UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE GUAYAQUIL

CARRERA DE MECATRÓNICA

DISEÑO DE UN CONTROLADOR INTELIGENTE PARA UN SISTEMA INESTABLE TIPO PÉNDULO INVERTIDO CON MONITOREO REMOTO POR IoT

Trabajo de titulación previo a la obtención del
Título de Ingeniero en Mecatrónica

AUTORES: Carlos Enrique Infante Mejia
Jorge Andrés Navarrete Burgos
TUTOR: Ing. Franklin Illich Kuonquí Gaínza, Mgtr.

Guayaquil - Ecuador

2024

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, **Carlos Enrique Infante Mejia** con documento de identificación N° **0952890655** y **Jorge Andres Navarrete Burgos** con documento de identificación N° **0932357700**; manifestamos que:

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

Guayaquil, 14 de abril del año 2024

Atentamente,

Carlos Enrique Infante Mejia
0952890655

Jorge Andres Navarrete Burgos
0932357700

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

Nosotros, **Carlos Enrique Infante Mejia** con documento de identificación N° **0952890655** y **Jorge Andres Navarrete Burgos** con documento de identificación N° **0932357700**, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del **Dispositivo Tecnológico: DISEÑO DE UN CONTROLADOR INTELIGENTE PARA UN SISTEMA INESTABLE TIPO PÉNDULO INVERTIDO CON MONITOREO REMOTO POR IoT**, el cual ha sido desarrollado para optar por el título de: Ingeniero en Mecatrónica, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

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

Guayaquil, 14 de Abril del año 2024

Atentamente,

Carlos Enrique Infante Mejia
0952890655

Jorge Andres Navarrete Burgos
0932357700

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, **Franklin Illich Kuonquí Gaínza**, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: **DISEÑO DE UN CONTROLADOR INTELIGENTE PARA UN SISTEMA INESTABLE TIPO PÉNDULO INVERTIDO CON MONITOREO REMOTO POR IoT**, realizado por **Carlos Enrique Infante Mejia** con documento de identificación N° **0952890655** y por **Jorge Andres Navarrete Burgos** con documento de identificación N° **0932357700**, obteniendo como resultado final el trabajo de titulación bajo la opción **Dispositivo Tecnológico** que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, 14 de Abril del año 2024

Atentamente,

Ing. Franklin Illich Kuonqui, Mgtr.
0914335484

DEDICATORIA

Este trabajo de titulación está dedicado a mi eterna madre Alexandra Maria Mejia Vera que se que siente muy orgullosa de mí y a mi padre Miguel Angel Infante Quiroz, por haberme enseñado el valor del esfuerzo, del sacrificio y del empeño a lograr llegar a la meta sin importar que tan difícil sea, ya que ellos son el motor de mi día a día.

También este título se lo dedico a mi novia Alanyz Nicole Alava Nieto que junto a sus padres me han brindado su apoyo y su paciencia y me han motivado a seguir adelante en este camino profesional y poder culminar esta etapa tan importante y especial

A mi hermana y mi hermano también le dedico este título y sobrinos, que ellos siempre han visto en mi mucha perseverancia, que jamás he bajado los brazos y sigo luchando por logrando mis objetivos

Carlos Enrique Infante Mejia

Dedicado este momento a Dios porque por el tengo fuerza cada día para seguir adelante a mis padres, hermanos, tíos, abuelos y familiares porque este logro no es mío es de ellos, a mis amigos que siempre me apoyaron sin importar la razón o el motivo y a mis profesores por su tiempo y paciencia para estar conmigo.

Jorge Andres Navarrete Burgos

AGRADECIMIENTO

En primer lugar, quiero agradecer a Dios por permitirme haber culminado una etapa más en mi vida, segundo quiero agradecer a una persona muy especial a la cual la llevo siempre en mi corazón que es a mi eterna madre Alexandra Maria Mejia Vera y a mis compañeros de vida que son mi familia mi padre Miguel Angel Infante Quiroz junto a mi hermana, hermano y sobrinos y novia Alanyz Nicole Álava Nieto

Agradezco a la familia Álava Nieto por apoyarme siempre, estar conmigo brindando su apoyo en todo momento, con su cariño, afecto y más en esta etapa tan importante en mi vida profesional,

A mis profesores y mentores, que han sabido guiarme por el camino correcto y sobre todo enseñarme porque he aprendido bastante de ellos, y he valorado cada una de sus palabras fuera del aula de clase para seguir por el camino del bien.

Agradezco a mi tutor el ING Franklin Kuonqui, que desde el primer día nos brindó su confianza su sabiduría, nos brindó la mayor parte de su tiempo y nos hizo confiar en nosotros mismo al poder lograr alcanzar esta nueva etapa ya que gracias a él lo logramos y este logro académico también es para él.

Carlos Enrique Infante Mejia

Agradezco a Dios por la vida y por la oportunidad de poder vivir este momento tan importante y alegre para muchas personas, a mi familia que siempre estás atrás mío apoyándome, aconsejándome y guiándome, a mis profesores quienes pasaron 5 años de su tiempo, brindándonos su conocimiento para convertirnos en los profesionales de hoy, a mis amigos con quienes compartí cada día de clases, reímos, nos divertimos y sobre todo nos apoyamos.

Agradezco a mi tutor el ING Franklin Kuonqui, que nos ayuda desde el primer día y quién a pesar de no querer seguir nos impulsó a dar un último esfuerzo para alcanzar este nuevo objetivo.

Jorge Andres Navarrete Burgos

RESUMEN

El presente proyecto: DISEÑO DE UN CONTROLADOR INTELIGENTE PARA UN SISTEMA INESTABLE TIPO PÉNDULO INVERTIDO CON MONITOREO REMOTO POR IoT, consiste en el desarrollo de un controlador que pueda estabilizar el péndulo invertido, un sistema que naturalmente tiende a caer debido a su inestabilidad. El sistema está equipado con tecnologías IoT que permiten la supervisión y el control remotos desde cualquier lugar con conexión a Internet. Para el desarrollo del controlador se optó por un algoritmo de lógica difusa, el cual se encarga de tratar la información en este caso los ángulos para que a través de las reglas implementadas en la programación se puedan definir las acciones a realizar sobre los motores. Para el monitoreo remoto se usó el software de programación Node-red, el cual nos permite observar el ángulo medido por el MPU6050, el ciclo de trabajo de los motores, el sentido de giro, así como detener o empezar el programa cargado en el microcontrolador ESP32.

Palabras claves: ESP32, MPU5060, Fuzzy Logic, Node-red, Péndulo invertido.

ABSTRACT

The present project: DESIGN OF AN INTELLIGENT CONTROLLER FOR AN UNSTABLE INVERTED PENDULUM TYPE SYSTEM WITH REMOTE MONITORING BY IoT, consists of the development of a controller that can stabilize the inverted pendulum, a system that naturally tends to fall due to its instability. The system is equipped with IoT technologies that enable remote monitoring and control from anywhere with an internet connection. For the development of the controller, a fuzzy logic algorithm was chosen, which is responsible for processing the information, in this case the angles, so that through the rules implemented in the programming, the actions to be performed on the motors can be defined. For remote monitoring, the Node-red programming software was used, which allows us to observe the angle measured by the MPU6050, the duty cycle of the motors, the direction of rotation, as well as to stop or start the program loaded into the ESP32 microcontroller.

Keywords: ESP32, MPU5060, Fuzzy Logic, Node-red, Inverted pendulum

ÍNDICE

I. Introducción 1

II. Problema 2

III. Objetivos 3
III-A. Objetivo general . . . 3
III-B. Objetivos específicos . . . 3

IV. Marco Teórico Referencial 4
IV-A. Control . . . 5
IV-A1. Conceptos básicos de control . . . 5
IV-A2. Control proporcional (P) . . . 5
IV-A3. Control Integral (I) . . . 5
IV-A4. Control Derivativo (D) . . . 6
IV-A5. Acción de Control proporcional-integral (PI) . . . 6
IV-A6. Controladores PID . . . 6
IV-B. Lógica Difusa . . . 7
IV-B1. Concepto . . . 7
IV-B2. Funcionalidad . . . 7
IV-B3. Péndulo invertido con lógica difusa . . . 8
IV-B4. PID Difusa . . . 8
IV-C. Diferencia entre método Mamdani y Takagi sugeno . . . 8
IV-C1. Método Mamdani . . . 8
IV-C2. Método Takagi y Sugeno . . . 9
IV-C3. Mandani y Takagi sugeno . . . 9
IV-D. MPU6050 . . . 9
IV-D1. Concepto . . . 9
IV-D2. Característica del MPU6050 . . . 10
IV-E. Puente H . . . 11
IV-E1. Concepto . . . 11
IV-E2. Características L298N . . . 11
IV-F. ESP32 . . . 12
IV-F1. Concepto . . . 12
IV-F2. Características ESP32 . . . 12
IV-G. IoT . . . 13
IV-G1. Concepto . . . 13
IV-G2. Comunicación dispositivo a dispositivo . . . 13
IV-H. Regulador Step-Down XL4015 . . . 13

V. Marco Metodológico 14
V-A. Preparación del entorno para la programación del ESP32. . . . 14
V-B. Prueba de componentes . . . 16
V-C. Controlador difuso . . . 21
V-D. Instalación del Node.js . . . 21

VI. Resultados 23
VI-A. Montaje de la etapa electrónica . . . 23
VI-B. Control y monitoreo . . . 24

VII. Cronograma 29

VIII. Presupuesto 30

IX. Conclusiones 31

X. Recomendaciones 32

Referencias 33

Anexo A: MPU6050 36

Anexo B: PUENTE H 37

Anexo C: Codigo del controlador y sistema de monitoreo 39

Anexo D: Monitorio Node Red 60

ÍNDICE DE FIGURAS

1. Sistema de control realimentado, tomado de [13]. . . . 5
2. Salida del controlador P para entrada escalonada, tomado de [15]. . . . 5
3. Acción de control Integral, tomado de [17]. . . . 6
4. Control Derivativo, tomado de [19]. . . . 6
5. Control proporcional-integral, tomado de [21]. . . . 6
6. Controlador PID [23]. . . . 7
7. Lógica Difusa, tomado de [25]. . . . 7
8. Lógica Difusa, tomado de [27]. . . . 8
9. Estructura de un Controlador PID Difuso de acción directa, tomado de [28]. . . . 8
10. Método de inferencia de Mamdani, tomado de [30]. . . . 9
11. Takagi y Sugeno, tomado de [32]. . . . 9
12. Módulo Acelerómetro y giroscopio MPU6050, tomado de [35]. . . . 10
13. Módulo MPU6050, tomado de [36]. . . . 10
14. Puente H L298N, tomado de [40]. . . . 12
15. Módulo ESP32, tomado de [42]. . . . 12
16. Módulo ESP32 pinout, tomado de [44]. . . . 12
17. Ejemplo de un modelo de comunicación dispositivo a dispositivo, tomado de [47]. . . . 13
18. Convertidor Voltaje Fuente, tomado de [48]. . . . 13
19. URL, por C. Infante y J. Navarrete, Arduino, 2024. . . . 14
20. Instalación de paquete de la placa de desarrollo, por C. Infante y J. Navarrete, Arduino, 2024. . . . 15
21. Selección de placa y puerto para la comunicación, por C. Infante y J. Navarrete, Arduino, 2024. . . . 16
22. Conexiones ESP32-MPU6050, por C. Infante y J. Navarrete, Arduino, 2024. . . . 17
23. Filtrado de las librerías por C. Infante y J. Navarrete, Arduino, 2024. . . . 17
24. Código para extraer los angulos por C. Infante y J. Navarrete, Arduino, 2024. . . . 18
25. Controlador de motores por C. Infante y J. Navarrete, Arduino, 2024. . . . 19
26. Código de prueba del motor (1), por C. Infante y J. Navarrete, Arduino, 2024. . . . 20
27. Código de prueba del motor (2), por C. Infante y J. Navarrete, Arduino, 2024. . . . 20
28. Node.js C. Infante y J. Navarrete, Node. js, 2024. . . . 21
29. CMD instalación del Node Red C. Infante y J. Navarrete, Node red 2024. . . . 22
30. Dirección ip del node red, por C. Infante y J. Navarrete, Node red 2024. . . . 22
31. Node red C. Infante y J. Navarrete, Node red 2024. . . . 23
32. Circuito en baquelita perforada, por C. infante y J. Navarrete. . . . 23
33. Montaje del microprocesador, puente H y el MPU6050, por C. infante y J. Navarrete. . . . 24
34. Prueba de ángulo, por C. infante y J. Navarrete. . . . 25
35. Prueba de motores, por C. infante y J. Navarrete. . . . 26
36. Lógica difusa, por C. infante y J. Navarrete. . . . 27
37. Ángulo del robot, por C. infante y J. Navarrete. . . . 28
38. Velocidad del motor, por C. infante y J. Navarrete. . . . 28
39. Código para extraer los ángulos del MPU6050, por C. Infante y J. Navarrete. . . . 36
40. motores con L298N parte 1, por C. Infante y J. Navarrete. . . . 37
41. motores con L298N parte 2, por C. Infante y J. Navarrete. . . . 38
42. motores con L298N parte 3, por C. Infante y J. Navarrete. . . . 39
43. código parte 1, por C. infante y J. Navarrete. . . . 39
44. código parte 2, por C. infante y J. Navarrete.. . . . 39
45. código parte 3, por C. infante y J. Navarrete. . . . 40
46. código parte 4, por C. infante y J. Navarrete. . . . 40
47. código parte 5, por C. infante y J. Navarrete. . . . 40
48. código parte 6, por C. infante y J. Navarrete. . . . 41
49. código parte 7, por C. infante y J. Navarrete. . . . 41

50. código parte 8, por C. infante y J. Navarrete. 41
51. código parte 9, por C. infante y J. Navarrete. 42
52. código parte 10, por C. infante y J. Navarrete. 42
53. código parte 11, por C. infante y J. Navarrete. 42
54. código parte 12, por C. infante y J. Navarrete.. 43
55. código parte 13, por C. infante y J. Navarrete. 43
56. código parte 14, por C. infante y J. Navarrete. 44
57. código parte 15, por C. infante y J. Navarrete. 44
58. código parte 16, por C. infante y J. Navarrete. 44
59. código parte 17, por C. infante y J. Navarrete. 45
60. código parte 1, por C. infante y J. Navarrete. 45
61. código parte 19, por C. infante y J. Navarrete. 46
62. código parte 20, por C. infante y J. Navarrete. 47
63. código parte 1, por C. infante y J. Navarrete. 48
64. código parte 2, por C. infante y J. Navarrete. 49
65. código parte 3, por C. infante y J. Navarrete. 50
66. código parte 4, por C. infante y J. Navarrete.. 51
67. código parte 5, por C. infante y J. Navarrete. 52
68. código parte 6, por C. infante y J. Navarrete. 53
69. código parte 7, por C. infante y J. Navarrete. 54
70. código parte 8, por C. infante y J. Navarrete. 55
71. código parte 9, por C. infante y J. Navarrete. 56
72. código parte 10, por C. infante y J. Navarrete. 57
73. código parte 11, por C. infante y J. Navarrete. 58
74. código parte 12, por C. infante y J. Navarrete. 59
75. Descarga del sistema Node.js, por C. infante y J. Navarrete. 60
76. Ingresar el comando en cmd, por C. infante y J. Navarrete. 60
77. Comando Node red e Ip de acceso, por C. infante y J. Navarrete. 60
78. Plataforma Node red, por C. infante y J. Navarrete. 61
79. Plataforma de descargar del EMQX, por C. infante y J. Navarrete. 61
80. Instalación por cmd del EMQX, por C. infante y J. Navarrete. 61
81. Dirección ip para el acceso al navegador del EMQX, por C. infante y J. Navarrete. 62
82. Navegador del EMQX, por C. infante y J. Navarrete. 62
83. Navegador del EMQX, por C. infante y J. Navarrete. 63
84. Crear una tab nueva, por C. infante y J. Navarrete. 63
85. Colocar el nombre a nuestra base, por C. infante y J. Navarrete. 64
86. Crear una tab nueva, por C. infante y J. Navarrete. 64
87. Colocar el nombre a nuestra base, por C. infante y J. Navarrete. 65
88. configuración del localhost de mqtt, por C. infante y J. Navarrete. 66
89. Verificación de puerto port, por C. infante y J. Navarrete. 67
90. Configuración de la opción functión para mostrar los ángulos desde los código del ESP32, por C. infante y J. Navarrete. 67
91. Nombre del controlador balancin, por C. infante y J. Navarrete. 68
92. Accionar del encendido y apagado via remota al robot, por C. infante y J. Navarrete. 69
93. Comunicación del node red hacia el microprocesador ESP32, por C. infante y J. Navarrete. 69
94. Monitorio del robot mediante la node red, por C. infante y J. Navarrete. 70
95. Robot balancin, por C. infante y J. Navarrete. 70
96. Robot balancin, por C. infante y J. Navarrete. 71

ÍNDICE DE TABLAS

I. Configuración de Pines MPU6050 11
II. Conexiones ESP32-MPU6050, por C. Infante y J. Navarrete, Arduino, 2024. 16
III. Conexiones entre el ESP32, L298N y motor, por C. Infante y J. Navarrete, Arduino, 2024. 18
IV. Cronograma 29
V. Presupuesto 30

I. Introducción

Este trabajo nace con la idea de implementar un controlador para un equipo físico debido a que en la actualidad existen muchos proyectos que son limitados a simulaciones, donde se descartan características de los sistemas reales que cambian drásticamente los resultados, y generan deficiencia en los profesionales al no haberlas enfrentado. A su vez, al incorporar un sistema de monitoreo IoT con la finalidad de no limitarse simplemente al desarrollo del controlador, sino que también permite incorporarse al uso de nuevas tecnologías. Por otro lado, el péndulo invertido es uno de los sistemas que mas se utilizan en control debido a su inestabilidad, lo que los lleva a tener aplicaciones en diferentes áreas de la industria, razón por la cual es importante que se aplique a un sistema físico y no solamente a uno simulado.

Para este proyecto, se resalta el uso de microcontroladores debido a su facilidad para ser adquirido por su bajo costo y alta demanda. Así como, el uso de sensores que nos permiten la medición de ángulos y drivers de control para motores, como lo son el MPU6050 y el L298N respectivamente. Todo esto con el fin de que las personas en la actualidad puedan desarrollarse en la programación de microcontroladores, la cual es una alternativa de bajo costo frente al uso de controladores más robustos como son los controladores lógicos programables (PLC de sus siglas en inglés).

Para este proyecto, se optó por usar fuzzy logic, la cual trata con la imprecisión y la incertidumbre en los sistemas planteando niveles de pertenencia a un conjunto, es decir en lugar de tener valores binarios, esta nos da valores de verdad que pueden ir 0 a 1. Adicionalmente, este tipo de algoritmos son muy compatibles con la programación de microcontroladores.

El uso del software Node-red ayuda a introducirse en el mundo de la programación y desarrollo de aplicaciones mediante el método de bloques, proporcionando así una mayor facilidad de programación a aquellas personas que no se ven muy involucradas en el desarrolo de código para aplicaciones remotas. Por lo cual permite desarrollar el sistema de monitoreo remoto por IoT, destacando su importancia y posible aplicaciones en diversas industrias y campos.

Por último, este proyecto puede abrir la puerta al diseño de prácticas de laboratorio más actualizadas en temas como sistemas embebidos, control automático y automatización.

II. Problema

En la actualidad, la automatización y el monitoreo remoto son cruciales para el sector industrial, así como para empresas pequeñas, medianas y proyectos independientes. Sin embargo, su implementación requiere de una inversión significativa debido a la cantidad de equipos que se necesitan para su desarrollo, siendo los controladores lógicos programables (PLC) el método habitual de control y monitoreo más utilizado hoy en día.

Los PLC son sistemas potentes y confiables, pero conllevan costos altamente elevados debido no solamente al hardware sino también a la programación y configuración requerida. Existen diversas marcas encargadas de desarrollar estos dispositivos, los cuales pueden variar en costos y características desde los $250 USD, dependiendo del modelo y las características [1].

El desarrollo de un sistema de monitoreo tiene un costo inicial de $3.177,00 USD, con la plataforma industrial Edge. En el mercado existen diversos tipos de plataformas y costos para el desarrollo de estos sistemas, dependiendo de las necesidades de cada empresa. Por lo tanto, la implementación de estos sistemas de monitoreo involucra una inversión que puede llegar a ser elevada, debido a que los costos dependen tanto de la empresa que lo desarrolla como del alcance del sistema y de los equipos involucrados, sin dejar de lado los gastos operativos a largo plazo [2].

De la misma manera, el monitoreo remoto de sistemas inestables con PLC es un desafío económico importante, ya que el uso de módulos de comunicación y mano de obra especializada necesaria para su instalación, configuración y puesta en marcha aumentan aún más su costo de desarrollo. Debido a ello, la rentabilidad y la eficiencia operativa de las pequeñas y medianas empresas pueden verse afectadas negativamente por los gastos incurridos, el mantenimiento continuo y las mejoras a largo plazo.

Adicionalmente, existen opciones económicas y eficientes de monitoreo y control remoto disponibles a través de las tecnologías de IoT. Sin embargo, en la carrera de Mecatrónica en la UPS de Guayaquil se trabaja muy poco en el desarrollo de este tipo de soluciones con IoT utilizando microcontroladores de bajo costo.

III. Objetivos

III-A. Objetivo general

Diseñar un controlador inteligente para un péndulo invertido utilizando dispositivos electrónicos de bajo costo y software libre para eliminar la dependencia de software privativo.

III-B. Objetivos específicos

- Diseñar las etapas de control y monitoreo usando software libre.
- Implementar las etapas de control y monitoreo para su validación.
- Validar el sistema mediante pruebas de campo.

IV. Marco Teórico Referencial

Debido a más de 20 años de avances tecnológicos y desarrollos en métodos de control e implementación de controladores PID, la mayoría de los artículos han utilizado motores sin escobillas, pero han seguido centrándose únicamente en el funcionamiento del motor. Aquí hay una explicación mínima, rudimentaria y básica de los controles [3].

Considerando los avances en el campo informática en los últimos años, se puede analizar su desarrollo histórico, el surgimiento del Internet de las Cosas (llamado IoT) y su gran aporte a otro tipo de tecnologías, como la computación en la nube de servidores, donde se puede procesar datos en tiempo real y analizarlos para tomar decisiones más eficientemente [4].

En aplicaciones basadas en la práctica del péndulo invertido se puede destacar su utilidad en varios ámbitos, como en la arquitectura donde pueden existir grandes edificios con grandes dimensiones, que son modelados como péndulos invertidos. Por otro lado tampoco dista mucho de la área robótica porque se utiliza este modelo para implementar sistemas de equilibrio, proporcionar sistemas de apoyo a pacientes con movilidad reducida o utilizar drones para transportar objetos, etc. [5].

El aspecto más complicado de trabajar en un proyecto como este, es sintonizar correctamente las ganancias del PID. Para alcanzar este objetivo es común utilizar el método Ziegler-Nichols, donde es posible obtener las ganancias aproximadas del PID aproximados y control total sin ajuste fino. Se realizan pruebas para obtener avances y errores, en este caso revisión [6].

El concepto de Internet de las Cosas (IoT) describe los parámetros y características más importantes para asegurar la integración de la conectividad, permitiendo el monitoreo con tecnologías económicas disponibles. El software tiene una buena función de interconexión con el sistema, lo que facilita a los usuarios acceder al sistema en cualquier momento y en cualquier lugar a través de Internet [7].

