.realvnc.com/es/connect/download/viewer/.

# Anexos

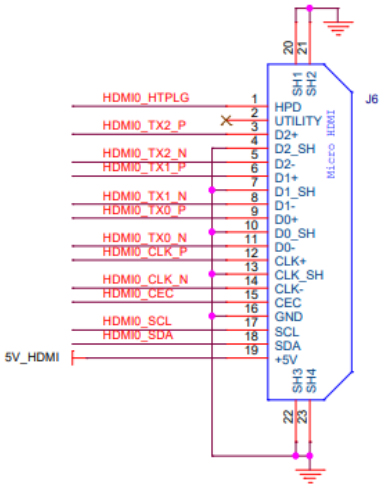
## 6.1. Hoja de datos de Arduino Compatible sparkfun Pro Micro



## 6.2. Esquemático de Raspberry Pi 4 Modelo B

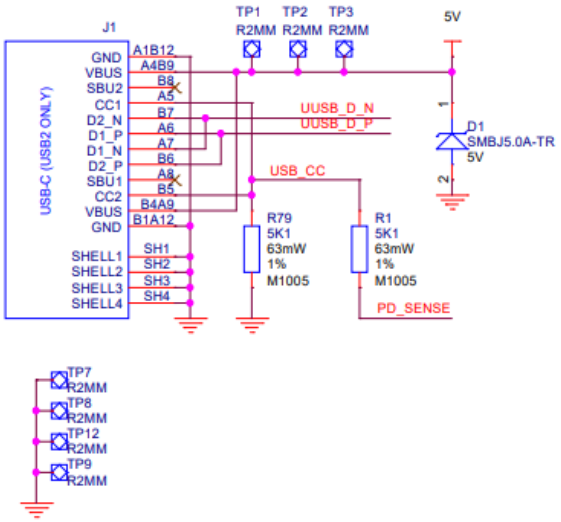


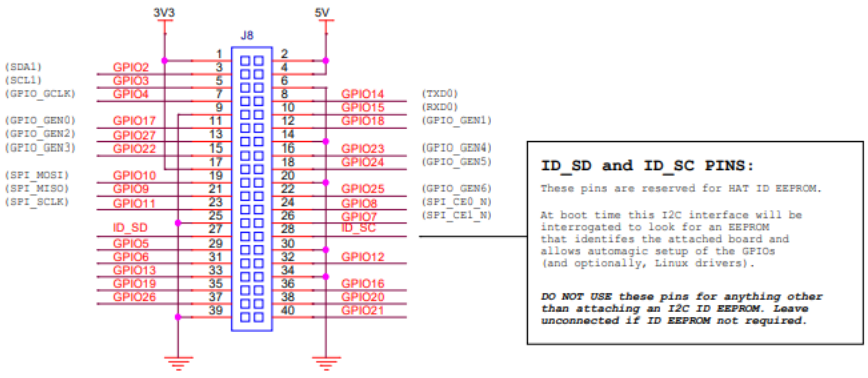

MICRO HDMI 1



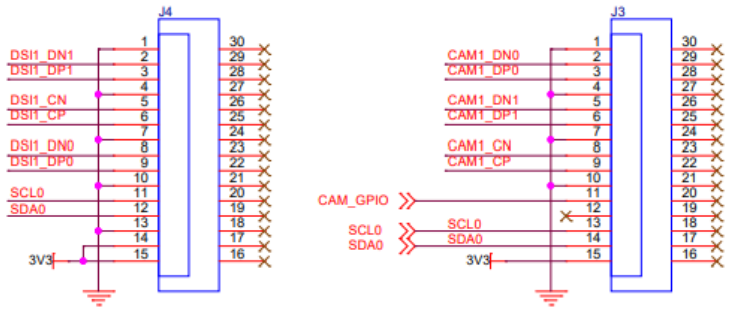

MICRO HDMI 0

USB-C POWER IN



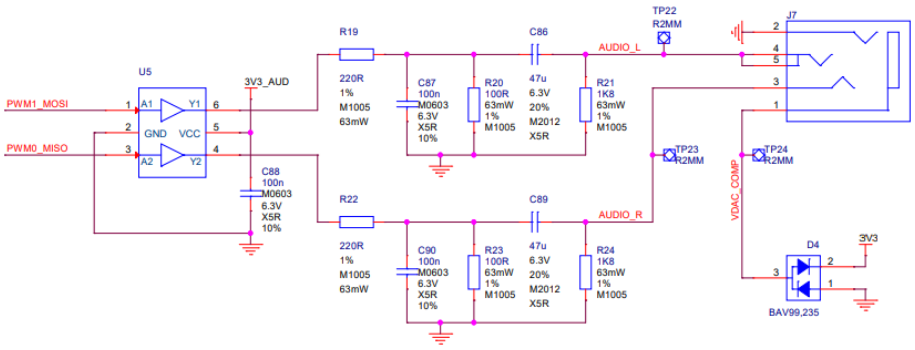




GPIO EXPANSION

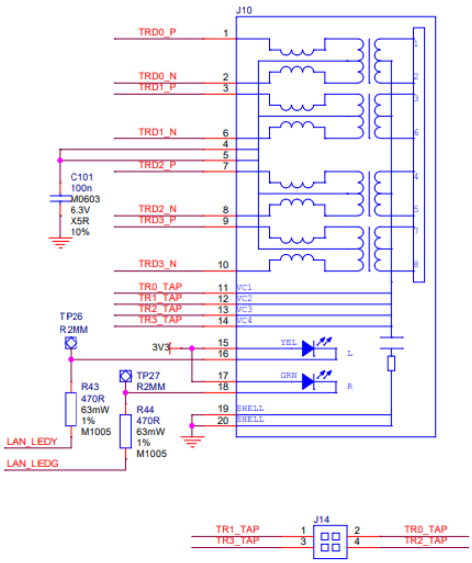



DISPLAY CAMERA

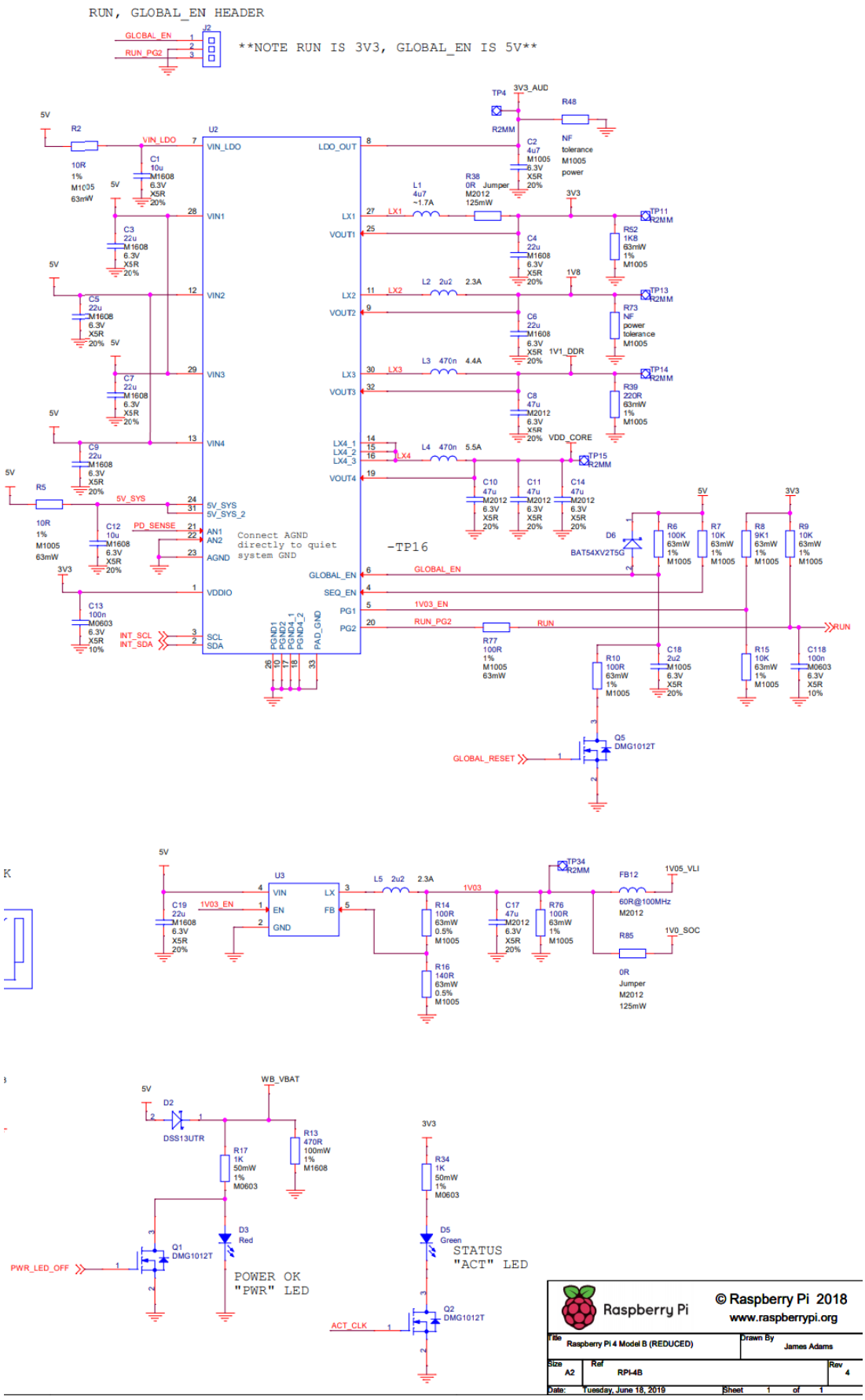
A/V JACK




GIGABIT ETHERNET




PoE HEADER


RUN, GLOBAL_EN HEADER
**NOTE RUN IS 3V3, GLOBAL_EN IS 5V**
Connect AGND directly to quiet system GND
-TP16
POWER OK "PWR" LED
STATUS "ACT" LED
Raspberry Pi
© Raspberry Pi 2018
www.raspberrypi.org
Raspberry Pi 4 Model B (REDUCED)
James Adams
RPI-4B
Tuesday, June 18, 2019

### 6.3. Código de la Arduino Nano

```
#if (ARDUINO >= 100)
#include <Arduino.h>