El desafío en este proyecto es mantener el péndulo en equilibrio en posición vertical cuando está es inestable implica controlar un sistema dinámico que toma una posición estable cuando el péndulo está cayendo o está en posición horizontal, pero una posición inestable cuando el péndulo está en posición vertical, mientras intenta mantener una posición vertical. En términos de equilibrio inestable, se refiere a una situación en la que cualquier alteración desaloja al cuerpo de un estado de equilibrio y sólo puede restablecerse mediante una intervención externa [8].

El diseño básico de un péndulo invertido, consta de una parte fija y otra móvil. Su función principal es mantener la estabilidad del coche, entendiendo estabilidad como la capacidad de mantener el equilibrio o volver a tal estado tras una perturbación. Sin un sistema de control, el mecanismo se alejaría del punto de equilibrio debido a la gravedad y la caída. Estamos ante un sistema no lineal inestable [9].

Los sistemas de péndulo invertido se destacan por ser una herramienta que simula sistemas dinámicos utilizados en varios campos de la industria actual, como la robótica, la biomecánica, la aeroespacial y algunos medios de transporte. Para aprovechar este sistema de manera efectiva, es fundamental contar con una herramienta que permita controlar las diversas variables que influyen en el movimiento del sistema [10].

En los últimos años, los robots auto equilibrados han despertado un considerable interés por parte de estudiantes, ingenieros e investigadores. Básicamente, estos robots parecen un péndulo invertido montado sobre dos ruedas. A diferencia de otros robots con ruedas, que generalmente son estables, el equilibrio de estos robots es inherentemente inestable. El complejo diseño de estos robots requiere un control constante para que permanezcan erguidos y respondan rápidamente y sin dudar a las perturbaciones externas [11].

IV-A. Control

IV-A1. Conceptos básicos de control:

- Cada proceso tiene una dinámica única que lo distingue de otros procesos, similar a la personalidad, la huella dactilar o el ADN de cada persona. Por tanto, es necesario estudiar, probar y comprender la ”personalidad”del proceso controlado ajustando los algoritmos P (proporcional), I (integral) y D (derivado) del controlador del circuito de control. Además de ajustar los parámetros del algoritmo de control, se deben medir, calibrar y mantener varias variables del proceso [12].

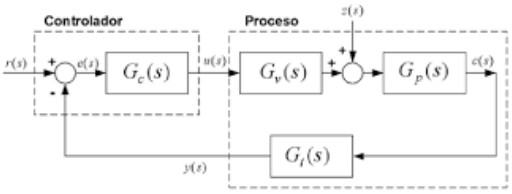


Figura 1. Sistema de control realimentado, tomado de [13].

IV-A2. Control proporcional (P):

- Los controladores solo P reducen la variación en las variables del proceso pero no necesariamente mantienen el sistema en el punto de ajuste deseado. Esto proporciona una respuesta más rápida que la mayoría de los otros controladores, lo que resulta en una respuesta inicial más rápida del controlador en segundos. Sin embargo, a medida que los sistemas se vuelven más complejos, las diferencias en el tiempo de respuesta pueden acumularse e incluso hacer que el controlador responda varios minutos más rápido. Aunque este controlador tiene la ventaja de una respuesta más rápida, también puede provocar una desviación del punto de ajuste [14].

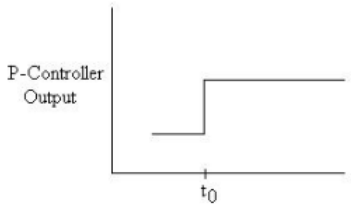


Figura 2. Salida del controlador P para entrada escalonada, tomado de [15].

IV-A3. Control Integral (I):

- La función de este control, como su nombre indica, es integrar la señal de error e(t) y multiplicarla por una constante Ki. Integral puede entenderse como la suma o acumulación de la señal de error. Con el tiempo, se acumularon pequeños errores que complicaron las operaciones integradas. Esto ayuda a reducir los errores del sistema de estado estable [16].

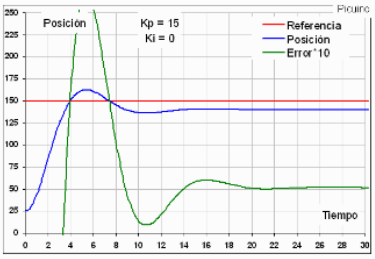


Figura 3. Acción de control Integral, tomado de [17].

IV-A4. Control Derivativo (D):

- Entre las limitaciones de control de los derivados, la más notable es su baja popularidad en la industria debido a las altas preocupaciones regulatorias. Por lo tanto, normalmente sólo se utiliza cuando el objetivo es predecir algún efecto futuro en el sistema. Vale la pena señalar que si el tiempo de derivación es bastante largo, significa que la expectativa de mal comportamiento en el futuro es muy larga, lo que puede conducir a predicciones inexactas [18].

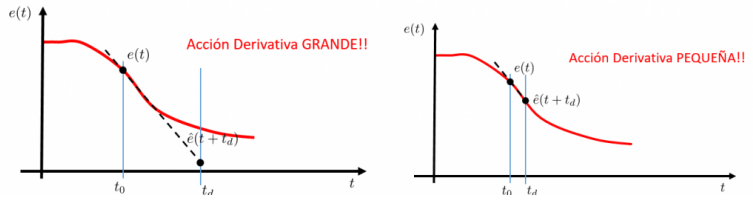


Figura 4. Control Derivativo, tomado de [19].

IV-A5. Acción de Control proporcional-integral (PI):

- El control proporcional requiere un error para producir una acción de control distinta de cero. Por otro lado, incluso pequeños errores no positivos en la acción integral siempre harán que el control aumente, y si el error es negativo, la señal de control disminuirá. Este sencillo razonamiento nos dice que el error en estado estacionario siempre será cero cuando la referencia es constante o de tipo escalón. Muchos controladores industriales realizan únicamente operaciones PI. Se puede demostrar que el controlador PI es adecuado para todos los procesos cuya dinámica sea principalmente de primer orden [20].

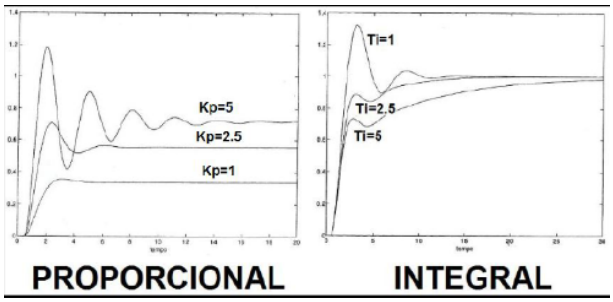


Figura 5. Control proporcional-integral, tomado de [21].

IV-A6. Controladores PID:

- El controlador PID es una parte importante del sistema de control industrial y es una herramienta de control de retroalimentación de circuito cerrado ampliamente utilizada. Su función principal es corregir la diferencia

entre el valor medido real de la variable de proceso y el valor de referencia requerido. Esto se hace calculando e implementando acciones correctivas, llamadas acciones de control, que cambian el proceso hasta que se minimicen los errores. El proceso de cálculo de un controlador PID incluye tres parámetros diferentes: un valor proporcional que afecta la respuesta ante la falla actual; una integral que afecta la operación con base en el efecto acumulativo de errores anteriores; y derivación, que predice las acciones necesarias para cambiar el comportamiento de salida del proceso [22].

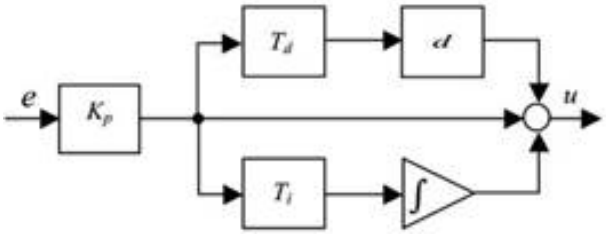


Figura 6. Controlador PID [23].

IV-B. Lógica Difusa

IV-B1. Concepto:

- La lógica difusa es una ampliación de la lógica clásica que nos ayuda a controlar valores difusos e imprecisos. En cambio la lógica clásica, dado el valor de verdad sólo puede ser positivo o negativo, sea cualquier número en el intervalo [0, 1]. Esto le permite utilizar valores de verdad distribuidos e ingresar grados de certeza a las declaraciones. La idea básica detrás de la lógica difusa es que, en cualquier situación del mundo real, la verdad seguira siendo nula, donde depende de múltiples factores y circunstancias. Absolutamente, pero depende de varios factores y circunstancias diferentes [24].

Figura 7. Lógica Difusa, tomado de [25].

IV-B2. Funcionalidad:

- Es la disciplina que nos permite lidiar con información inexacta. Veamos un ejemplo. La temperatura se puede medir en grados Celsius. Es decir, podemos decir 16 grados y medio, pero también podemos medirlo de forma imprecisa. Por ejemplo, “un poco de frío” o “un poco de calor”. Esto significa que debemos determinar qué valores son fríos, calientes, etc. [26].

IV-B3. Péndulo invertido con lógica difusa:

- Se proponen controladores basados en lógica difusa como una alternativa para superar las dificultades de modelado, ya que tienen la ventaja de que su diseño no requiere de un modelo matemático del sistema controlado. Estos controles toman expresiones lingüísticas proporcionadas por expertos con amplio conocimiento del comportamiento de dichas expresiones lingüísticas y las traducen al lenguaje matemático y se basan en ellas utilizando lógica difusa. Derivar lógicamente la posible señal de control que permita que la variable controlada alcance el estado deseado [27].

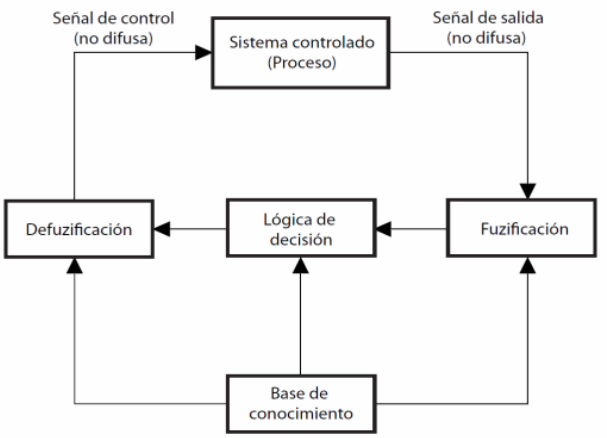


Figura 8. Lógica Difusa, tomado de [27].

IV-B4. PID Difusa :

- Para desarrollar el controlador PID difuso se definió la estructura y se obtuvieron los parámetros que caracterizan la operación del proceso realizando una simulación con un alto grado de certeza con el proceso real. Después de encontrar los parámetros que caracterizan la operación del proceso, se realiza la simulación del proceso utilizando PID tradicional. A partir de esto, se puede obtener el comportamiento dinámico del error, el margen de error y la salida y luego se puede definir la base de reglas del controlador difuso [28].

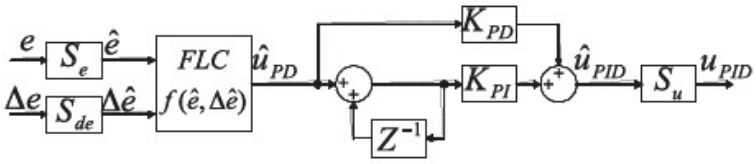


Figura 9. Estructura de un Controlador PID Difuso de acción directa, tomado de [28].

IV-C. Diferencia entre método Mamdani y Takagi sugeno

IV-C1. Método Mamdani:

- El método de Mamdani utiliza los grados de pertenencia de las variables de entrada obtenidos de las pruebas de fugas y los aplica a las reglas de la base de conocimientos para determinar sus grados de pertenencia en el conjunto de variables de salida. Utilice la membresía de salida para crear un polígono que represente los valores de membresía del conjunto de salida a lo largo de la variable de salida [29].

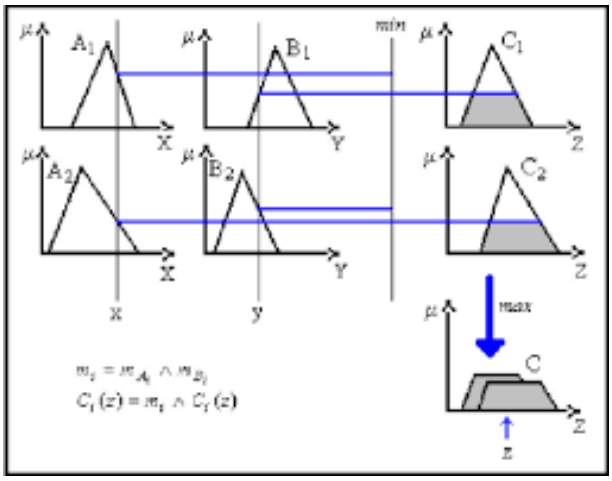


Figura 10. Método de inferencia de Mamdani, tomado de [30].

IV-C2. Método Takagi y Sugeno:

- El modelo difuso de Takagi y Sugeno (1985) nos permite aproximar el comportamiento de un sistema no lineal combinando las características de un sistema lineal, donde el subsistema lineal es parte de la salida de la regla y representa el comportamiento del sistema cerca de un punto definido operación. punto [31].

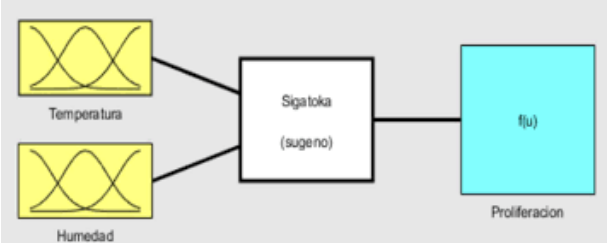


Figura 11. Takagi y Sugeno, tomado de [32].

IV-C3. Mandani y Takagi sugeno:

- La principal diferencia entre el método TSK y el método Mamdani es que no es necesario realizar el proceso de defusificación. Esto se debe a que no obtenemos un conjunto difuso, sino un conjunto de funciones lineales. Por tanto, en el método TSK, podemos obtener directamente el valor de salida del sistema utilizando los siguientes tipos de expresiones [33].

$$Z_0 = \frac{\sum_{i=1}^{n} w_i f_i(x_i y_i)}{\sum_{i=1}^{n} w_i} \tag{1}$$

IV-D. MPU6050

IV-D1. Concepto:

- El MPU6050 es un sensor de medida inercial, también conocido como IMU (unidad de medida inercial), con 6 grados de libertad (DoF) ya que incorpora un acelerómetro de 3 ejes y un giroscopio de 3 ejes. Este dispositivo es ampliamente utilizado en diversas aplicaciones como navegación, medición de ángulos, estabilidad, etc. Los giroscopios utilizan sistemas MEMS (sistemas microelectromecánicos) que utilizan el efecto Coriolis para determinar la velocidad angular. También se puede medir la velocidad angular y, si se

integra en función del tiempo, se puede calcular el desplazamiento angular (o la posición angular si se conoce la posición inicial de la rotación) [34].

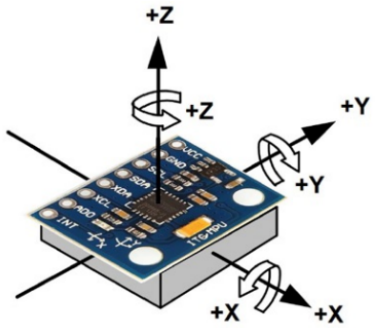


Figura 12. Módulo Acelerómetro y giroscopio MPU6050, tomado de [35].

IV-D2. Característica del MPU6050:

- Se comunica a través de una interfaz I2C y existen bibliotecas muy utilizadas para facilitar su integración inmediata. Este sensor proporciona 6 grados de libertad y tiene un regulador incorporado de 3,3 V y una resistencia pull-up para conexión directa a través de I2C. La biblioteca i2cdevlib se utiliza para la integración con Arduino. Su conexión se simplifica a través de una interfaz host I2C, que permite controlar sensores externos adicionales como magnetómetros o barómetros sin la intervención del procesador principal, ayudando así a optimizar los recursos disponibles [36].

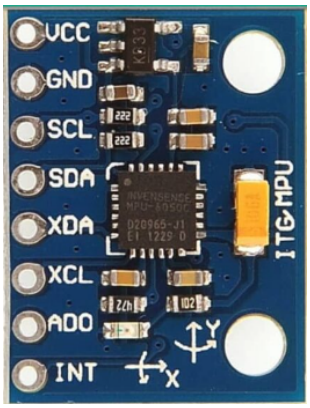


Figura 13. Módulo MPU6050, tomado de [36].

Tabla I
CONFIGURACIÓN DE PINES MPU6050

VCC	Pin de fuente de alimentación (Conecte este pin al suministro de +5V CC.)
GND	Clavija de tierra. Conecte este pin a la conexión a tierra
INT	Interrumpe el pin de salida digital
XDA	Pin de datos serie auxiliar. Este pin se utiliza para conectar el pin SDA de otros sensores habilitados para interfaz I2c al MPU-6050
SCL	Pin de reloj serie auxiliar. Conecte este pin al pin SCL del microcontrolador
XCL	Pin de reloj serie auxiliar. Este pin se utiliza para conectar el pin SCL de otros sensores habilitados para interfaz 12C al MPU-6050
SDA	Pin de datos serie. Conecte este pin al pin SDA del microcontrolador.
ADO	Pin LSB de dirección esclava I2C. Este es el bit 0 en la dirección esclava de 7 bits del dispositivo. Si está conectado a VCC, se lee omo uno lógicos y la dirección de esclavo cambia.

IV-E. Puente H

IV-E1. Concepto:

- Un puente H es un dispositivo que ayuda a controlar la dirección de rotación y manejo de corriente de un motor. Existen varios controladores equipados con circuitos integrados para cambiar la rotación del motor. Estos controladores accionan dos motores paso a paso de CC y proporcionan pulsos de hasta 2 amperios. En el mercado se encuentran disponibles una variedad de puentes en H con juntas soldadas para conectar cables y conducir la entrada y salida de señales [37].

  El controlador de puente H L298N es el módulo más utilizado y puede controlar motores de CC de hasta 2 amperios. El chip L298N tiene dos puentes H completos que pueden accionar 2 motores de CC o motores paso a paso bipolares/unipolares. Este módulo le permite controlar la dirección y velocidad de rotación del motor utilizando señales TTL disponibles de microcontroladores y placas de desarrollo como Arduino, Raspberry Pi o Texas Instruments Launchpad. Cada motor controla la dirección de rotación mediante dos pines y se puede utilizar la modulación de ancho de pulso (PWM) para ajustar la velocidad de rotación. [38].

IV-E2. Características L298N:

- Las principales características de los puentes H se centran en su capacidad para controlar el movimiento de motores eléctricos, proporcionando varias funciones que los hacen indispensables en proyectos de electrónica y robótica. Este dispositivo proporciona la capacidad de cambiar la dirección de rotación del motor, lo cual es esencial para lograr un control de movimiento bidireccional [39].

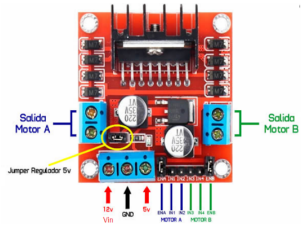


Figura 14. Puente H L298N, tomado de [40].

IV-F. ESP32

IV-F1. Concepto:

- El módulo ESP32 a utilizar es un chip que integra un procesador de 32 bits. WIFI, comunicación Bluetooth, sensores táctiles capacitivos, sensores de efecto Hall, Ethernet, SPI, UART, I2S e I2C. Diseñado para aplicaciones móviles, electrónica e Internet de las cosas, con bajo consumo de energía por su precio, contiene una variedad de software, lenguajes de programación, bibliotecas, códigos, frameworks y otros recursos que permiten desarrollar proyectos [41].

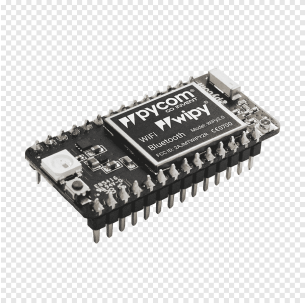


Figura 15. Módulo ESP32, tomado de [42].

IV-F2. Características ESP32:

- El ESP32 tiene interruptores de antena altamente integrados, balun RF, amplificadores de potencia, amplificadores de ganancia de bajo ruido, filtros y módulos de administración de energía, ¡todo integrado en un solo chip adicional tiene el paquete 1x NodeMcu ESP-32S-WROOM Bluetooth + WiFi de 38 Pines USB-C [43].

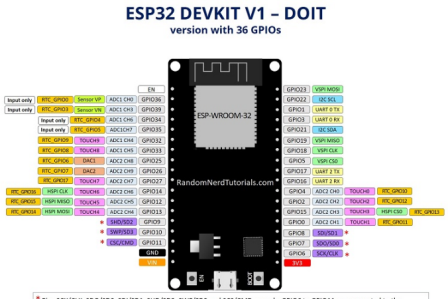


Figura 16. Módulo ESP32 pinout, tomado de [44].

IV-G. IoT

IV-G1. Concepto:

- La traducción literal del término Internet de las Cosas (a partir de ahora IoT) es igual que el término Ïnternet de las cosas", esta traducción es claramente insuficiente para definir su alcance y complejidad. La tecnología también es una infraestructura de red dinámica global, opción de autoconfiguración basada en protocolo de comunicación las cosas físicas y virtuales que están estandarizadas e interoperables, tiene identidad, características físicas, realización y uso de interfaces los dispositivos inteligentes que se integran perfectamente en las redes de información [45].

IV-G2. Comunicación dispositivo a dispositivo :

- Modelo de comunicación del dispositivo a dispositivo se refiere a dos o más dispositivos que se conectan y comunican directamente entre sí y no a través de un servidor de aplicaciones un intermediario. Estos dispositivos se comunican mediante diferentes tipos de redes, incluidas redes IP o Internet. A menudo se utilizan protocolos como Bluetooth para establecer una comunicación directa entre dispositivos [46].

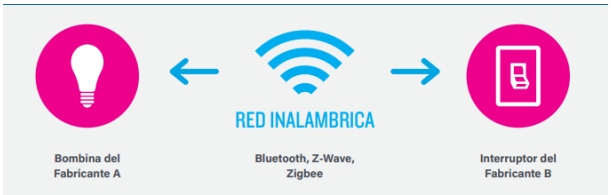


Figura 17. Ejemplo de un modelo de comunicación dispositivo a dispositivo, tomado de [47].

IV-H. Regulador Step-Down XL4015

- La función del regulador es entregar un voltaje de salida constante más bajo que el de entrada. Soporta corrientes de salida de hasta 5A con un voltaje de entrada de 4 a 36V y un voltaje de salida entre 1,25V a 32V. La tensión de salida se puede ajustar mediante el potenciómetro multivuelta que lleva integrado el regulador.

  Por ejemplo, si desea obtener 5 V, 3,3 V, 1,8 V de una fuente de 12 V, puede utilizar este regulador. Surgirán problemas de eficiencia y rendimiento si los voltajes de entrada y salida no superan los 1,5 V. Cuando se utiliza para una corriente superior a 3,5 A es recomendable añadir un disipador de calor [48].

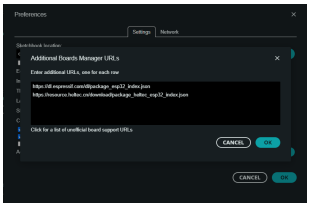


Figura 18. Convertidor Voltaje Fuente, tomado de [48].

V. Marco Metodológico

En este apartado se definen los pasos realizados para el desarrollo del algoritmo de control de los motores del balancín, así como la infraestructura electrónica y el programa de monitoreo considerado en el proyecto propuesto. Para programar la placa de desarrollo NODEMCU ESP-32S, en primer lugar se debe configurar el entorno Arduino IDE para que se pueda trasnferir el programa a la placa. Para este fin se debe agregar la librería de la placa a utilizar.

V-A. Preparación del entorno para la programación del ESP32.

Para preparar el entorno de programación, primero se debe abrir el software Arduino IDE y seleccionar "File → Preferences". Una vez realizado esto se deberá dar clic en el botón "Additional Boards Manager URLs" y se debe ingresar las siguientes URLs una por cada fila como se muestra en la figura 19:

- https://dl.espressif.com/dl/package_esp32_index.json
- https://resource.heltec.cn/download/package_heltec_esp32_index.json

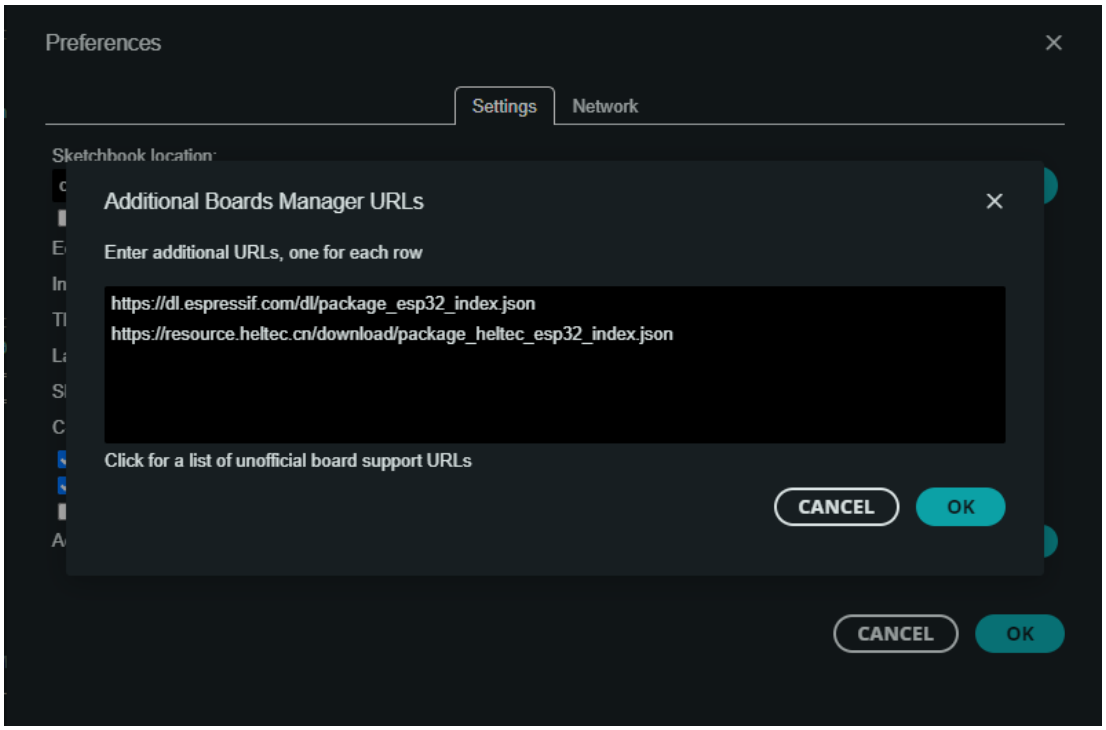


Figura 19. URL, por C. Infante y J. Navarrete, Arduino, 2024.

Una vez agregadas las URLs, es necesario dar clic en el botón "OK" de la ventana, para regresar a la ventana de preferencias, donde también se debe hacer clic en el botón "OK". A continución, se debe seguir la ruta para instalar el soporte del módulo ESP32 y las placas de desarrollo. Para esto, hay que dirigirse a "Tools→Board→Boards Manager", y se debe despliega la ventana "Boards Manager". Luego, se escribe "ESP32" en la barra de búsqueda, como se muestra en la figura 20, para filtrar las placas disponibles e instalar el paquete que contenga la placa de desarrollo a utilizar.

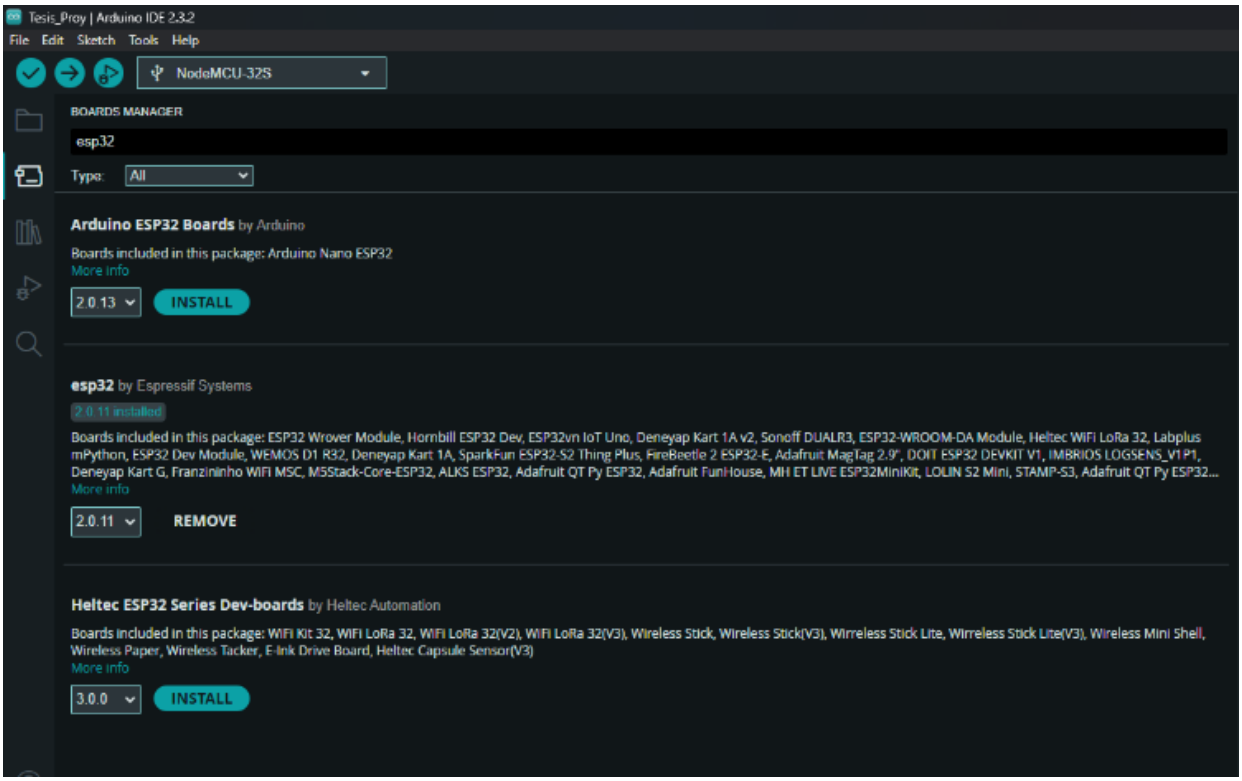


Figura 20. Instalación de paquete de la placa de desarrollo, por C. Infante y J. Navarrete, Arduino, 2024.

El siguiente paso es comunicar el ESP32 con el software Arduino IDE. Para esto se debe utilizar un cable USB a micro USB. Una vez conectado al ordenador, debe encender el LED de la placa, indicando que la misma esta correctamente energizada. Luego se debe seleccionar la placa a utilizarse y el puerto en el que se encuentra conectada la misma, el cual muchas veces es determinado por el propio IDE pero en otras ocasiones es necesario especificarlo, como se muestra en la figura 21.

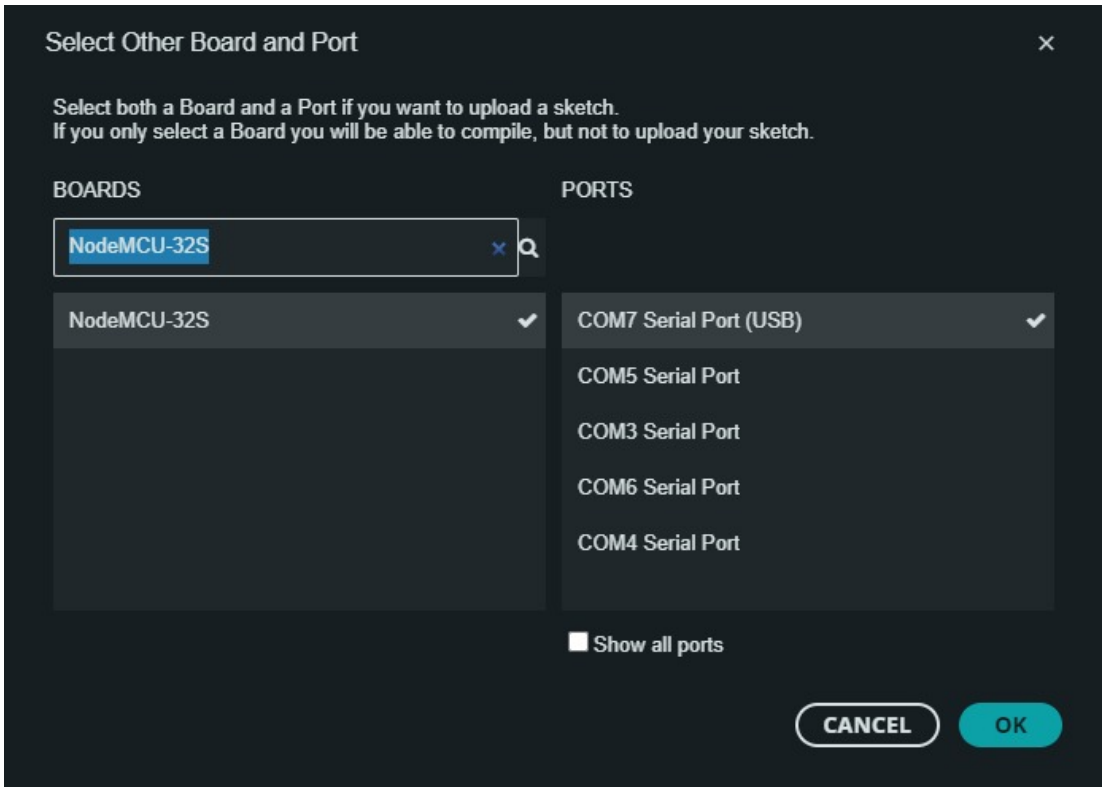


Figura 21. Selección de placa y puerto para la comunicación, por C. Infante y J. Navarrete, Arduino, 2024.

V-B. Prueba de componentes .

Para la prueba de los componentes, estos se dividen en dos grupos, El primer grupo uno compuesto por motores, ruedas y el driver L298N y mientras que el grupo dos solo tiene el giroscopio MPU6050.

El MPU6050 es un módulo sensor que gracias a su comunicación I2C permite la interacción con un Arduino o cualquier otro microcontrolador a través del reloj serial (SCL) y datos (SDA), que debe ser conectado como se muestra en la figura 22 y se detalla en la tabla II.

Tabla II
CONEXIONES ESP32-MPU6050, POR C. INFANTE Y J. NAVARRETE, ARDUINO, 2024.

PIN MPU6050	PIN ESP32
SCL	36
SDA	33
VCC	1
GND	GND

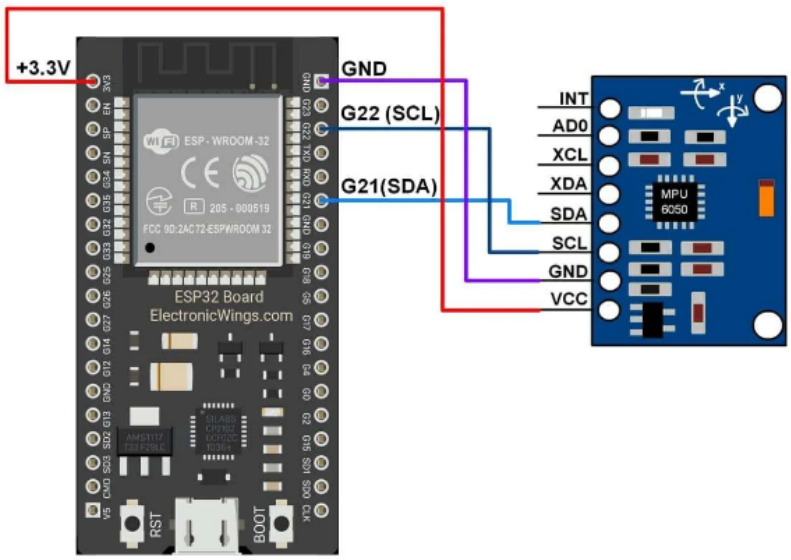


Figura 22. Conexiones ESP32-MPU6050, por C. Infante y J. Navarrete, Arduino, 2024.

Para la comunicación entre el módulo ESP32 y el MPU6050 es necesario la instalación de la librería que permite la interacción entre ellos. En el software de Arduino IDE se va al apartado de "Tools→Manage Libraries", y se despliega la ventana Library Manager. Luego, en la barra de búsqueda, se ingresa "MPU6050" para filtrar las librerías y se instala el paquete que contenga la librería a utilizar para el módulo sensor como se muestra en la figura 23.

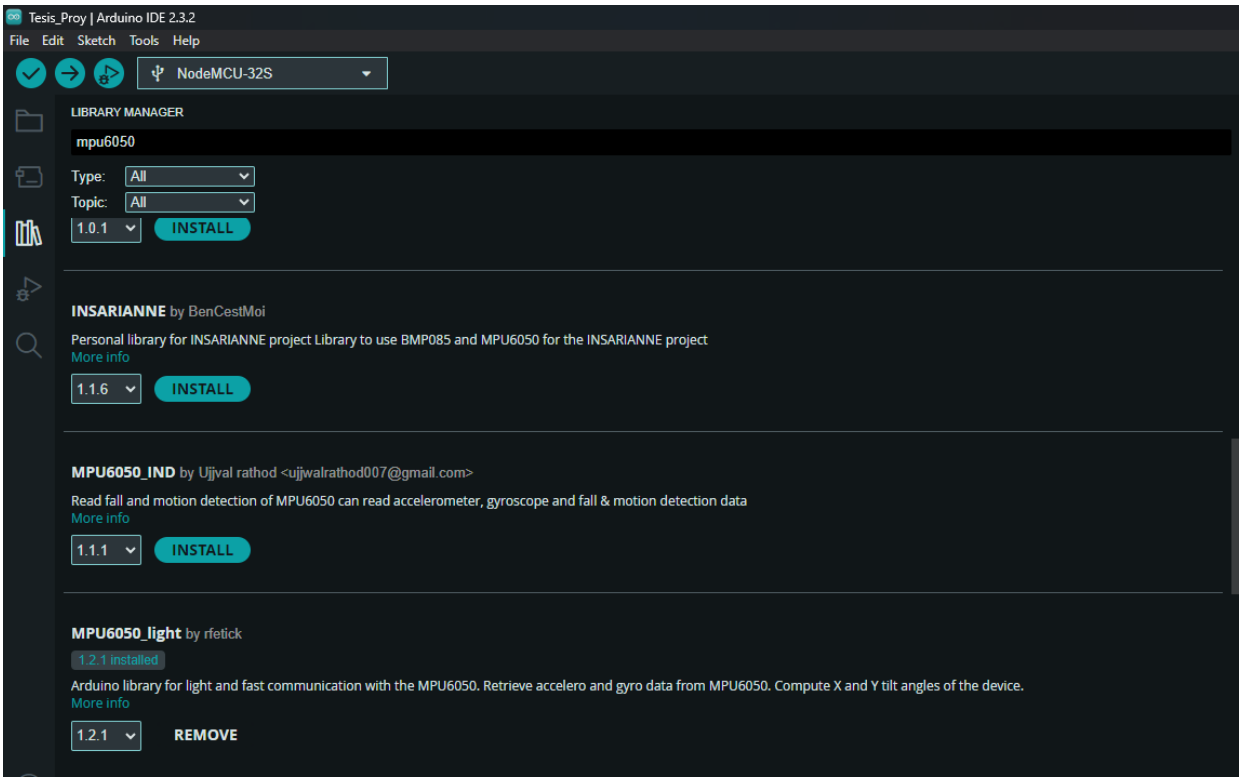


Figura 23. Filtrado de las librerías por C. Infante y J. Navarrete, Arduino, 2024.

Para extraer los ángulos medidos por el MPU6050, se utilizó el código mostrado en la figura 24, en el cual invoca las librerías a utilizar para calibrar el sensor. A continuación, el código actualiza los datos, y los mismos son impresos cada determinado tiempo e indicando su respectivo eje (X, Y, Z) comprobando así que el sensor se encuentre en buen estado para su uso.

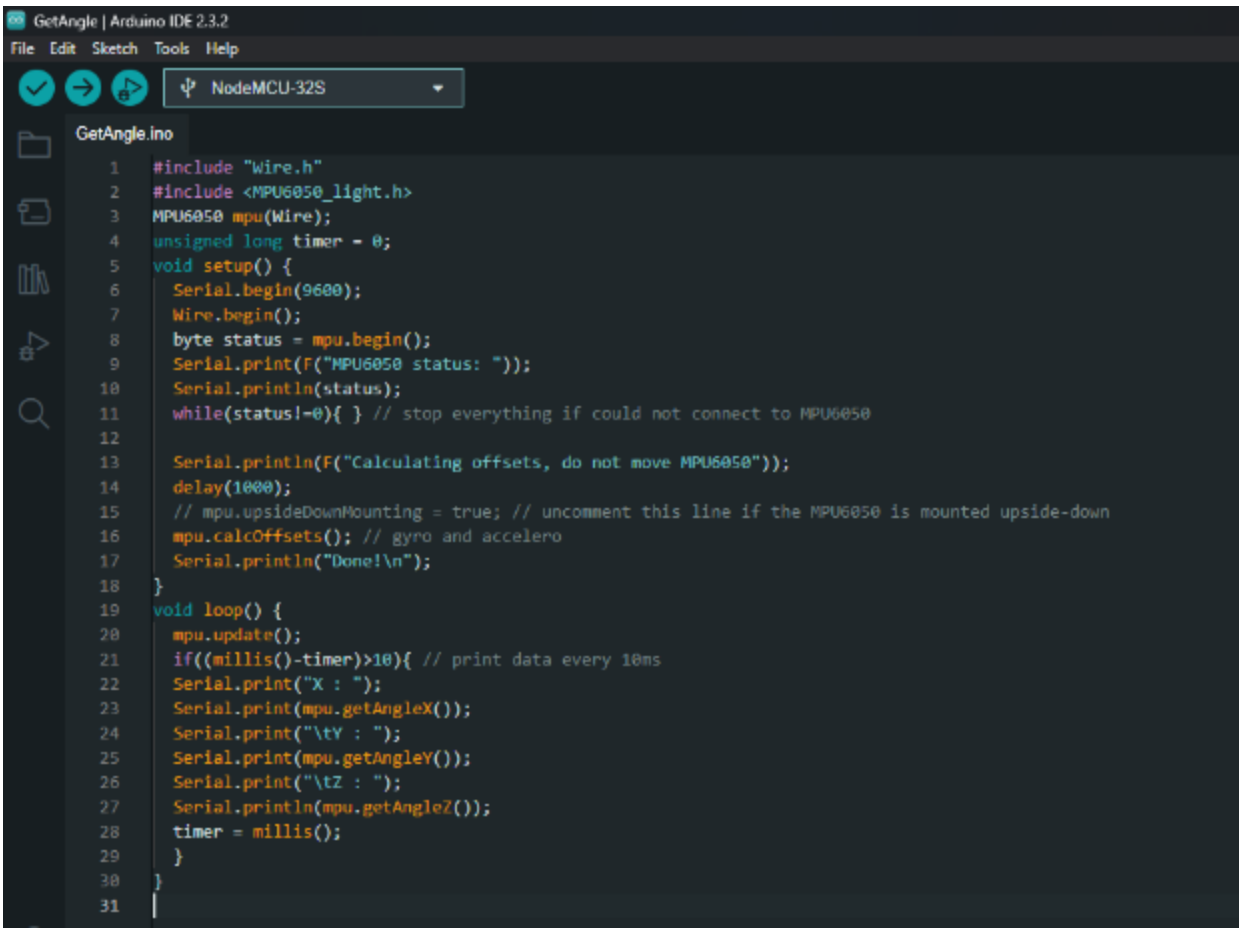


Figura 24. Código para extraer los angulos por C. Infante y J. Navarrete, Arduino, 2024.

A continuación, se prueba el sistema de motricidad empezando por el driver L298N que permite el control del giro y la velocidad de motores DC, siendo de gran ayuda para el desarrollo de este proyecto. La conexiones entre el ESP32, el driver y los motores se pueden observar en la figura 25 y se detallan en la tabla III.

Tabla III
CONEXIONES ENTRE EL ESP32, L298N Y MOTOR, POR C. INFANTE Y J. NAVARRETE, ARDUINO, 2024.

Pin L298N	Pin llegada	Elemento
ENA, ENB	12	ESP32
IN1, IN2	11	ESP32
IN3, IN4	10	ESP32
OUT1, OUT3	1 -	Motor
OUT2,OUT4	2 +	Motor

El driver L298N puede controlar 2 motores, a través de las salidas OUT1/OUT2 y OUT3/OUT4. La polarización y el ciclo de trabajo de las salidas le dan el sentido de giro y la velocidad a cada motor. Para la manipulación de las características mencionadas en el motor conectado a las salidas OUT1/OUT2, el driver tiene las entradas ENA,

IN1, IN2. La velocidad es controlada por el tren de pulsos con su respectivo ciclo de trabajo recibido en la entrada ENA, mientras que los valores lógicos de las entradas IN1/IN2 definen el sentido de giro. De manera similar, el motor conectado a las salidas OUT3/OUT4 es manipulado con las entradas ENB, IN3, IN4.

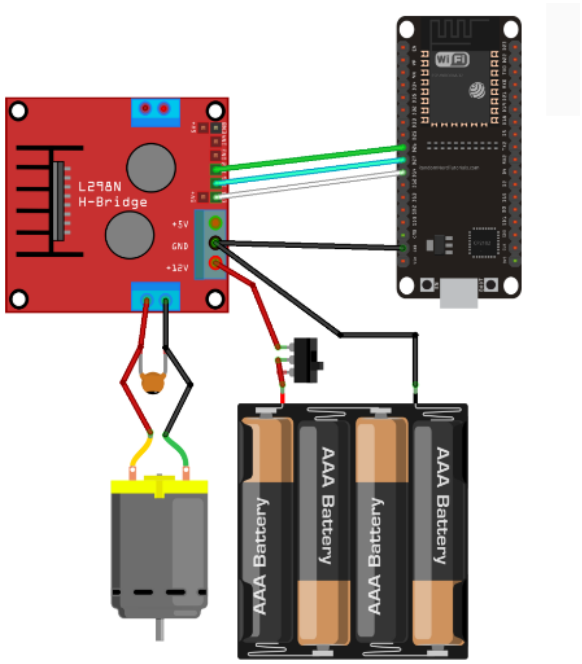


Figura 25. Controlador de motores por C. Infante y J. Navarrete, Arduino, 2024.