#else
#include <WProgram.h>
#endif

#include <ros.h>
#include <geometry_msgs/Twist.h>
// Pin variables for motors.
const int adelante = 5;
const int atras = 6;
const int izquierda = 7;
const int derecha = 8;
const int stop1 =9;
const int ledPin = 13;
ros::NodeHandle  nh;

void MoveFwd() {
  digitalWrite(adelante, HIGH);
  digitalWrite(stop1, LOW);
 digitalWrite(ledPin, HIGH);
}

void MoveStop() {
  digitalWrite(stop1, HIGH);
  digitalWrite(izquierda, LOW);
   digitalWrite(derecha, LOW);
    digitalWrite(adelante, LOW);
    digitalWrite(atras, LOW);
    digitalWrite(ledPin, HIGH);
}
void MoveLeft() {
  digitalWrite(izquierda, HIGH);
  digitalWrite(stop1, LOW);
  digitalWrite(ledPin, LOW);
}
void MoveRight() {
 digitalWrite(derecha, HIGH);
 digitalWrite(stop1, LOW);
```

```
 digitalWrite(ledPin, HIGH);
}
void MoveBack() {
  digitalWrite(atras, HIGH);
  digitalWrite(stop1, LOW);
  digitalWrite(ledPin, LOW);
 }

void cmd_vel_cb(const geometry_msgs::Twist & msg) {
  // Read the message. Act accordingly.
  // We only care about the linear x, and the rotational z.
  const float x = msg.linear.x;
  const float z_rotation = msg.angular.z;

  // Decide on the morot state we need, according to command.
  if (x > 0 && z_rotation == 0) {
    MoveFwd();

  }
  //else if (x == 0 && z_rotation == 1) {
  else if (x == 0 && z_rotation > 0) {
    MoveRight();
  }
else if (x == 0 && z_rotation < 0) {
    MoveLeft();
  }
else if (x < 0 && z_rotation == 0) {
    MoveBack();

  }
else if (x == 0 && z_rotation == 0) {
    MoveStop();

  }
}
ros::Subscriber<geometry_msgs::Twist> sub("cmd_vel", cmd_vel_cb);
void setup() {
  pinMode(adelante, OUTPUT);
  pinMode(atras, OUTPUT);
  pinMode(izquierda, OUTPUT);
  pinMode(derecha, OUTPUT);
```

```
  pinMode(stop1, OUTPUT);
  pinMode(ledPin, OUTPUT);
  nh.initNode();
  nh.subscribe(sub);
}

void loop() {
  nh.spinOnce();
  delay(1);
}
```