Para comprobar el funcionamiento del driver y los motores, se utiliza el siguiente código presentado en las figuras 26 y 27, el cual permite controlar la velocidad y sentido de giro de los motores. Esto demuestra que el L298N es un componente clave para el desarrollo del proyecto.

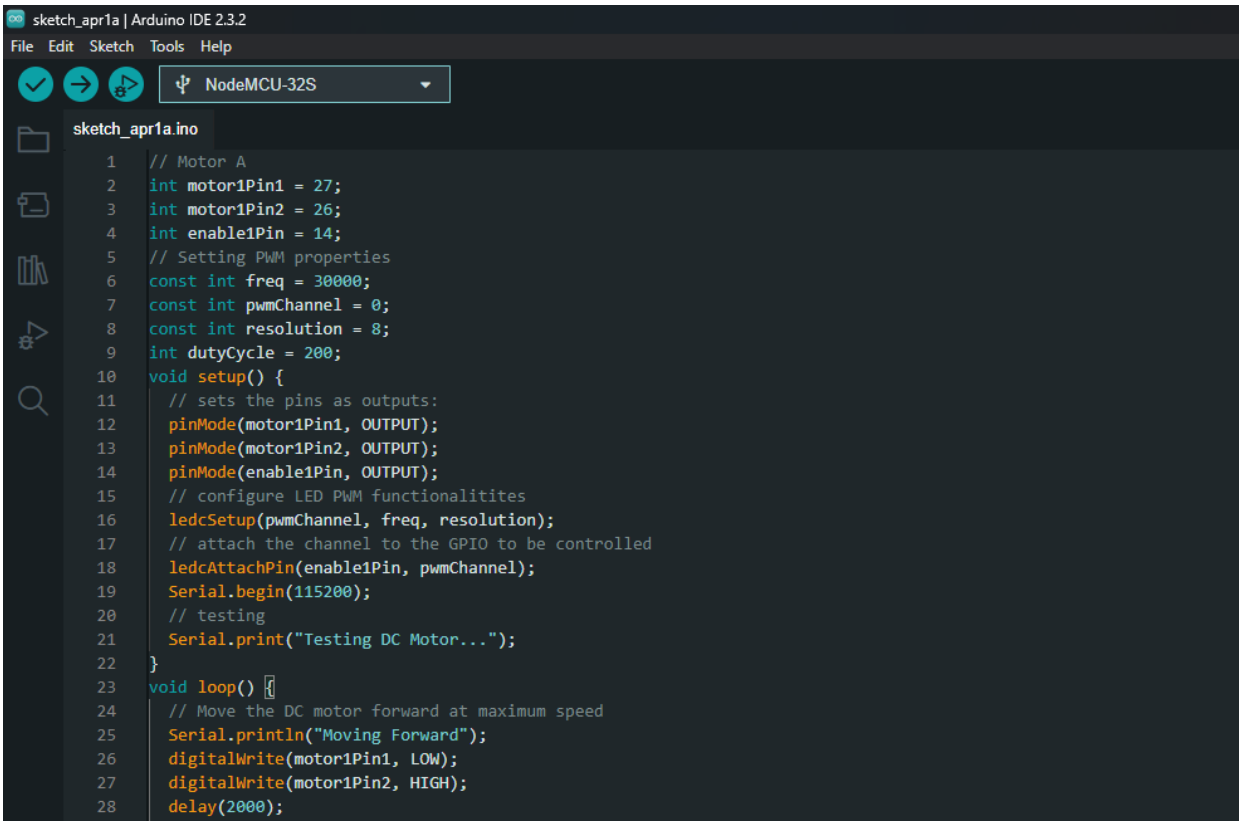


Figura 26. Código de prueba del motor (1), por C. Infante y J. Navarrete, Arduino, 2024.

Figura 27. Código de prueba del motor (2), por C. Infante y J. Navarrete, Arduino, 2024.

V-C. Controlador difuso

Para el desarrollo del controlador difuso, se requiere de la adaptación de los programas de lectura del giroscopio, de manipulación de velocidad y sentido de giro de los motores; así como también, de la implementación de funciones de membresía, reglas de implicación, fuzificación y defuzificación con la matemática asociada.

En primer lugar, se corrigió el código de la lectura de ángulos para trabajar únicamente con el eje asociado al movimiento de la estructura.

Luego, se adaptó el código de control de rotación de las ruedas para eliminar la zona muerta en la que no giran los motores a pesar de tener un ciclo de trabajo distinto de 0.

A continuación, se implementó el cálculo del error y su derivada considerando que la referencia es un ángulo de 0^o. A partir de este cálculo, se realiza la fuzificación considerando tres funciones de membresía de tipo triangular para cada variable, siendo sus valores lingüísticos: negativo, cero y positivo.

Adicionalmente, la variable de salida se la definió con el nombre mando, y tenía tres valores lingúísticos idénticos a los de las entradas pero en este caso al ser un sistema Takagi-Sugeno solo tenían valores puntuales sus funciones de membresía.

Las reglas de implicación se realizaron asociando cada combinación posible de las funciones de membresía de las entradas con una función de membresía de la salida que procure reducir el error y su derivada.

Todo lo expresado se puede observar en las figuras presentadas en el anexo C.

V-D. Instalación del Node.js

Se procede el Node.js y buscarla versión v20.12.1(LTS) en la página web https://nodejs.org/en/download/current mostrada en la figura 28 el cual nos permite tener acceso al node red.

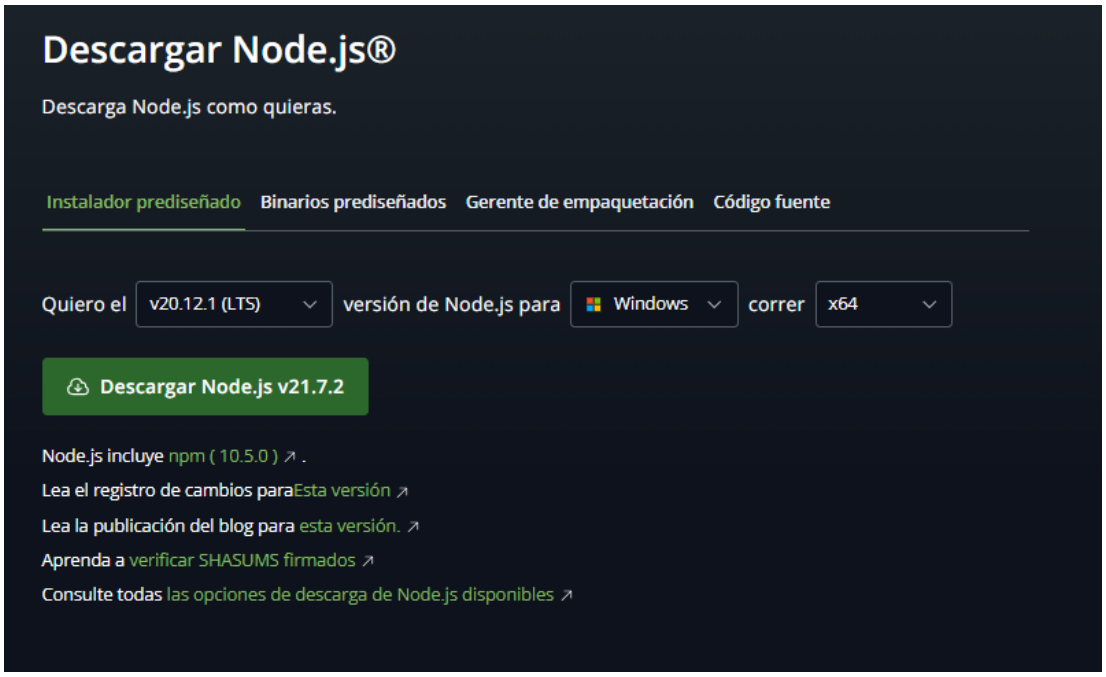


Figura 28. Node.js C. Infante y J. Navarrete, Node. js, 2024.

Una vez realizada la descarga, se procede abrir el cmd como administrador para poder ejecutar el siguiente programa “npm install -g –unsafe-perm node-red”, como se puede observar en la figura 29, el cual instala el node-red.

Una vez culminada la instalación, se abrirá nuevamente la interfaz de comando y se ejecuta el programa “node red” donde ya cargada la información nos va a mostrar un enlace de dirección ip “http://127.0.0.1:1880/” como muestra la figura 30 que es el servidor de la máquina que nos direccionara hacia la plataforma del node red.

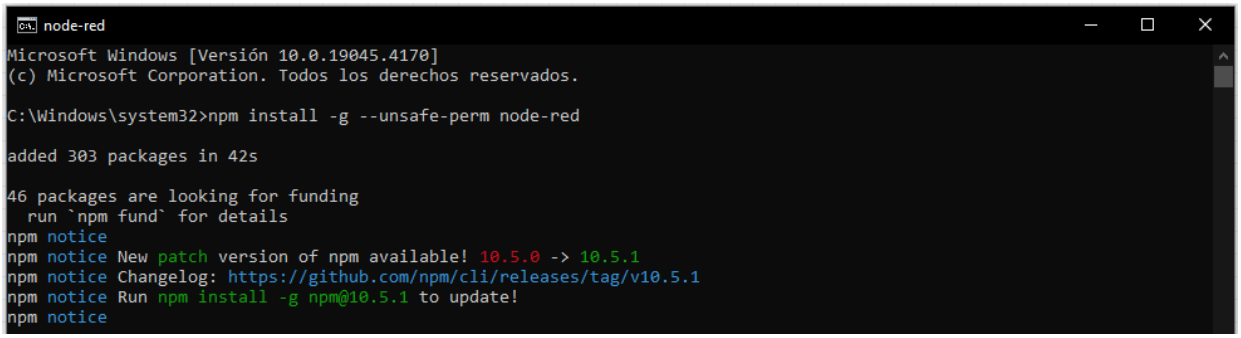


Figura 29. CMD instalación del Node Red C. Infante y J. Navarrete, Node red 2024.

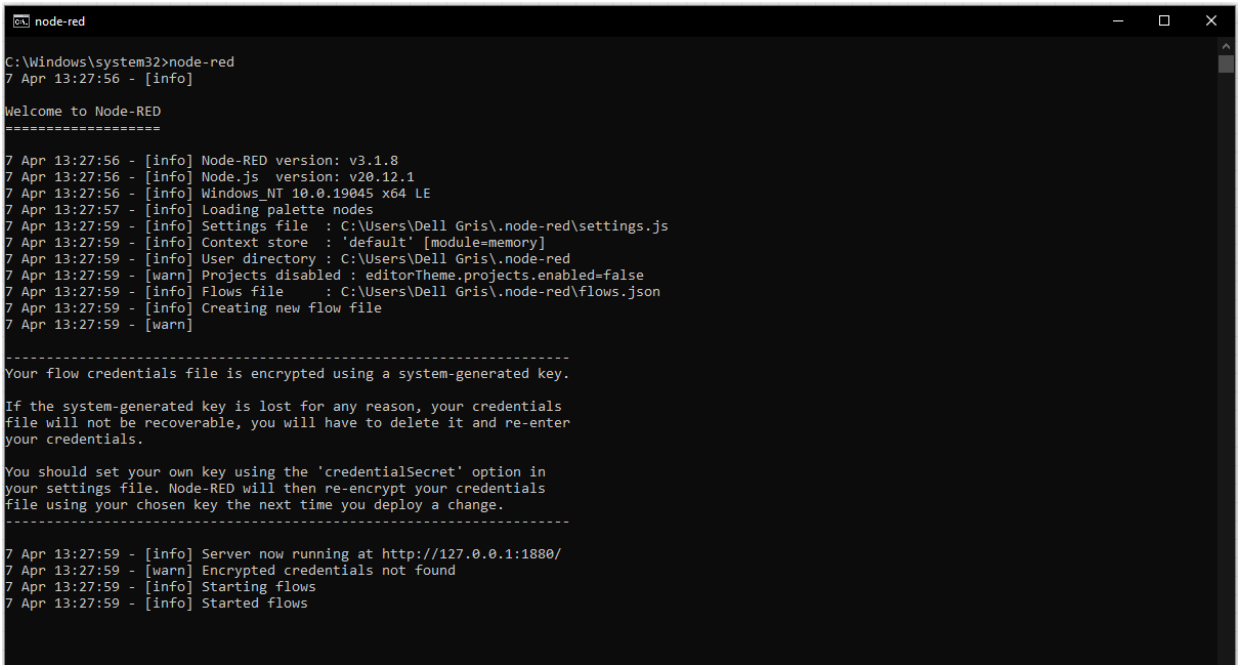


Figura 30. Dirección ip del node red, por C. Infante y J. Navarrete, Node red 2024.

El siguiente paso es abrir el navegador, en este caso Google Chrome, donde se introducirá en la barra de dirección la ip obtenida en el cmd de la figura 30 para tener acceso a la plataforma node red como se muestra en la figura 31, para poder comenzar a programar. Adicionalmente, la programación realizada su puede observar en el anexo D.

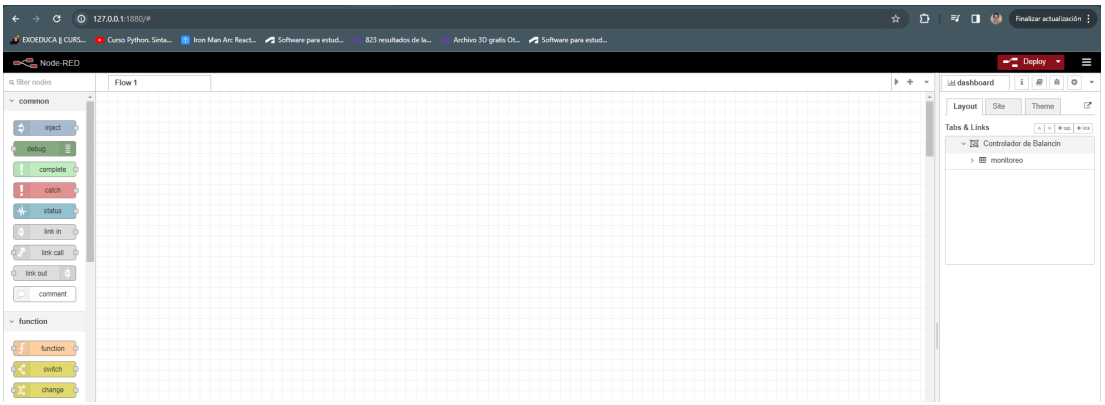


Figura 31. Node red C. Infante y J. Navarrete, Node red 2024.

VI. Resultados

En esta sección se mostrarán los resultados de la implementación de las diferentes etapas del sistema desarrolladas en la sección Marco Metodológico.

VI-A. Montaje de la etapa electrónica

Aquí se muestra como quedaron las conexiones físicamente implementadas, utilizando jumpers en algunos casos y en otros soldadura como se puede apreciar en las figuras 32 y 33.

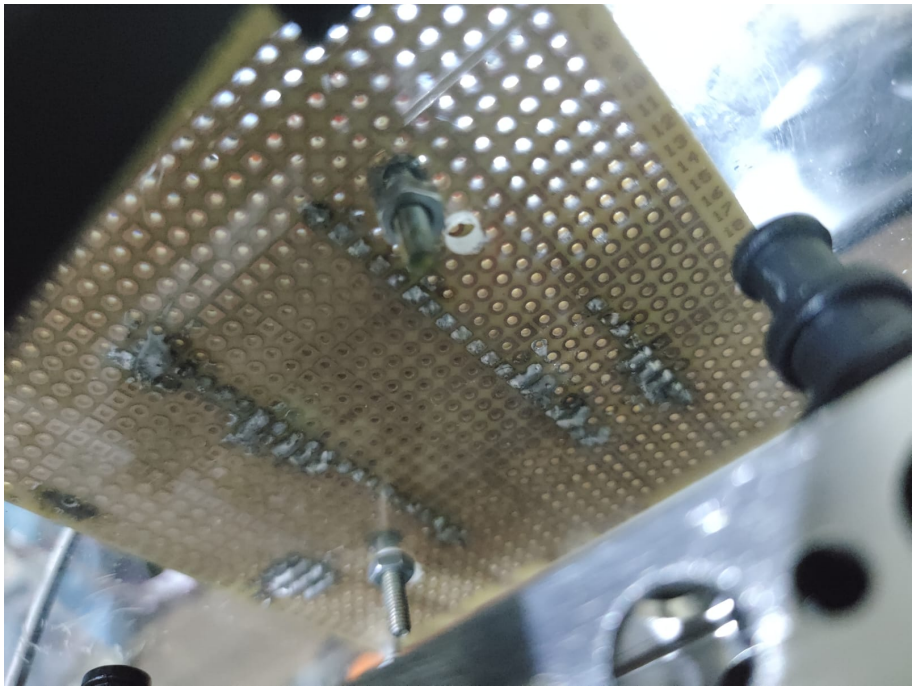


Figura 32. Circuito en baquelita perforada, por C. infante y J. Navarrete.

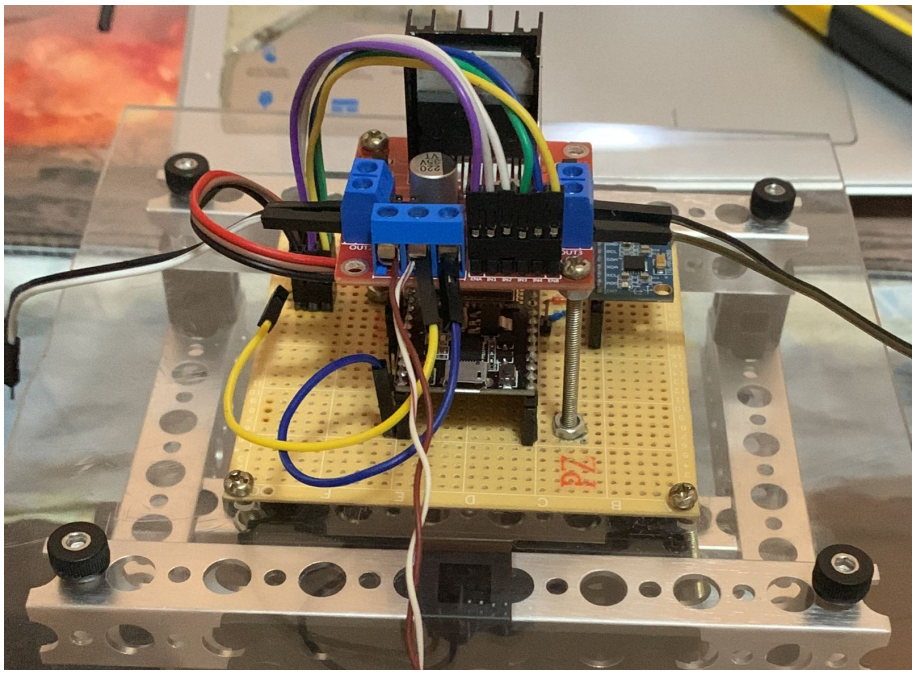


Figura 33. Montaje del microprocesador, puente H y el MPU6050, por C. infante y J. Navarrete.

VI-B. Control y monitoreo

En primer lugar, se muestran los resultados de las pruebas de lectura del sensor MPU6050 con los ángulos obtenidos al rotar el balancín como se puede observar en la figura 34.

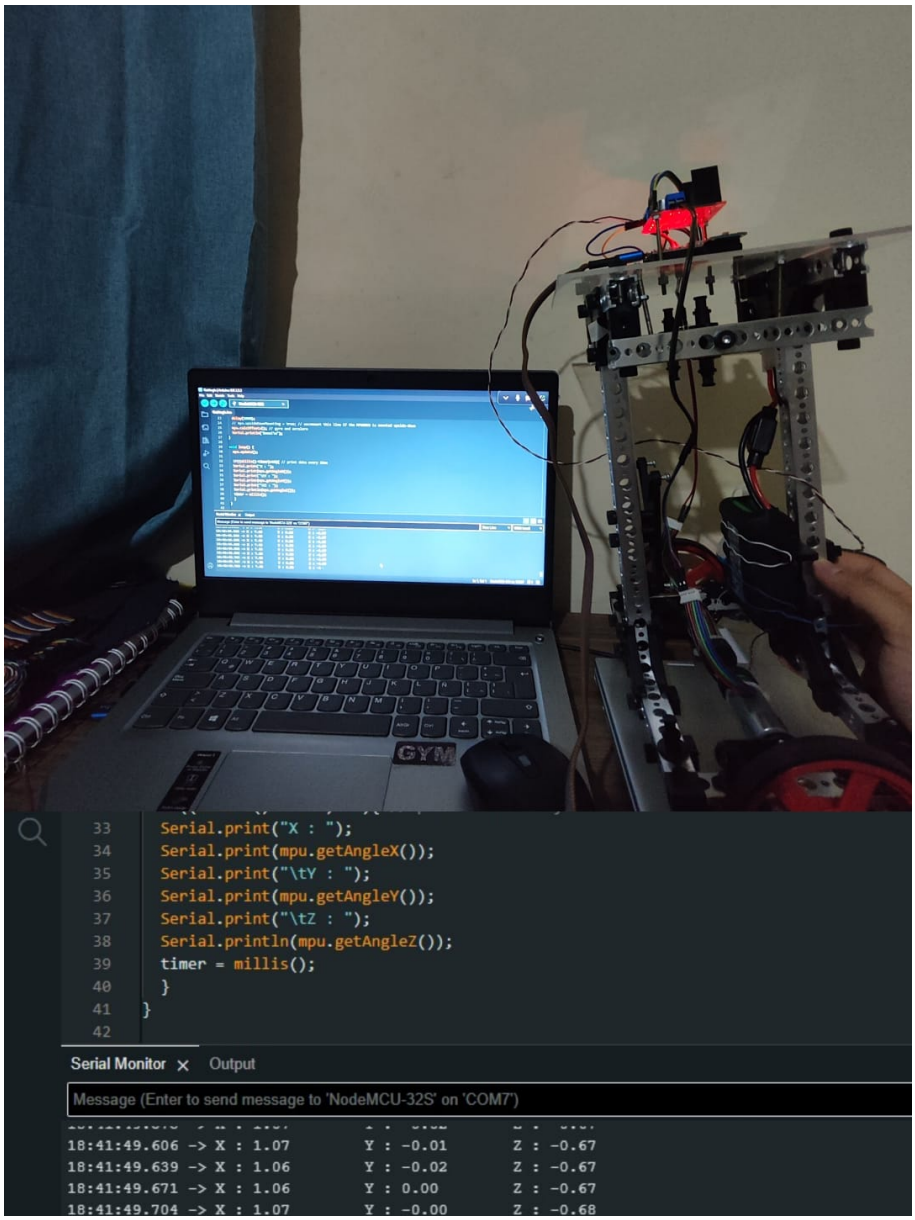


Figura 34. Prueba de ángulo, por C. infante y J. Navarrete.

Ahora, en la figura 35 se puede apreciar por la distorsión en la imagen que las ruedas están girando, lo que demuestra la manipulación del comportamiento de los motores.

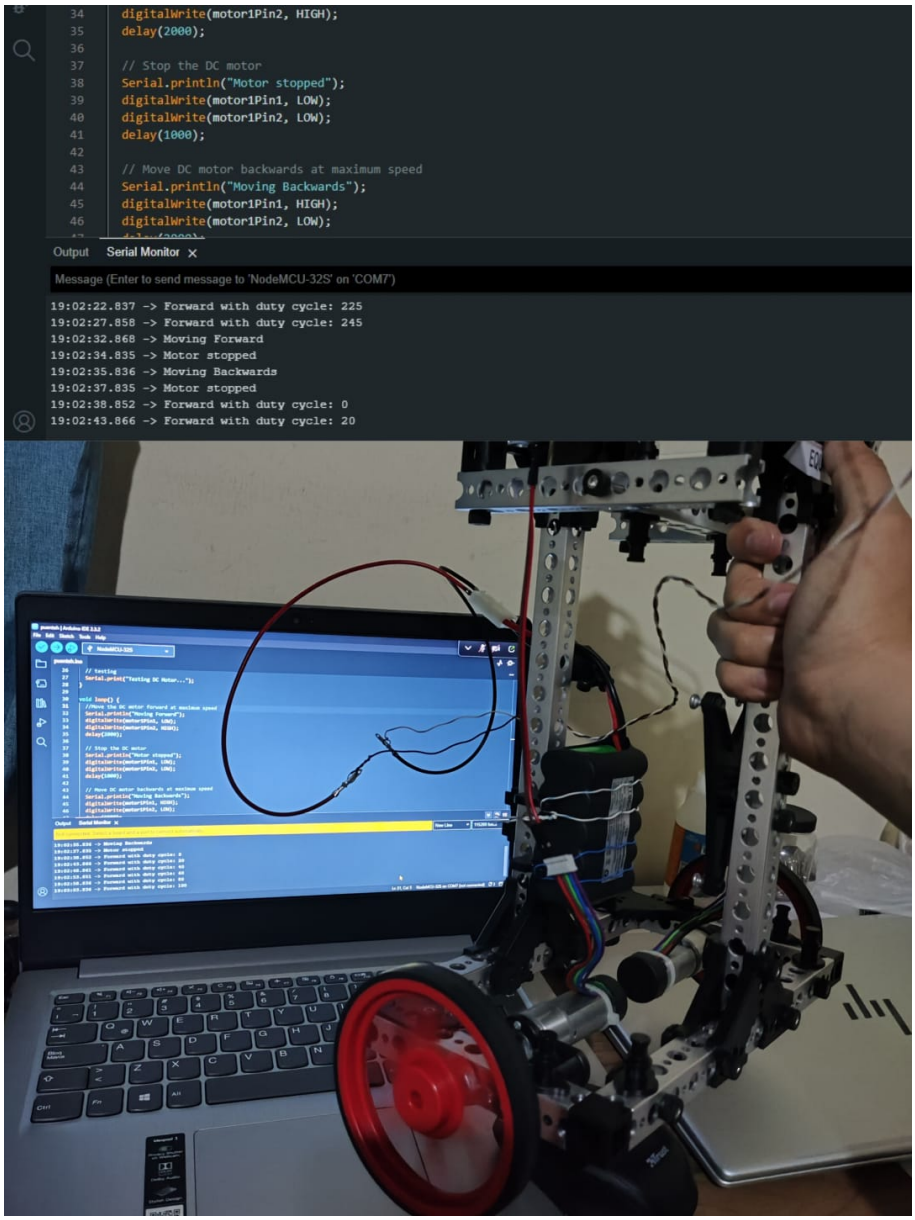


Figura 35. Prueba de motores, por C. infante y J. Navarrete.

A continuación la figura 36 muestra como cambia la señal de mando para los motores cuando cambia el ángulo de inclinación del balancín.

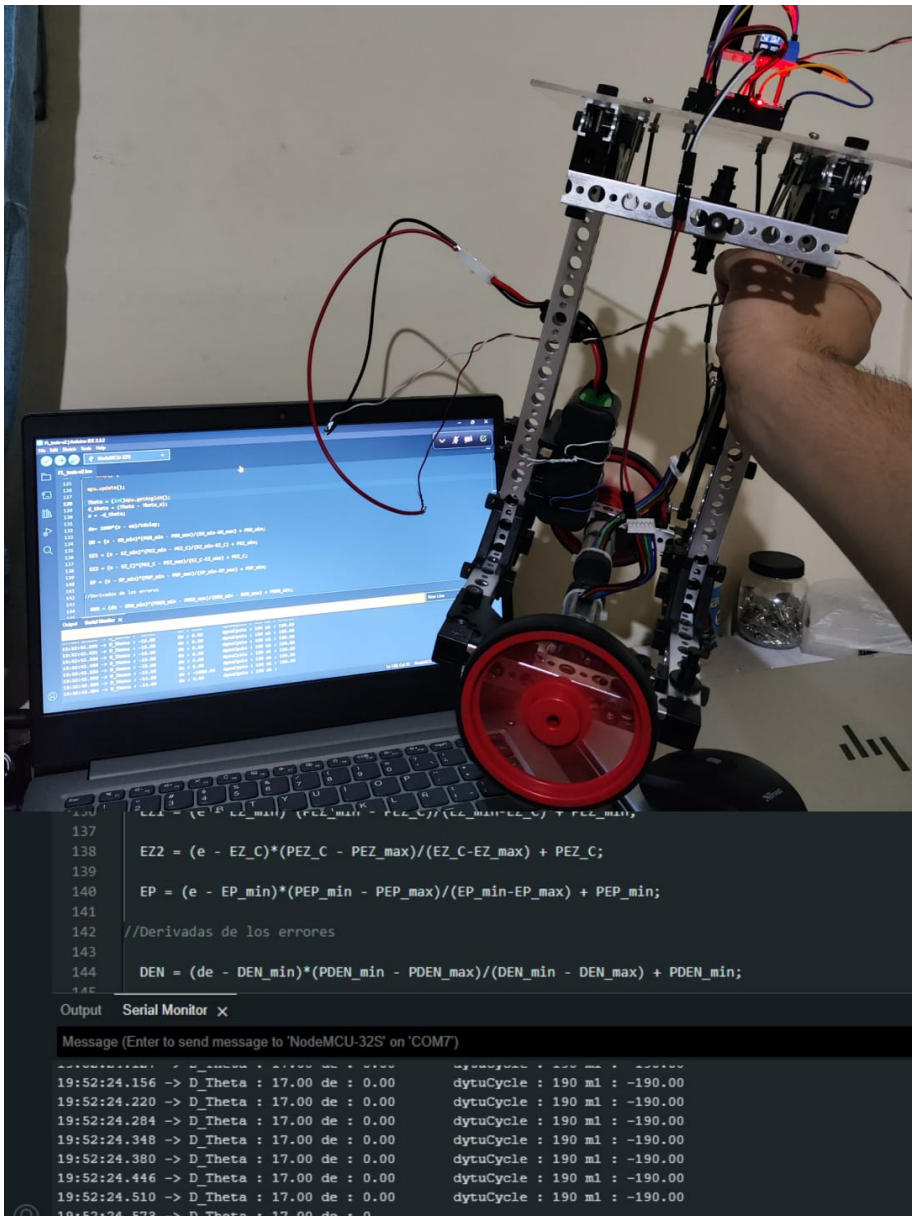


Figura 36. Lógica difusa, por C. infante y J. Navarrete.

Por último en las figuras 37 se muestra el monitoreo realizado con Node red celular que corresponde a la implementación de IoT.

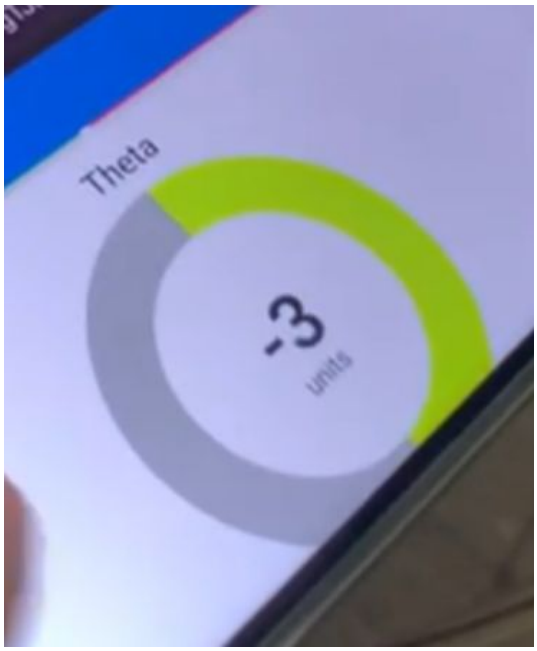


Figura 37. Ángulo del robot, por C. infante y J. Navarrete.

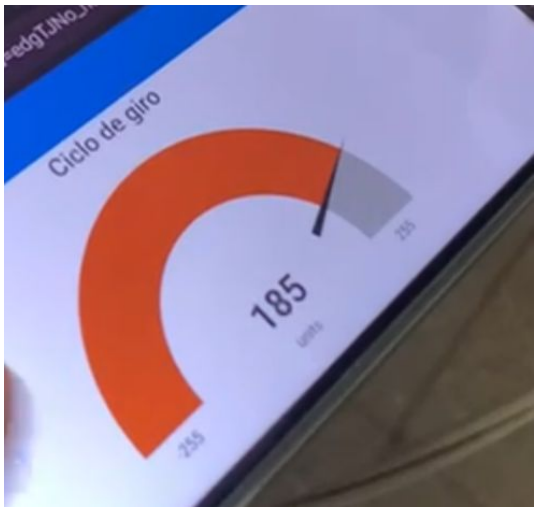
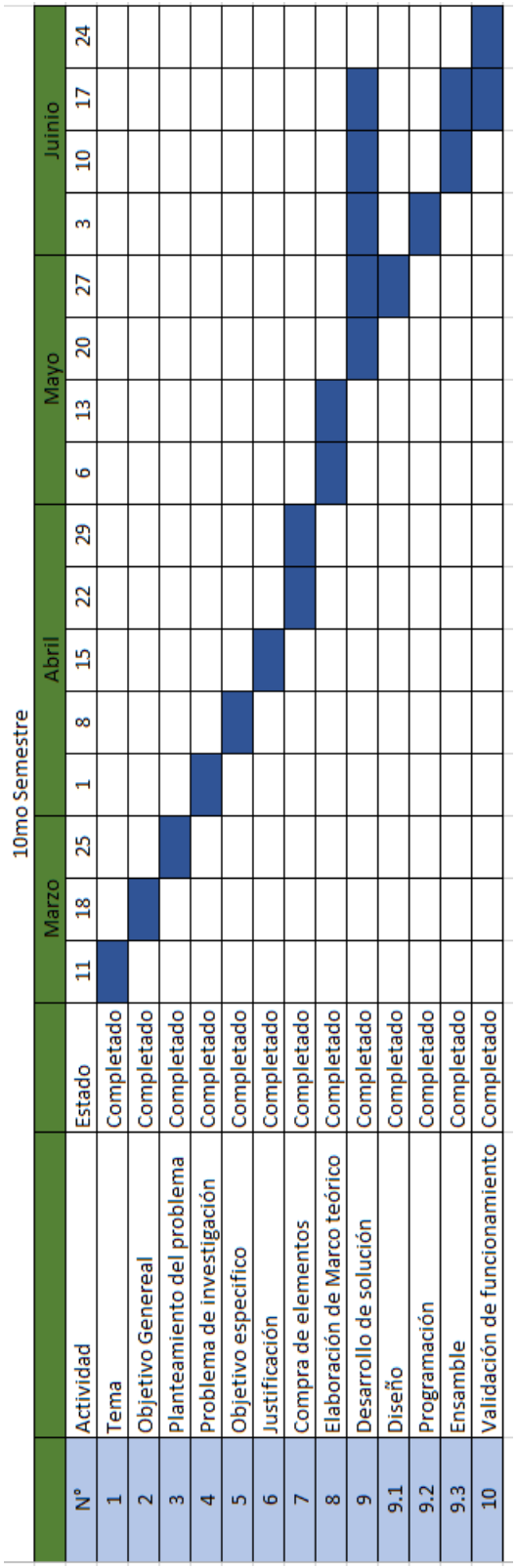


Figura 38. Velocidad del motor, por C. infante y J. Navarrete.

VII. CRONOGRAMA

Tabla IV
CRONOGRAMA



VIII. PRESUPUESTO

Tabla V
PRESUPUESTO

PRESUPUESTO				
Descripción	Componente	Valor	Cantidad	Total
Puente H	L298N	$4,70	1	$4,70
Microcontrolador	ESP32	$13,99	1	$13,99
Giroscopio/Acelerometro	MPU6050	$3,38	1	$3,38
Motoreductor amarillo		$1,25	2	$2,50
Bateria 9V Alcalina		$2,50	3	$7,50
Regulador Step-Down	Modulo LM2596	$2,67	2	$5,34
Movilización		$5,00	10	$50,00
Horas de trabajo		$10,00	80	$800,00
				$887,41

IX. Conclusiones

- El uso de software libre como Arduino IDE y Node Red permitieron el diseño e implementación del sistema de control del balancín o péndulo invertido; así como, el monitoreo remoto por IoT.

- En las pruebas se pudo apreciar que el monitoreo remoto funcionó correctamente. Sin embargo, el controlador tuvo ciertas deficiencias debido a la zona muerta que tienen los motores, ya que no se movían cuando el ciclo de trabajo es menor al $48{,}24\,\%$.

- El desarrollo de este proyecto permitirá realizar nuevas experiencias en asignaturas asociadas con sistemas embebidos, control y automatización, ya que la universidad cuenta con 10 de estos dispositivos que pueden ser ensamblados para formar tres diferentes plantas.

X. Recomendaciones

- Realizar una revisión más profunda del marco teórico para un desarrollo más eficiente de la solución.
- Fortalecer el conocimiento del sistema operativo Windows y de protocolos de comunicación para entender de mejor manera el funcionamiento de Node Red.
- Probar con anterioridad las diferentes librerías existentes para que el microcontrolador interactúe con el sensor MPU6050 porque su uso depende de la aplicación que se esté realizando.
- Es recomendable, primero verificar que el sistema pueda ser estabilizado con algún algoritmo de control sencillo para entender el funcionamiento del sistema en lazo cerrado.
- Realizar pruebas exhaustivas para determinar las no linealidades que va a enfrentar el desarrollo del controlador.
- Realizar las pruebas de forma independiente para cada componente y su interacción con el microcontrolador para verificar el correcto funcionamiento de las librerías.

REFERENCIAS

[1] *Lista de Precios Siemens Ecuador Junio 2023*, Accedido el 17 de marzo de 2024, Siemens, 2023. dirección: https://assets.new.siemens.com/siemens/assets/api/uuid:8f8448cd-3779-4ab9-ba58-50fff3c1c903/ListadepreciosSiemensEcuadorJun2023.pdf.

[2] Siemens. «Getting Started with Industrial Edge.» Accedido el 17 de marzo de 2024, Siemens. (2024), dirección: https://www.dex.siemens.com/industrialsoftware/digital-enterprise-services/getting-started-with-industrial-edge?cclcl=en_US.

[3] Á. P. Araujo Sánchez y D. E. Sanaguano Pincay, «Diseño e implementación de controlador PID para un balancín de dos grados de libertad,» B.S. thesis, 2020.

[4] A. D. Ñacata Paucar, «Implementación de un sistema WNCS mediante una plataforma basada en la nube para el control de un péndulo invertido,» B.S. thesis, 2022.

[5] D. Camacho-Montero et al., «Diseño basado en microcontrolador para el control de un péndulo invertido.,» 2023.

[6] A. Gracia Moisés, «Diseño y construcción de un robot auto-balanceado mediante Arduino,» 2017.

[7] C. C. Vaca y D. S. Morocho, «Estudio y diseño de un prototipo para el monitoreo de acuarios utilizando tecnología Wifi (IEEE 802.11 b/g/n) enfocada al IoT (Internet of things) mediante una plataforma Raspberry Pi y el sistema operativo Android,» *Quito Ecuador*, 2016.

[8] C. Jiménez Cortés, «Control de un vehículo equilibrista mediante una raspberry pi,» 2019.

[9] J. I. Cámara Molina, «Diseño, realización y control de un robot autoequilibrado de bajo coste basado en una Raspberry Pi,» 2020.

[10] J. E. Rojas Benavides y A. F. Navarro Vega, «Control de un Péndulo Invertido Didáctico Usando Sincronización en Tiempo Real con Matlab,» Tesis de grado, Universidad Industrial de Santander, Bucaramanga, Colombia, 2023. dirección: https://noesis.uis.edu.co/server/api/core/bitstreams/0c1a1f19-d74b-463c-94b0-b3b5e1612815/content.

[11] H. Pilamala, «Robot Balancing Basico,» mar. de 2019.

[12] K. J. Åström y T. Hägglund, *Control PID avanzado*. Pearson, Madrid, 2009.

[13] V. Alfaro, «ECUACIONES PARA CONTROLADORES PID UNIVERSALES,» *Ingeniería*, vol. 12, págs. 11-20, ene. de 2002. DOI: 10.15517/ring.v12i1-2.6429.

[14] *Control de P, I, D, PI, PD y PID*, [Online; accessed 2024-03-24], 2022.

[15] M. Woolf. «Control proporcional-integral-derivado (PID).» (2024), dirección: https://espanol.libretexts.org/Ingenieria/Ingenier%C3%ADa_Industrial_y_de_Sistemas/Libro%3A_Din%C3%A1mica_y_Controles_de_Procesos_Qu%C3%ADmicos_(Woolf)/09%3A_Control_proporcional-integral-derivado_(PID)/9.02%3A_Control_de_P%2C_I%2C_D%2C_PI%2C_PD_y_PID.

[16] C. F. P. Martín. «Control PID: qué es y cómo funciona.» Fecha de acceso: 24/3/2024. (2018), dirección: https://www.picuino.com/es/control-pid.html.

[17] Picuino. «Control PID.» (2024), dirección: https://www.picuino.com/es/control-pid.html.

[18] S. A. C. Giraldo. «Control Automático Educación.» Fecha de acceso: 24/3/2024. (2019), dirección: https://controlautomaticoeducacion.com.

[19] C. A. Educación. «Acción de control derivativo - Control PID.» (2024), dirección: https://controlautomaticoeducacion.com/control-realimentado/accion-de-control-derivativo-control-pid/.

[20] V. Mazzone, *Controladores PID*. 2002. dirección: https://www.eng.newcastle.edu.au/~jhb519/teaching/caut1/Apuntes/PID.pdf.

[21] Novus Automation. «Artículo: Control PID – Rompiendo la barrera del tiempo.» (2024), dirección: https://www.novusautomation.com/es/noticia/articulo-control-pid-rompiendo-la-barrera-del-tiempo.

[22] L. A. Bermeo Varon, J. G. Alvarez y W. Mantilla Arenas, «Comparación del desempeño de un controlador PID sobre el proceso de nivel usando un controlador lógico programable y un sistema embebido,» *Ingeniare. Revista chilena de ingeniería*, vol. 29, n.º 4, págs. 622-632, 2021.

[23] SciELO. «Control proporcional-integral-derivativo (PID).» (2021), dirección: https://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0718-33052021000400622.

[24] A. F. Alaminos-Fernández, «Introducción a la teoría de conjuntos difusos y sus aplicaciones en investigación social e IA,» 2023.
[25] R. Pérez Pueyo, *Procesado y optimización de espectros Raman mediante técnicas de lógica difusa: aplicación a la identificación de materiales pictóricos*. Universitat Politècnica de Catalunya, 2005.
[26] E Bello, «Lógica Difusa o Fuzzy Logic: Qué es y cómo funciona+ Ejemplos,» *Think. Innov. Available online: https://www. iebschool. com/blog/fuzzy-logic-que-es-big-data/(accessed on 4 December 2023)*, 2021.
[27] J. G. V. Hernández, O. D. M. Giraldo y D. G. Buitrago, «Lógica Difusa Aplicada al Control Local del Péndulo Invertido con Rueda de Reacción,» *Scientia et technica*, vol. 18, n.º 4, págs. 623-632, 2013.
[28] L. J. López, F. M. Zulay y A. S. Pateti, «Metodología de implementación de un controlador PID difuso en una FPGA,» *Universidad, Ciencia y Tecnología*, vol. 10, n.º 39, págs. 130-133, 2006.
[29] O. R. Ramos, «Simulación en simmechanics de un sistema de control difuso para el robot udlap,» 2008.
[30] ResearchGate. «Método de inferencia de Mamdani.» (n.d.), dirección: https://www.researchgate.net/figure/Metodo-de-inferencia-de-Mamdani_fig2_273128252.
[31] M. Espinoza Morillo, «Diseño de un controlador difuso basado en el esquema del compensador paralelo distribuido (PDC),» *Revista de la Facultad de Ingeniería Universidad Central de Venezuela*, vol. 26, n.º 1, págs. 95-105, 2011.
[32] D Rosales-Manzo, N García-Díaz, A Ruiz-Tadeo, J García-Virgen y N Farías-Mendoza, «Sistema difuso Takagi-Sugeno para predecir el riesgo de propagación de Sigatoka Negra Mycosphaerella fijiensis en el cultivo de plátano,» *RIIIT. Revista internacional de investigación e innovación tecnológica*, vol. 8, n.º 44, págs. 12-28, 2020.
[33] S. Diciembre Sanahuja, «Sistemas de control con lógica difusa: Métodos de mamdani y de takagi-sugeno-kang (tsk),» 2017.
[34] N. Mechatronics, «Tutorial MPU6050, Acelerómetro y Giroscopio,» *Naylamp Mechatronics Blog*, 2017. dirección: https://naylampmechatronics.com/blog/45_tutorial-mpu6050-acelerometro-y-giroscopio.html.
[35] Naylamp Mechatronics. «Tutorial MPU6050: Acelerómetro y Giroscopio.» (n.d.), dirección: https://naylampmechatronic com/blog/45_tutorial-mpu6050-acelerometro-y-giroscopio.html#:~:text=EL%20MPU6050%20es%20una%20unidad,%2C%20goniometr%C3%ADa%2C%20estabilizaci%C3%B3n%2C%20etc.
[36] *Acelerómetro y Giroscopio MPU6050*, Novatronicec, 24/3/2024. dirección: https://novatronicec.com/index.php/product/acelerometro-giroscopio-mpu6050/.
[37] J. S. Gómez González, «Desarrollo de un prototipo basado en sistemas embebidos para obtener el modelo matemático de la superficie cóncava en lentes oftalmológicos,» Universidad Católica de Santiago de Guayaquil, 2019. dirección: http://repositorio.ucsg.edu.ec/handle/3317/13373.
[38] N. Mechatronics, *Driver Puente H L298N 2A*, Naylamp Mechatronics Blog, 24/3/2024. dirección: https://naylampmechatronics.com/drivers/11-driver-puente-h-l298n.html.
[39] J. L. Quinde Llerena y L. D. Ulloa Patiño, «Diseño y Construcción de Dos Robots Tipo Warbot,» Tesis de mtría., Universidad de Guayaquil, 2012. dirección: https://dspace.ups.edu.ec/bitstream/123456789/6849/1/UPS-GT000656.pdf.
[40] Tecnología y Pedagogía. «Puente H.» (2020), dirección: https://www.tecnologiaypedagogia.net/2020/01/puente-h.html.
[41] J. E. Loaiza Armijos, «Diseño y construcción de un prototipo de control y monitoreo inalámbrico de un inventario de activos,» B.S. thesis, Quito, 2020., 2020.
[42] HobbyTronica. «NodeMCU ESP32 WiFi Bluetooth 4.2 IoT WROOM ESP32S USB-C.» Accedido el día Mes, Año. (Sin fecha de publicación), dirección: https://www.hobbytronica.com.ar/MLA-1463410834-nodemcu-esp32-wifi-bluetooth-42-iot-wroom-esp32s-usb-c-_JM.
[43] *ESP32 MANUAL*, Accedido el 24 de marzo de 2024, microelectrónicash, 2019. dirección: https://www.microelectronicash.com/downloads/ESP32_MANUAL.pdf.
[44] R. Santos. «ESP32 Pinout Reference: Which GPIO pins should you use?» Accedido el día Mes, Año. (Sin fecha de publicación), dirección: https://randomnerdtutorials.com/esp32-pinout-reference-gpios/.
[45] D. B. del Valle, «IoT: TECNOLOGÍAS, usos, tendencias y desarrollo futuro.,» B.S. thesis, 2014.