## 6.4. Código de la Pro Micro

```
1. #include "Keyboard.h"
2.
3. const int buttonPin1 = 2;          // input pin for pushbutton
4. const int buttonPin2 = 3;          // input pin for pushbutton
5. const int buttonPin3 = 4;          // input pin for pushbutton
6. const int buttonPin4 = 5;          // input pin for pushbutton
7. const int buttonPin5 = 6;          // input pin for pushbutton
8. int previousButtonState1 = HIGH;   // for checking the state of a
   pushButton
9. int previousButtonState2 = HIGH;   // for checking the state of a
   pushButton
10. int previousButtonState3 = HIGH;   // for checking the state of a
   pushButton
11. int previousButtonState4 = HIGH;   // for checking the state of a
   pushButton
12. int previousButtonState5 = HIGH;   // for checking the state of a
   pushButton
13. int counter = 0;                  // button push counter
14.
15. void setup() {
16.
17.   pinMode(9,OUTPUT);
18.   Serial.begin(9600);
19.   digitalWrite(9, HIGH);
20.   delay(1000);
21.   digitalWrite(9, LOW);
22.   Serial.println("on");
23.   delay(1000);
24.     digitalWrite(9, HIGH);
25.   delay(1000);
26.   digitalWrite(9, LOW);
27.  delay(7000);
28.   // make the pushButton pin an input:
29.   pinMode(buttonPin1, INPUT);
30.   pinMode(buttonPin2, INPUT);
31.   pinMode(buttonPin3, INPUT);
32.   pinMode(buttonPin4, INPUT);
33.   pinMode(buttonPin5, INPUT);
34.   // initialize control over the keyboard:
```

```
  Keyboard.begin();
}

void loop() {
  // Keyboard.press("W");
  //  Keyboard.releaseAll();


  int buttonState1 = digitalRead(buttonPin1);
  int buttonState2 = digitalRead(buttonPin2);
  int buttonState3 = digitalRead(buttonPin3);
  int buttonState4 = digitalRead(buttonPin4);
  int buttonState5 = digitalRead(buttonPin5);
  // if the button state has changed,
  if ((buttonState1 != previousButtonState1)
      // and it's currently pressed:
      && (buttonState1 == HIGH)) {
    // increment the button counter

    // type out a message
    Keyboard.press('w');
    digitalWrite(9, HIGH);

  }
  if ((buttonState2 != previousButtonState2)
      // and it's currently pressed:
      && (buttonState2 == HIGH)) {
    // increment the button counter
    // type out a message
    Keyboard.press('s');
    digitalWrite(9, LOW);
  }

  if ((buttonState3 != previousButtonState3)
      // and it's currently pressed:
      && (buttonState3 == HIGH)) {
    // increment the button counter
    Keyboard.press('d');
    digitalWrite(9, HIGH);
  }
  if ((buttonState4 != previousButtonState4)
```

```
      // and it's currently pressed:
      && (buttonState4 == HIGH)) {
    // increment the button counter
    Keyboard.press('a');
    digitalWrite(9, LOW);
  }
    if ((buttonState5 != previousButtonState5)
      // and it's currently pressed:
      && (buttonState5 == HIGH)) {
    // increment the button counter
    //Keyboard.press("D");
    Keyboard.releaseAll();
    digitalWrite(9, LOW);
  }
 // else {


 //   Keyboard.releaseAll();
 // }

  // save the current button state for comparison next time:
  previousButtonState1 = buttonState1;
  previousButtonState2 = buttonState2;
  previousButtonState3 = buttonState3;
  previousButtonState4 = buttonState4;
  previousButtonState5 = buttonState5;
  delay(100);
}
```

## 6.5. LEDs de la batería del S1 e indicadores

| Indicadores del nivel de batería durante la carga | | | | |
|---|---|---|---|---|
| Led 1 | Led 2 | Led 3 | Led 4 | Nivel de batería |
| ☀ | ☀ | ○ | ○ | 0 %~50 % |
| ☀ | ☀ | ☀ | ○ | 50 %~75 % |
| ☀ | ☀ | ☀ | ☀ | 75 %~100 % |
| ○ | ○ | ○ | ○ | Carga completa |

| Indicadores de nivel de batería para protección de la batería | | | | | |
|---|---|---|---|---|---|
| Led 1 | Led 2 | Led 3 | Led 4 | Patrón de parpadeo | Elemento de protección de la batería |
| ○ | ☀ | ○ | ○ | El led 2 parpadea dos veces por segundo | Se ha detectado sobrecorriente |
| ○ | ☀ | ○ | ○ | El led 2 parpadea tres veces por segundo | Se ha detectado un cortocircuito |
| ○ | ○ | ☀ | ○ | El led 3 parpadea dos veces por segundo | Se ha detectado sobrecarga |
| ○ | ○ | ☀ | ○ | El led 3 parpadea tres veces por segundo | Se ha detectado sobrevoltaje del cargador |
| ○ | ○ | ○ | ☀ | El led 4 parpadea dos veces por segundo | Temperatura de carga demasiado baja (<0 °C) |
| ○ | ○ | ○ | ☀ | El led 4 parpadea tres veces por segundo | Temperatura de carga demasiado alta (>40 °C) |