[46] K. Rose, S. Eldridge y L. Chapin, «La Internet de las Cosas—Una breve reseña,» Internet Society, inf. téc., 2015. dirección: https://www.internetsociety.org/wp-content/uploads/2017/09/report-InternetOfThings-20160817-es-1.pdf.

[47] I. Society. «Informe sobre el Internet de las Cosas.» Accedido el 17 de septiembre de 2017. (2016), dirección: https://www.internetsociety.org/wp-content/uploads/2017/09/report-InternetOfThings-20160817-es-1.pdf.

[48] Avelectronics, *Módulo de fuente de alimentación XL4015*, https://avelectronics.cc/producto/convertidor-voltaje-buck-dc-dc-adjustable-step-down-xl4015-5a/, 24/3/2024.

ANEXO A
MPU6050

```
MPU6050 mpu(Wire);
unsigned long timer = 0;

void setup() {
  Serial.begin(9600);
  Wire.begin();

  byte status = mpu.begin();
  Serial.print(F("MPU6050 status: "));
  Serial.println(status);
  while(status!=0){ } // stop everything if could not connect to MPU6050
  |
  Serial.println(F("Calculating offsets, do not move MPU6050"));
  delay(1000);
  // mpu.upsideDownMounting = true; // uncomment this line if the MPU6050 is mounted
upside-down
  mpu.calcOffsets(); // gyro and accelero
  Serial.println("Done!\n");
}

void loop() {
  mpu.update();

  if((millis()-timer)>10){ // print data every 10ms
  Serial.print("X : ");
  Serial.print(mpu.getAngleX());
  Serial.print("\tY : ");
  Serial.print(mpu.getAngleY());
  Serial.print("\tZ : ");
  Serial.println(mpu.getAngleZ());
  timer = millis();
  }
}
```

Figura 39. Código para extraer los ángulos del MPU6050, por C. Infante y J. Navarrete.

ANEXO B
PUENTE H

```
// Motor A
int motor1Pin1 = 27;
int motor1Pin2 = 26;
int enable1Pin = 14;

// Setting PWM properties
const int freq = 30000;
const int pwmChannel = 0;
const int resolution = 8;
int dutyCycle = 145;

void setup() {
 // sets the pins as outputs:
 pinMode(motor1Pin1, OUTPUT);
 pinMode(motor1Pin2, OUTPUT);
 pinMode(enable1Pin, OUTPUT);

 // configure LED PWM functionalitites
 ledcSetup(pwmChannel, freq, resolution);

 // attach the channel to the GPIO to be controlled
 ledcAttachPin(enable1Pin, pwmChannel);

 Serial.begin(115200);

 // testing
 Serial.print("Testing DC Motor...");
}
```

Figura 40. motores con L298N parte 1, por C. Infante y J. Navarrete.

```
void loop() {
  Move the DC motor forward at maximum speed
  Serial.println("Moving Forward");
  digitalWrite(motor1Pin1, LOW);
  digitalWrite(motor1Pin2, HIGH);
  delay(2000);

  // Stop the DC motor
  Serial.println("Motor stopped");
  digitalWrite(motor1Pin1, LOW);
  digitalWrite(motor1Pin2, LOW);
  delay(1000);

  // Move DC motor backwards at maximum speed
  Serial.println("Moving Backwards");
  digitalWrite(motor1Pin1, HIGH);
  digitalWrite(motor1Pin2, LOW);
  delay(2000);

  // Stop the DC motor
  Serial.println("Motor stopped");
  digitalWrite(motor1Pin1, LOW);
  digitalWrite(motor1Pin2, LOW);
  delay(1000);

  // Move DC motor forward with increasing speed
  digitalWrite(motor1Pin1, HIGH);
  digitalWrite(motor1Pin2, LOW);
  while (dutyCycle <= 255){
   ledcWrite(pwmChannel, dutyCycle);
   Serial.print("Forward with duty cycle: ");
```

Figura 41. motores con L298N parte 2, por C. Infante y J. Navarrete.

```
  Serial.println(dutyCycle);
  dutyCycle = dutyCycle + 20;
  delay(5000);
 }
 dutyCycle = 0;
}
```

Figura 42. motores con L298N parte 3, por C. Infante y J. Navarrete.

ANEXO C
CODIGO DEL CONTROLADOR Y SISTEMA DE MONITOREO

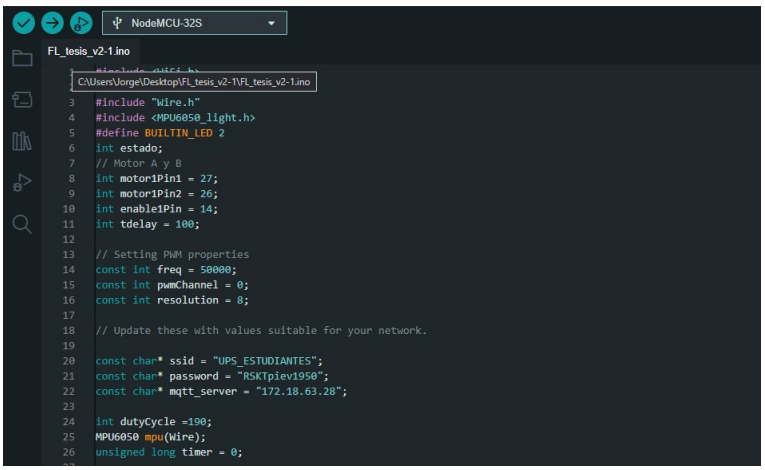


Figura 43. código parte 1, por C. infante y J. Navarrete.

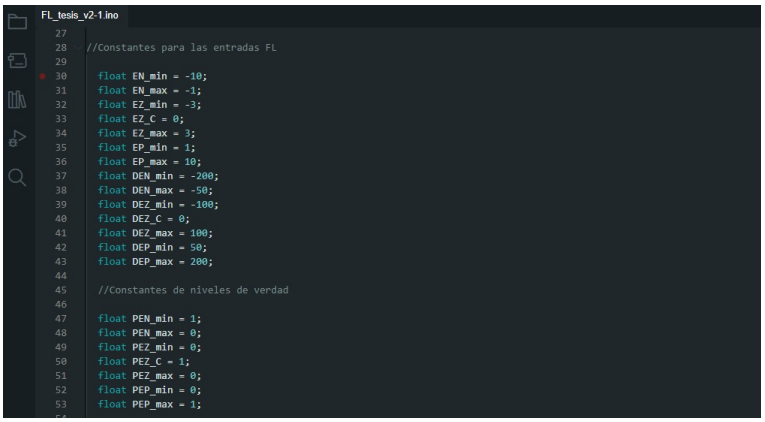


Figura 44. código parte 2, por C. infante y J. Navarrete..

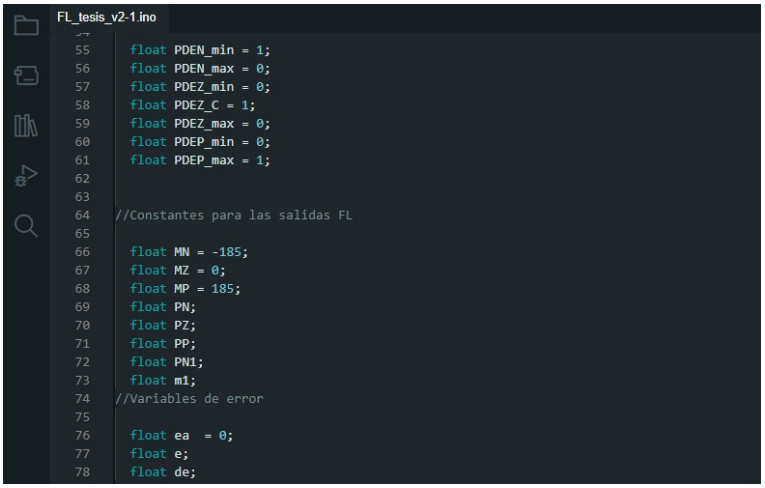


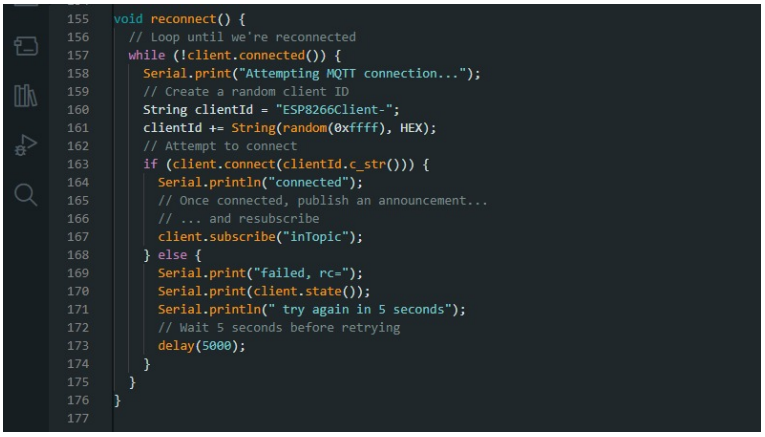
Figura 45. código parte 3, por C. infante y J. Navarrete.

Figura 46. código parte 4, por C. infante y J. Navarrete.

Figura 47. código parte 5, por C. infante y J. Navarrete.

```
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void callback(char* topic, byte* payload, unsigned int length) {
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println(estado);

  // Switch on the LED if an 1 was received as first character
  if ((char)payload[0] == '1') {
    digitalWrite(BUILTIN_LED, HIGH);   // Turn the LED on (Note that LOW is the voltage level
    estado=1;
    // but actually the LED is on; this is because
    // it is active low on the ESP-01)
  } else {
    digitalWrite(BUILTIN_LED, LOW);  // Turn the LED off by making the voltage HIGH
        estado=0;
  }

}
```

Figura 48. código parte 6, por C. infante y J. Navarrete.

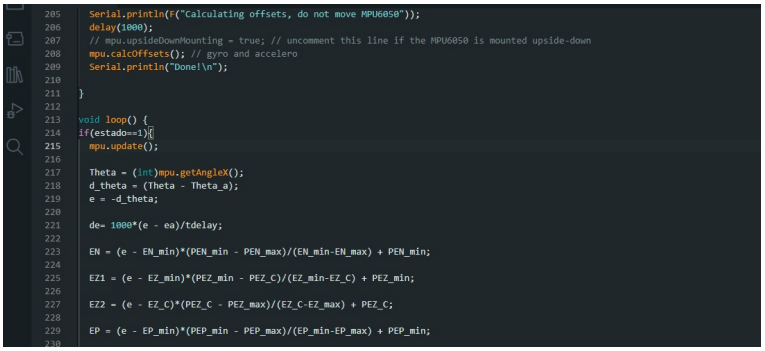


```
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP8266Client-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      // ... and resubscribe
      client.subscribe("inTopic");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
```

Figura 49. código parte 7, por C. infante y J. Navarrete.

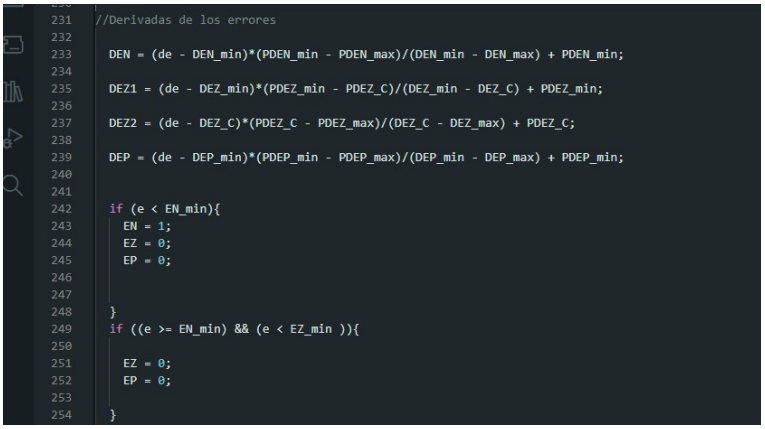
```
void setup() {
// sets the pins as outputs:

  pinMode(motor1Pin1, OUTPUT);
  pinMode(motor1Pin2, OUTPUT);
  pinMode(enable1Pin, OUTPUT);

  // configure LED PWM functionalitites
  ledcSetup(pwmChannel, freq, resolution);

  // attach the channel to the GPIO to be controlled
  ledcAttachPin(enable1Pin, pwmChannel);

  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
  // testing
  Serial.print("Testing DC Motor...");
    Wire.begin();

  byte status = mpu.begin();
  Serial.print(F("MPU6050 status: "));
  Serial.println(status);
  while(status!=0){ } // stop everything if could not connect to MPU6050
```

Figura 50. código parte 8, por C. infante y J. Navarrete.



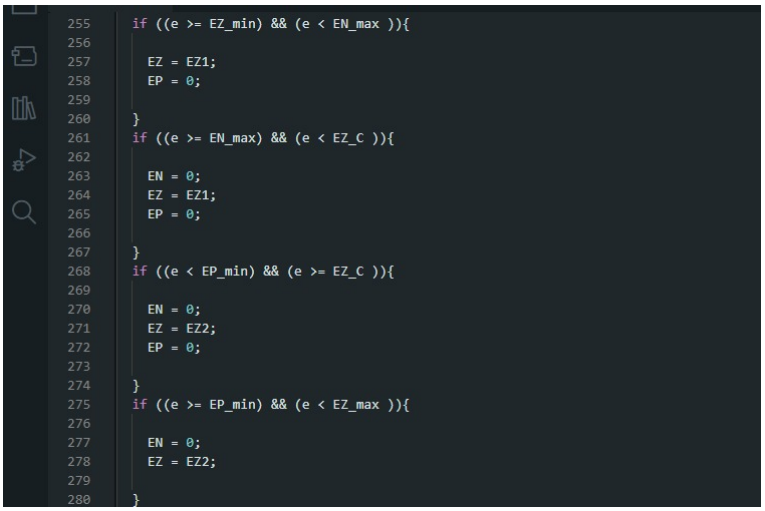
```
  Serial.println(F("Calculating offsets, do not move MPU6050"));
  delay(1000);
  // mpu.upsideDownMounting = true; // uncomment this line if the MPU6050 is mounted upside-down
  mpu.calcOffsets(); // gyro and accelero
  Serial.println("Done!\n");

}

void loop() {
if(estado==1){
  mpu.update();

  Theta = (int)mpu.getAngleX();
  d_theta = (Theta - Theta_a);
  e = -d_theta;

  de= 1000*(e - ea)/tdelay;

  EN = (e - EN_min)*(PEN_min - PEN_max)/(EN_min-EN_max) + PEN_min;

  EZ1 = (e - EZ_min)*(PEZ_min - PEZ_C)/(EZ_min-EZ_C) + PEZ_min;

  EZ2 = (e - EZ_C)*(PEZ_C - PEZ_max)/(EZ_C-EZ_max) + PEZ_C;

  EP = (e - EP_min)*(PEP_min - PEP_max)/(EP_min-EP_max) + PEP_min;
```

Figura 51. código parte 9, por C. infante y J. Navarrete.



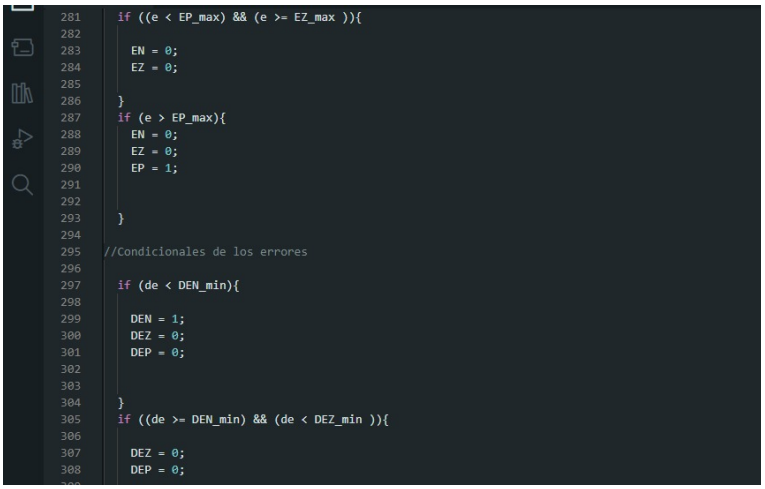
```
//Derivadas de los errores

  DEN = (de - DEN_min)*(PDEN_min - PDEN_max)/(DEN_min - DEN_max) + PDEN_min;

  DEZ1 = (de - DEZ_min)*(PDEZ_min - PDEZ_C)/(DEZ_min - DEZ_C) + PDEZ_min;

  DEZ2 = (de - DEZ_C)*(PDEZ_C - PDEZ_max)/(DEZ_C - DEZ_max) + PDEZ_C;

  DEP = (de - DEP_min)*(PDEP_min - PDEP_max)/(DEP_min - DEP_max) + PDEP_min;


  if (e < EN_min){
    EN = 1;
    EZ = 0;
    EP = 0;


  }
  if ((e >= EN_min) && (e < EZ_min )){

    EZ = 0;
    EP = 0;

  }
```

Figura 52. código parte 10, por C. infante y J. Navarrete.



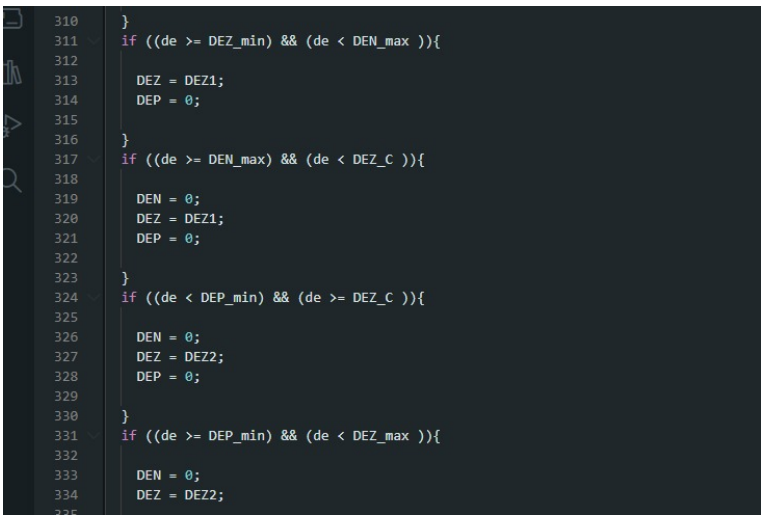
```
  if ((e >= EZ_min) && (e < EN_max )){

    EZ = EZ1;
    EP = 0;

  }
  if ((e >= EN_max) && (e < EZ_C )){

    EN = 0;
    EZ = EZ1;
    EP = 0;

  }
  if ((e < EP_min) && (e >= EZ_C )){

    EN = 0;
    EZ = EZ2;
    EP = 0;

  }
  if ((e >= EP_min) && (e < EZ_max )){

    EN = 0;
    EZ = EZ2;

  }
```

Figura 53. código parte 11, por C. infante y J. Navarrete.



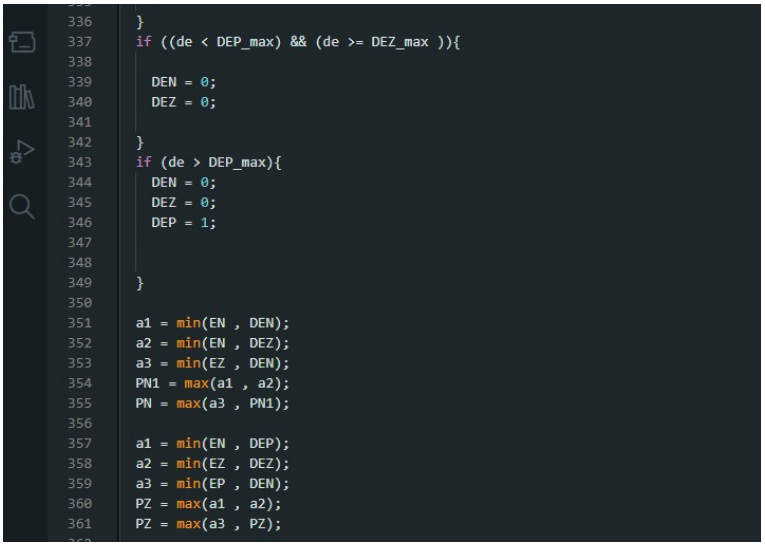
```
281   if ((e < EP_max) && (e >= EZ_max )){
282
283     EN = 0;
284     EZ = 0;
285
286   }
287   if (e > EP_max){
288     EN = 0;
289     EZ = 0;
290     EP = 1;
291
292
293   }
294
295 //Condicionales de los errores
296
297   if (de < DEN_min){
298
299     DEN = 1;
300     DEZ = 0;
301     DEP = 0;
302
303
304   }
305   if ((de >= DEN_min) && (de < DEZ_min )){
306
307     DEZ = 0;
308     DEP = 0;
```

Figura 54. código parte 12, por C. infante y J. Navarrete..



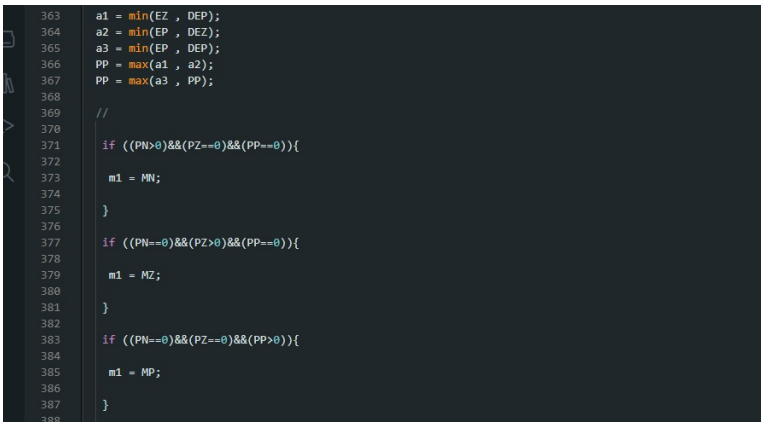
```
310   }
311   if ((de >= DEZ_min) && (de < DEN_max )){
312
313     DEZ = DEZ1;
314     DEP = 0;
315
316   }
317   if ((de >= DEN_max) && (de < DEZ_C )){
318
319     DEN = 0;
320     DEZ = DEZ1;
321     DEP = 0;
322
323   }
324   if ((de < DEP_min) && (de >= DEZ_C )){
325
326     DEN = 0;
327     DEZ = DEZ2;
328     DEP = 0;
329
330   }
331   if ((de >= DEP_min) && (de < DEZ_max )){
332
333     DEN = 0;
334     DEZ = DEZ2;
```

Figura 55. código parte 13, por C. infante y J. Navarrete.



```
  }
  if ((de < DEP_max) && (de >= DEZ_max )){

    DEN = 0;
    DEZ = 0;

  }
  if (de > DEP_max){
    DEN = 0;
    DEZ = 0;
    DEP = 1;



  }

  a1 = min(EN , DEN);
  a2 = min(EN , DEZ);
  a3 = min(EZ , DEN);
  PN1 = max(a1 , a2);
  PN = max(a3 , PN1);

  a1 = min(EN , DEP);
  a2 = min(EZ , DEZ);
  a3 = min(EP , DEN);
  PZ = max(a1 , a2);
  PZ = max(a3 , PZ);
```

Figura 56. código parte 14, por C. infante y J. Navarrete.



```
a1 = min(EZ , DEP);
a2 = min(EP , DEZ);
a3 = min(EP , DEP);
PP = max(a1 , a2);
PP = max(a3 , PP);

//

 if ((PN>0)&&(PZ==0)&&(PP==0)){

  m1 = MN;

 }

 if ((PN==0)&&(PZ>0)&&(PP==0)){

  m1 = MZ;

 }

 if ((PN==0)&&(PZ==0)&&(PP>0)){

  m1 = MP;

 }
```

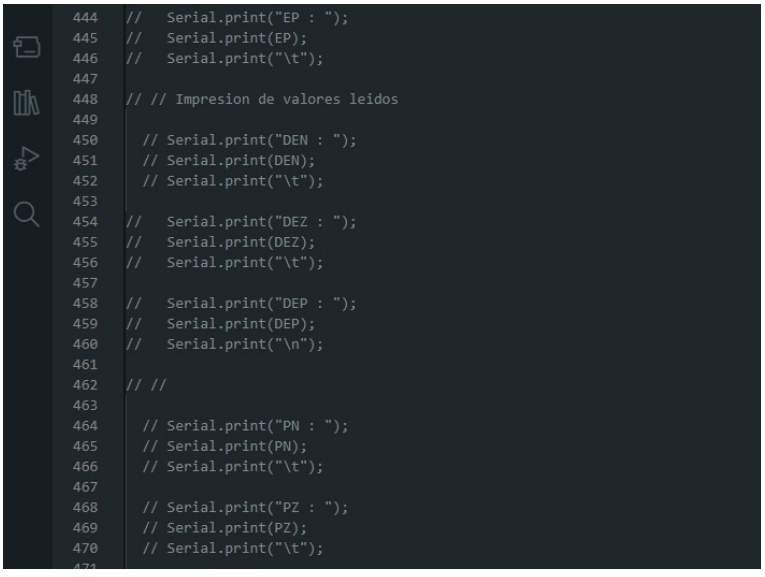
Figura 57. código parte 15, por C. infante y J. Navarrete.

```
  if ((PN>0)&&(PZ>0)&&(PP==0)){

   m1 = ((PZ*MZ)+(PN*MN))/(PN+PZ);

  }

  if ((PN==0)&&(PZ>0)&&(PP>0)){

   m1 = ((PP*MP)+(PZ*MZ))/(PP+PZ);

  }
// Control de motores
  if (m1>0){
 digitalWrite(motor1Pin1,HIGH);
 digitalWrite(motor1Pin2,LOW );
 dutyCycle = (int) m1;
 // if(dutyCycle > 1){

 //   dutyCycle = 180;

 // }

 //ledcWrite(pwmChannel, dutyCycle);
 //m2= (int)m1;
 }
```

Figura 58. código parte 16, por C. infante y J. Navarrete.

```
414   else if (m1<0){
415   digitalWrite(motor1Pin1, LOW);
416   digitalWrite(motor1Pin2, HIGH);
417   dutyCycle = -(int) m1;
418  //  if(dutyCycle > -1){
419
420  //     dutyCycle = -180;
421
422  //   }
423
424   //ledcWrite(pwmChannel, dutyCycle);
425   //m2= (int)-m1;
426   }
427   else {
428   digitalWrite(motor1Pin1, LOW);
429   digitalWrite(motor1Pin2, LOW);
430   }
431   if((dutyCycle>0) && (dutyCycle<145)){
432   dutyCycle = 145;
433   }
434   ledcWrite(pwmChannel, dutyCycle);
435   ea=e;
436   // Serial.print("EN : ");
437   // Serial.print(EN);
438   // Serial.print("\t");
439
440  //   Serial.print("EZ : ");
441  //   Serial.print(EZ);
442  //   Serial.print("\t");
```

Figura 59. código parte 17, por C. infante y J. Navarrete.



```
444  //   Serial.print("EP : ");
445  //   Serial.print(EP);
446  //   Serial.print("\t");
447
448  // // Impresion de valores leidos
449
450    // Serial.print("DEN : ");
451    // Serial.print(DEN);
452    // Serial.print("\t");
453
454  //   Serial.print("DEZ : ");
455  //   Serial.print(DEZ);
456  //   Serial.print("\t");
457
458  //   Serial.print("DEP : ");
459  //   Serial.print(DEP);
460  //   Serial.print("\n");
461
462  // //
463
464    // Serial.print("PN : ");
465    // Serial.print(PN);
466    // Serial.print("\t");
467
468    // Serial.print("PZ : ");
469    // Serial.print(PZ);
470    // Serial.print("\t");
```

Figura 60. código parte 1, por C. infante y J. Navarrete.

```
472   // Serial.print("PP : ");
473   // Serial.print(PP);
474   // Serial.print("\t");
475   Serial.print("Theta : ");
476   Serial.print(Theta);
477   Serial.print("\t");
478
479   Serial.print("de : ");
480   Serial.print(de);
481   Serial.print("\t");
482
483   Serial.print("dytuCycle : ");
484   Serial.print(dutyCycle);
485   Serial.print("\t");
486
487   Serial.print("m1 : ");
488   Serial.print(m1);
489   Serial.print("\n");
490   delay (tdelay);
491 }
492   else {
493   dutyCycle = 0;
494   ledcWrite(pwmChannel, dutyCycle);
495   }
```

Figura 61. código parte 19, por C. infante y J. Navarrete.

```
497
498   if (!client.connected()) {
499   reconnect();
500   }
501   else{}
502   client.loop();
503
504   unsigned long now = millis();
505   if (now - lastMsg > 2000) {
506     lastMsg = now;
507     int h=(int)mpu.getAngleX();
508     snprintf (msg, MSG_BUFFER_SIZE, "%d", h);
509     snprintf (msg2, MSG_BUFFER_SIZE, "%d",dutyCycle);
510     client.publish("theta", msg);
511     client.publish("Ciclo_giro", msg2);
512
513   }
514 }
515
```

Figura 62. código parte 20, por C. infante y J. Navarrete.

```
#include <WiFi.h>
#include <PubSubClient.h>
#include "Wire.h"
#include <MPU6050_light.h>
#define BUILTIN_LED 2
int estado;
// Motor A y B
int motor1Pin1 = 27;
int motor1Pin2 = 26;
int enable1Pin = 14;
int tdelay = 100;

// Setting PWM properties
const int freq = 50000;
const int pwmChannel = 0;
const int resolution = 8;

// Update these with values suitable for your network.

const char* ssid = "UPS_ESTUDIANTES";
const char* password = "RSKTpiev1950";
const char* mqtt_server = "172.18.63.28";

int dutyCycle =190;
MPU6050 mpu(Wire);
unsigned long timer = 0;

//Constantes para las entradas FL

  float EN_min = -10;
  float EN_max = -1;
  float EZ_min = -3;
  float EZ_C = 0;
  float EZ_max = 3;
  float EP_min = 1;
  float EP_max = 10;
  float DEN_min = -200;
  float DEN_max = -50;
  float DEZ_min = -100;
  float DEZ_C = 0;
  float DEZ_max = 100;
  float DEP_min = 50;
  float DEP_max = 200;
```

Figura 63. código parte 1, por C. infante y J. Navarrete.

```
  //Constantes de niveles de verdad

  float PEN_min = 1;
  float PEN_max = 0;
  float PEZ_min = 0;
  float PEZ_C = 1;
  float PEZ_max = 0;
  float PEP_min = 0;
  float PEP_max = 1;

  float PDEN_min = 1;
  float PDEN_max = 0;
  float PDEZ_min = 0;
  float PDEZ_C = 1;
  float PDEZ_max = 0;
  float PDEP_min = 0;
  float PDEP_max = 1;


//Constantes para las salidas FL

  float MN = -185;
  float MZ = 0;
  float MP = 185;
  float PN;
  float PZ;
  float PP;
  float PN1;
  float m1;
//Variables de error

  float ea  = 0;
  float e;
  float de;
// Lectura angulo

  float Theta ;
  float Theta_a = 0;
  float d_theta = 0;
```

Figura 64. código parte 2, por C. infante y J. Navarrete.

```
//Variables para fussificacion

  float EN;
  float EZ1;
  float EZ2;
  float EP;
  float DEN;
  float DEZ1;
  float DEZ2;
  float DEP;
  float EZ;
  float DEZ;

//Variables de apoyo

  float a1;
  float a2;
  float a3;

//
WiFiClient espClient;
PubSubClient client(espClient);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE (50)
char msg[MSG_BUFFER_SIZE];
char msg2[MSG_BUFFER_SIZE];
int value = 0;

void setup_wifi() {

  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  randomSeed(micros());

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

Figura 65. código parte 3, por C. infante y J. Navarrete.

```
void callback(char* topic, byte* payload, unsigned int length) {
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println(estado);

  // Switch on the LED if an 1 was received as first character
  if ((char)payload[0] == '1') {
    digitalWrite(BUILTIN_LED, HIGH);   // Turn the LED on (Note that LOW is the voltage level
    estado=1;
    // but actually the LED is on; this is because
    // it is active low on the ESP-01)
  } else {
    digitalWrite(BUILTIN_LED, LOW);  // Turn the LED off by making the voltage HIGH
        estado=0;
  }

}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP8266Client-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      // ... and resubscribe
      client.subscribe("inTopic");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
```

Figura 66. código parte 4, por C. infante y J. Navarrete..

```
void setup() {
// sets the pins as outputs:

  pinMode(motor1Pin1, OUTPUT);
  pinMode(motor1Pin2, OUTPUT);
  pinMode(enable1Pin, OUTPUT);

  // configure LED PWM functionalitites
  ledcSetup(pwmChannel, freq, resolution);

  // attach the channel to the GPIO to be controlled
  ledcAttachPin(enable1Pin, pwmChannel);

  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
  // testing
  Serial.print("Testing DC Motor...");
    Wire.begin();

  byte status = mpu.begin();
  Serial.print(F("MPU6050 status: "));
  Serial.println(status);
  while(status!=0){ } // stop everything if could not connect to MPU6050

  Serial.println(F("Calculating offsets, do not move MPU6050"));
  delay(1000);
  // mpu.upsideDownMounting = true; // uncomment this line if the MPU6050 is mounted upside-down
  mpu.calcOffsets(); // gyro and accelero
  Serial.println("Done!\n");

}
```

Figura 67. código parte 5, por C. infante y J. Navarrete.

```
void loop() {
if(estado==1){
  mpu.update();

  Theta = (int)mpu.getAngleX();
  d_theta = (Theta - Theta_a);
  e = -d_theta;

  de= 1000*(e - ea)/tdelay;

  EN = (e - EN_min)*(PEN_min - PEN_max)/(EN_min-EN_max) + PEN_min;

  EZ1 = (e - EZ_min)*(PEZ_min - PEZ_C)/(EZ_min-EZ_C) + PEZ_min;

  EZ2 = (e - EZ_C)*(PEZ_C - PEZ_max)/(EZ_C-EZ_max) + PEZ_C;

  EP = (e - EP_min)*(PEP_min - PEP_max)/(EP_min-EP_max) + PEP_min;

//Derivadas de los errores

  DEN = (de - DEN_min)*(PDEN_min - PDEN_max)/(DEN_min - DEN_max) + PDEN_min;

  DEZ1 = (de - DEZ_min)*(PDEZ_min - PDEZ_C)/(DEZ_min - DEZ_C) + PDEZ_min;

  DEZ2 = (de - DEZ_C)*(PDEZ_C - PDEZ_max)/(DEZ_C - DEZ_max) + PDEZ_C;

  DEP = (de - DEP_min)*(PDEP_min - PDEP_max)/(DEP_min - DEP_max) + PDEP_min;


  if (e < EN_min){
    EN = 1;
    EZ = 0;
    EP = 0;


  }
  if ((e >= EN_min) && (e < EZ_min )){

    EZ = 0;
    EP = 0;

  }
  if ((e >= EZ_min) && (e < EN_max )){

    EZ = EZ1;
    EP = 0;

  }
```

Figura 68. código parte 6, por C. infante y J. Navarrete.

```
if ((e >= EN_max) && (e < EZ_C )){

  EN = 0;
  EZ = EZ1;
  EP = 0;

}
if ((e < EP_min) && (e >= EZ_C )){

  EN = 0;
  EZ = EZ2;
  EP = 0;

}
if ((e >= EP_min) && (e < EZ_max )){

  EN = 0;
  EZ = EZ2;

}
if ((e < EP_max) && (e >= EZ_max )){

  EN = 0;
  EZ = 0;

}
if (e > EP_max){
  EN = 0;
  EZ = 0;
  EP = 1;


}
```

Figura 69. código parte 7, por C. infante y J. Navarrete.

```
//Condicionales de los errores

  if (de < DEN_min){

    DEN = 1;
    DEZ = 0;
    DEP = 0;


  }
  if ((de >= DEN_min) && (de < DEZ_min )){

    DEZ = 0;
    DEP = 0;

  }
  if ((de >= DEZ_min) && (de < DEN_max )){

    DEZ = DEZ1;
    DEP = 0;

  }
  if ((de >= DEN_max) && (de < DEZ_C )){

    DEN = 0;
    DEZ = DEZ1;
    DEP = 0;

  }
  if ((de < DEP_min) && (de >= DEZ_C )){

    DEN = 0;
    DEZ = DEZ2;
    DEP = 0;

  }
  if ((de >= DEP_min) && (de < DEZ_max )){

    DEN = 0;
    DEZ = DEZ2;

  }
  if ((de < DEP_max) && (de >= DEZ_max )){

    DEN = 0;
    DEZ = 0;

  }
  if (de > DEP_max){
    DEN = 0;
    DEZ = 0;
    DEP = 1;
        |
  }
```

Figura 70. código parte 8, por C. infante y J. Navarrete.

```
a1 = min(EN , DEN);
a2 = min(EN , DEZ);
a3 = min(EZ , DEN);
PN1 = max(a1 , a2);
PN = max(a3 , PN1);

a1 = min(EN , DEP);
a2 = min(EZ , DEZ);
a3 = min(EP , DEN);
PZ = max(a1 , a2);
PZ = max(a3 , PZ);

a1 = min(EZ , DEP);
a2 = min(EP , DEZ);
a3 = min(EP , DEP);
PP = max(a1 , a2);
PP = max(a3 , PP);

//

 if ((PN>0)&&(PZ==0)&&(PP==0)){

  m1 = MN;

 }

 if ((PN==0)&&(PZ>0)&&(PP==0)){

  m1 = MZ;

 }

 if ((PN==0)&&(PZ==0)&&(PP>0)){

  m1 = MP;

 }

 if ((PN>0)&&(PZ>0)&&(PP==0)){

  m1 = ((PZ*MZ)+(PN*MN))/(PN+PZ);

 }

 if ((PN==0)&&(PZ>0)&&(PP>0)){

  m1 = ((PP*MP)+(PZ*MZ))/(PP+PZ);

 }
```

Figura 71. código parte 9, por C. infante y J. Navarrete.

```
// Control de motores
   if (m1>0){
  digitalWrite(motor1Pin1,HIGH);
  digitalWrite(motor1Pin2,LOW );
  dutyCycle = (int) m1;
  // if(dutyCycle > 1){

  //   dutyCycle = 180;

  // }

  //ledcWrite(pwmChannel, dutyCycle);
  //m2= (int)m1;
  }
  else if (m1<0){
  digitalWrite(motor1Pin1, LOW);
  digitalWrite(motor1Pin2, HIGH);
  dutyCycle = -(int) m1;
//  if(dutyCycle > -1){

//     dutyCycle = -180;

//   }

  //ledcWrite(pwmChannel, dutyCycle);
  //m2= (int)-m1;
  }
  else {
  digitalWrite(motor1Pin1, LOW);
  digitalWrite(motor1Pin2, LOW);
  }
  if((dutyCycle>0) && (dutyCycle<145)){
  dutyCycle = 145;
  }
  ledcWrite(pwmChannel, dutyCycle);
  ea=e;
  // Serial.print("EN : ");
  // Serial.print(EN);
  // Serial.print("\t");

//   Serial.print("EZ : ");
//   Serial.print(EZ);
//   Serial.print("\t");

//   Serial.print("EP : ");
//   Serial.print(EP);
//   Serial.print("\t");

// // Impresion de valores leidos

  // Serial.print("DEN : ");
  // Serial.print(DEN);
  // Serial.print("\t");
```

Figura 72. código parte 10, por C. infante y J. Navarrete.

```
//   Serial.print("DEZ : ");
//   Serial.print(DEZ);
//   Serial.print("\t");

//   Serial.print("DEP : ");
//   Serial.print(DEP);
//   Serial.print("\n");

// //

  // Serial.print("PN : ");
  // Serial.print(PN);
  // Serial.print("\t");

  // Serial.print("PZ : ");
  // Serial.print(PZ);
  // Serial.print("\t");

  // Serial.print("PP : ");
  // Serial.print(PP);
  // Serial.print("\t");
  Serial.print("Theta : ");
  Serial.print(Theta);
  Serial.print("\t");

  Serial.print("de : ");
  Serial.print(de);
  Serial.print("\t");

  Serial.print("dytuCycle : ");
  Serial.print(dutyCycle);
  Serial.print("\t");

  Serial.print("m1 : ");
  Serial.print(m1);
  Serial.print("\n");
  delay (tdelay);
}
  else {
  dutyCycle = 0;
  ledcWrite(pwmChannel, dutyCycle);
  }
```

Figura 73. código parte 11, por C. infante y J. Navarrete.

```
  if (!client.connected()) {
  reconnect();
  }
  else{}
  client.loop();

  unsigned long now = millis();
  if (now - lastMsg > 2000) {
    lastMsg = now;
    int h=(int)mpu.getAngleX();
    snprintf (msg, MSG_BUFFER_SIZE, "%d", h);
    snprintf (msg2, MSG_BUFFER_SIZE, "%d",dutyCycle);
    client.publish("theta", msg);
    client.publish("Ciclo_giro", msg2);

  }
}
```

Figura 74. código parte 12, por C. infante y J. Navarrete.

ANEXO D
MONITORIO NODE RED

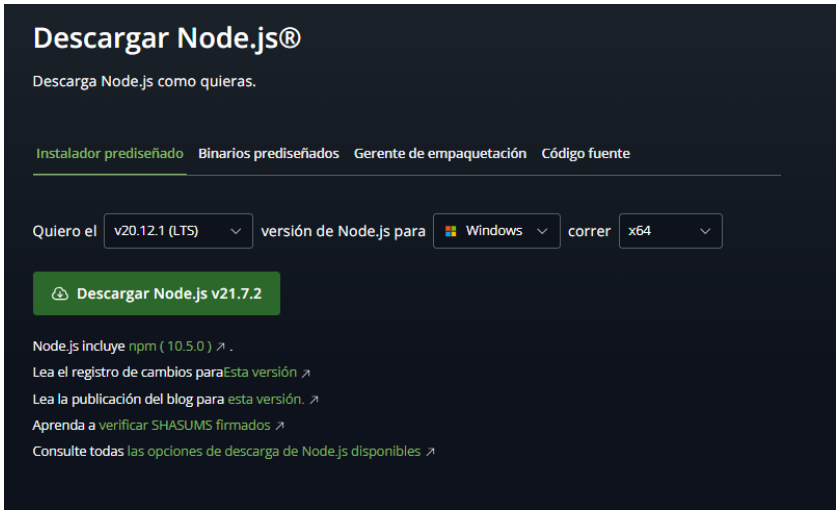


Figura 75. Descarga del sistema Node.js, por C. infante y J. Navarrete.

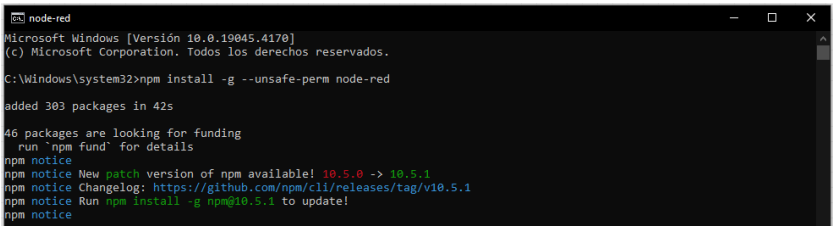


Figura 76. Ingresar el comando en cmd, por C. infante y J. Navarrete.

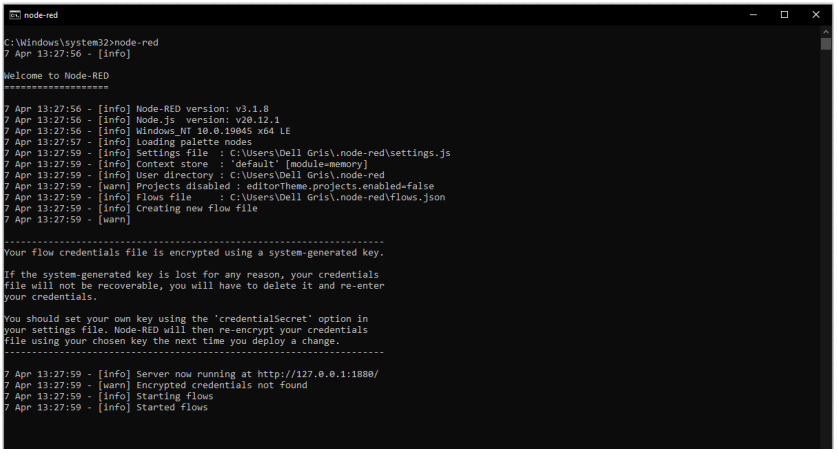


Figura 77. Comando Node red e Ip de acceso, por C. infante y J. Navarrete.

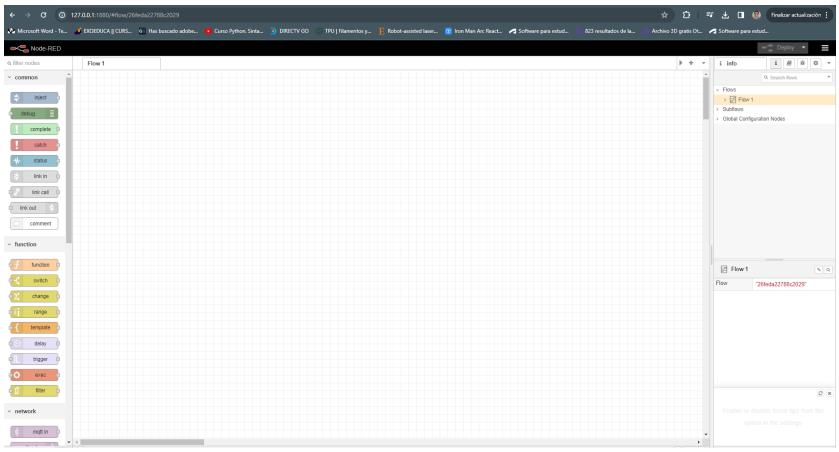


Figura 78. Plataforma Node red, por C. infante y J. Navarrete.

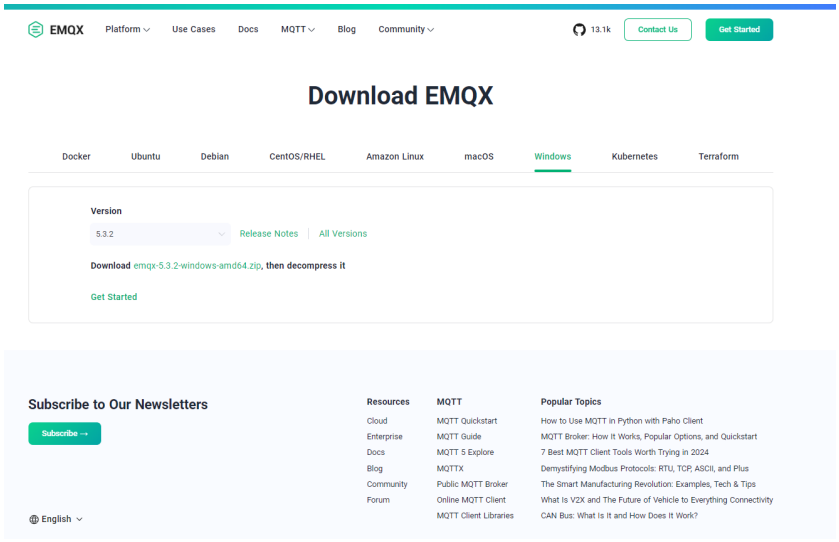


Figura 79. Plataforma de descargar del EMQX, por C. infante y J. Navarrete.

En la figura a continuación se procede con la instalación del EMQX mediante el cmd, para eso la carpeta que descarga de la figura anterior, se la debe guardar en el disco C: para proceder abrir la carpeta bin, se escribe en la ruta de carpetas donde esta resaltado “cmd” y te abrira el software ya con la interfaz direccionado a esa carpeta y lo único que se debe de escribir “emqx star” tal cual como esta en la imagen para poder darle acceso al servidor

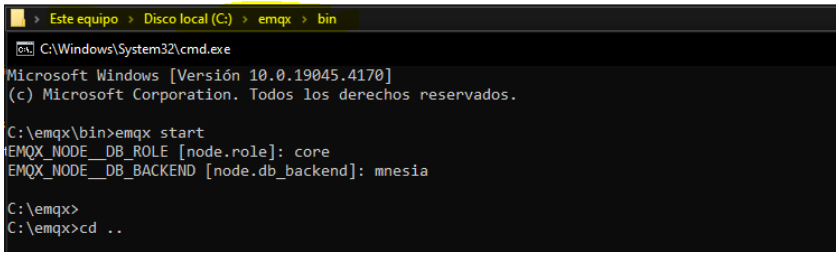


Figura 80. Instalación por cmd del EMQX, por C. infante y J. Navarrete.

A continuación de eso debes nuevamente colocar “cd”, esto no sirve para ir prompt principal del disco local C: en donde se va a colocar el siguiente codigo “Ipconfig” tal cual como muestra en la imagen, esto nos ayudara a saber la Ip de la cual nos estamos conectando y a su vez nos servira como ruta para el navegador del EMXQ

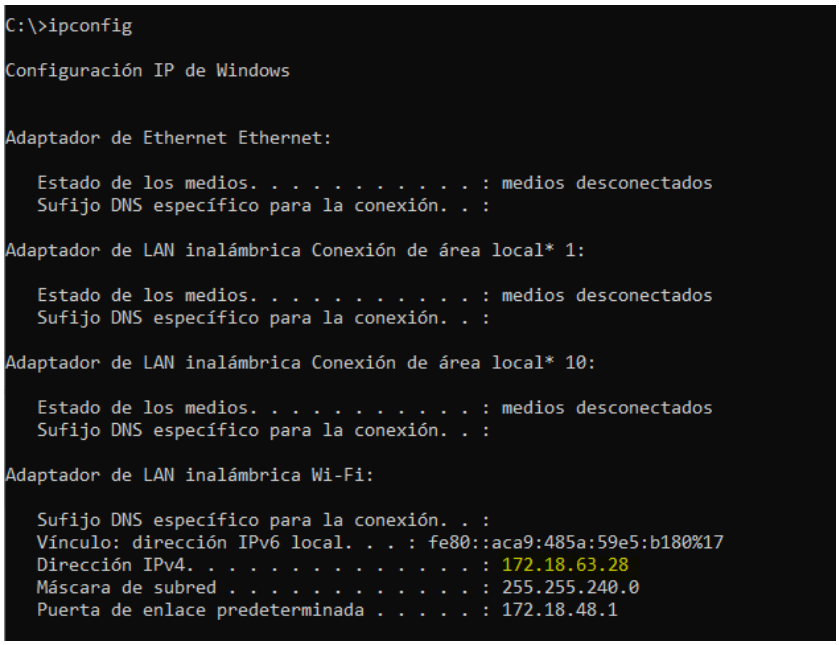


Figura 81. Dirección ip para el acceso al navegador del EMQX, por C. infante y J. Navarrete.

Con la dirección Ip que muestra en la figura anterior ya mencionado, vamos al buscador de Google Chrome y pegamos la ip y añadismo lo siguiente “:18083/ ” de tal manera que queda de esta forma

- http://localhost:18083/

donde el local host es la dirección ip de la red que se esta conectada en ese momento y seria como la figura que se muestra a continuación

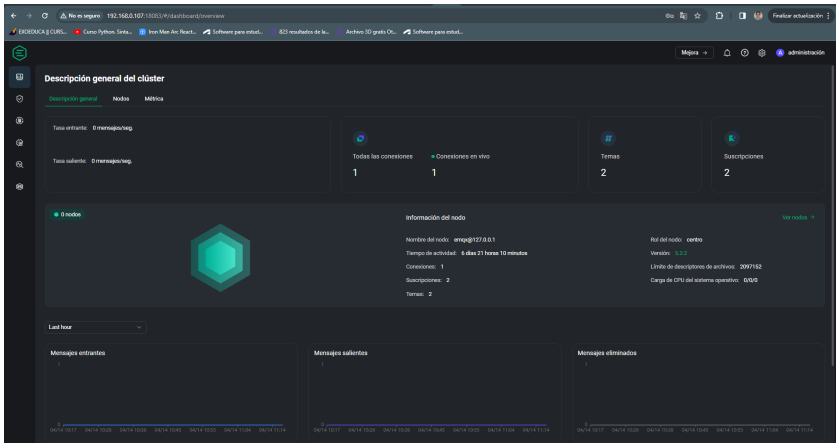


Figura 82. Navegador del EMQX, por C. infante y J. Navarrete.

Se direcciona a la barrita del lado izquierdo donde se encuentra resaltado en la figura que se encuentra a continuación y seleciona “Cliente WebSocket/ ” donde se verifica si esta todo correctamente instalado y configurado

para poder tener una adecuada programación en el node red, ya por la razón que la plataforma se trabaja con el servidor MQTT5

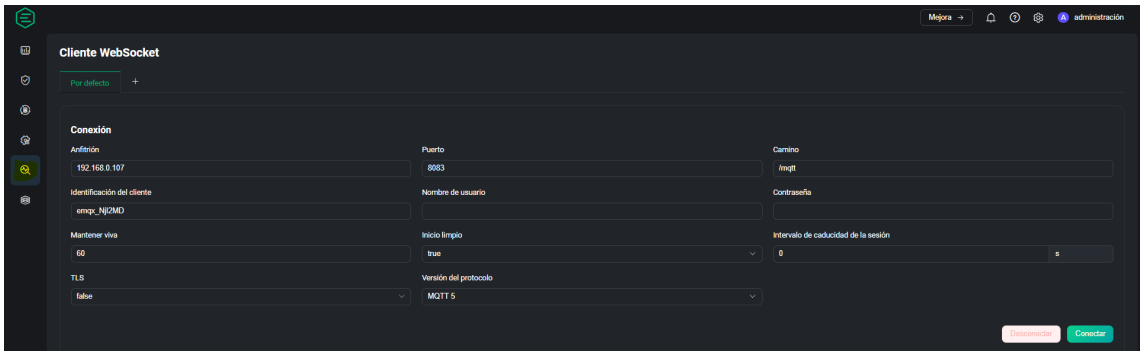


Figura 83. Navegador del EMQX, por C. infante y J. Navarrete.

regresamos a la pagina de node red una vez instalado el EMQX con el servidor de MQQT5 cuya imagen ya fue mencionada con anterioridad, nos direccionamos a las tres rayas que estan en el lado posterior derecho para instalas la los paquetes de la node red, una vez que se intala, nos dirigimos la flechita para abrir las otras opciones donde direccionamos a la opción de dashboard como se muestras en la imagen.

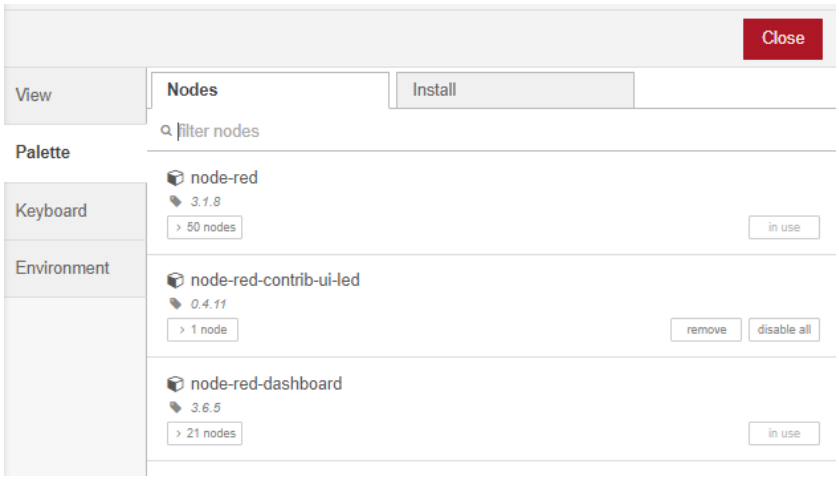


Figura 84. Crear una tab nueva, por C. infante y J. Navarrete.

y procedemos a crear un tab para poder configurar los servidores para que pueda leer y recibir la señal como se muestra en la siguiente figura

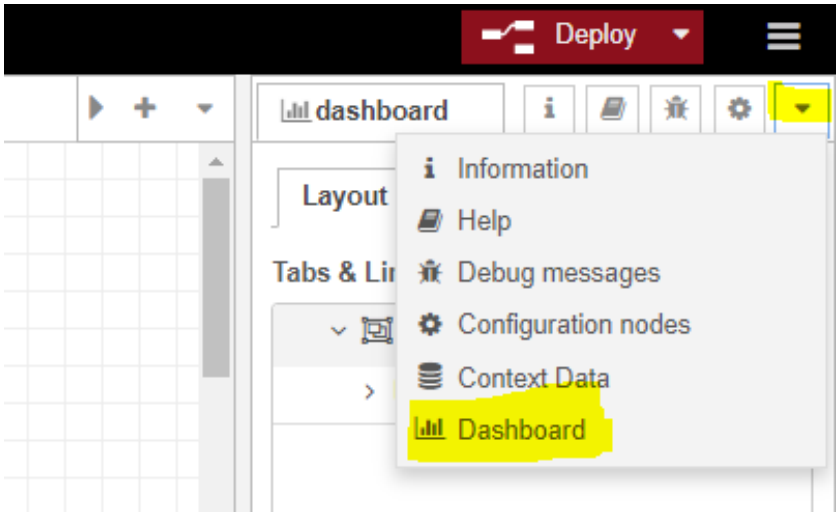


Figura 85. Colocar el nombre a nuestra base, por C. infante y J. Navarrete.

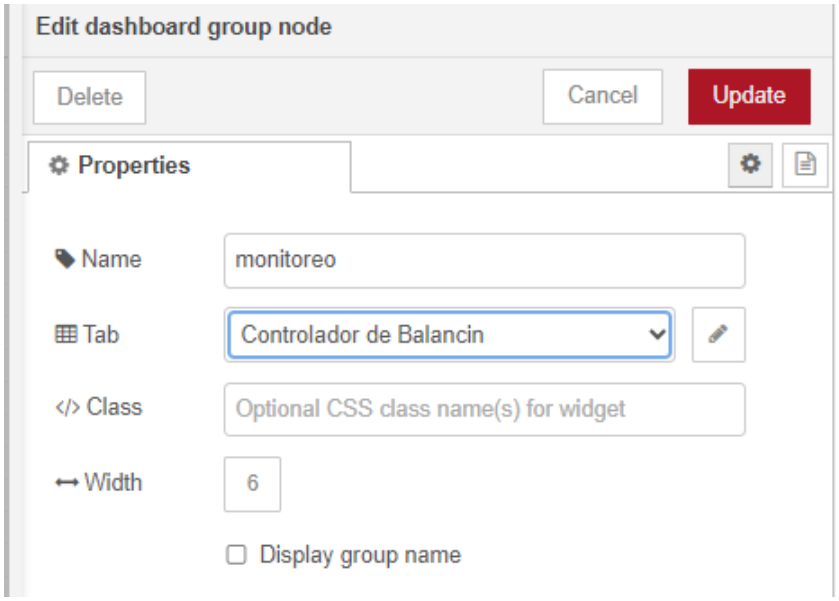


Figura 86. Crear una tab nueva, por C. infante y J. Navarrete.

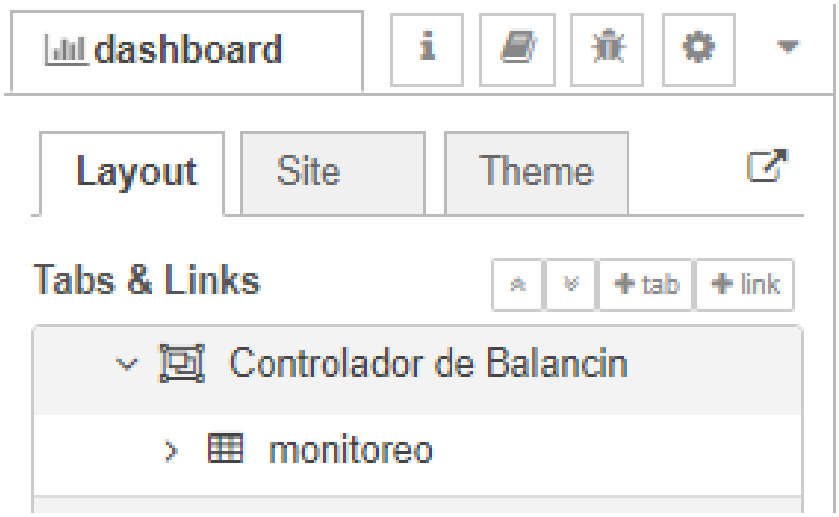


Figura 87. Colocar el nombre a nuestra base, por C. infante y J. Navarrete.

Se procede a programar para poder enviar la señal a nuestro microcontrolador ESP32 en donde solo vamos a necesitar la comunicación del mqtt in (entrada) y la salida que es mqtt out el cual se debe de configurar con la dirección ip de la maquina que se indico con anterioridad, también se necesitara una función para mostrar el sentido del giro en donde nos indicara los ángulos del robot adicional otro para saber la velocidad del motor, los botones de inicio y alto para controlar al robot cuya especificaciones son mostradas en la siguiente figuras.

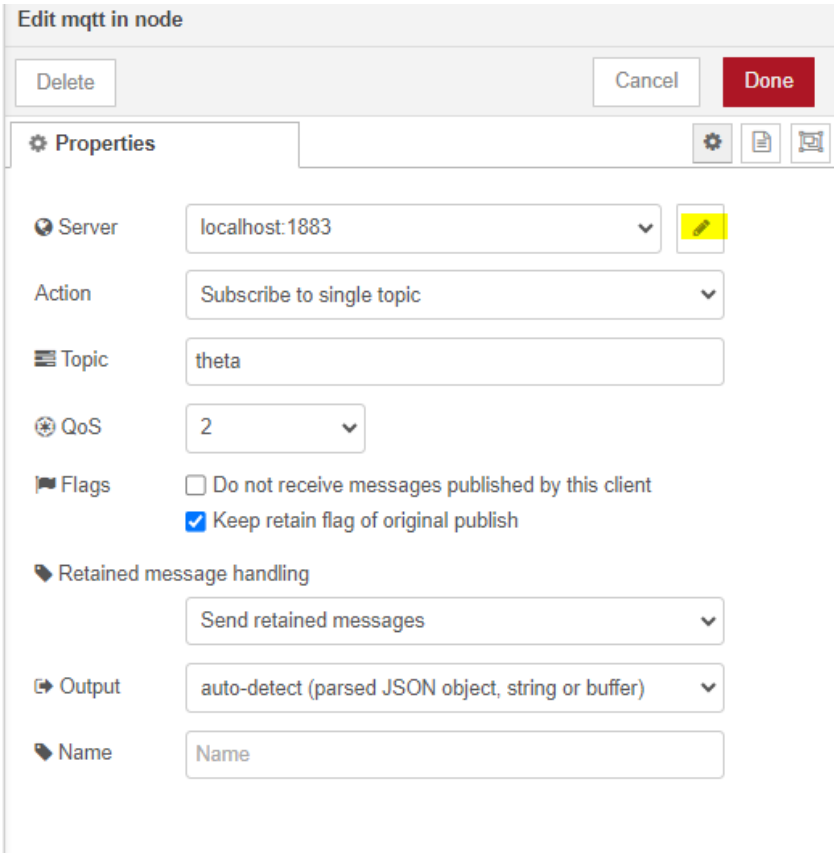


Figura 88. configuración del localhost de mqtt, por C. infante y J. Navarrete.

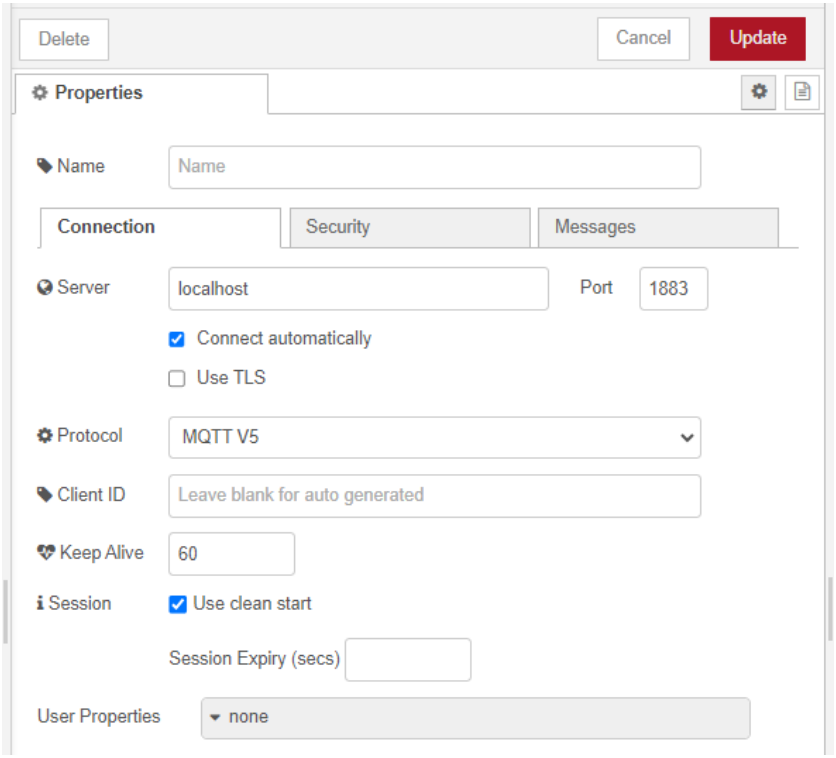


Figura 89. Verificación de puerto port, por C. infante y J. Navarrete.

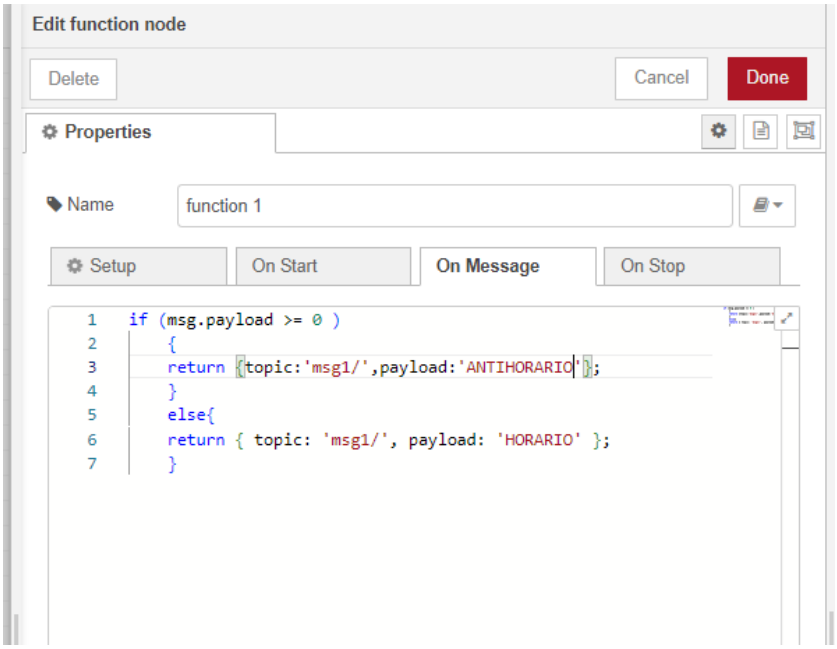


Figura 90. Configuración de la opción función para mostrar los ángulos desde los código del ESP32, por C. infante y J. Navarrete.

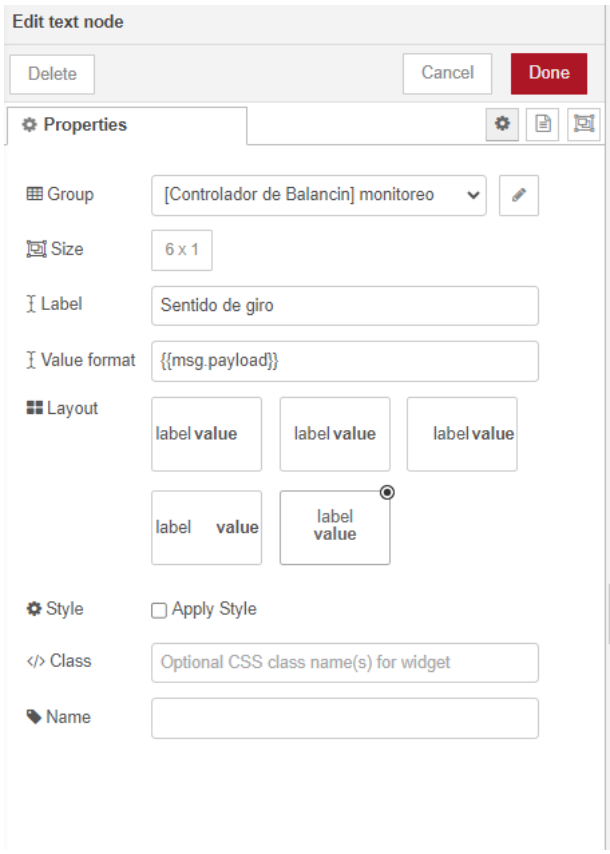


Figura 91. Nombre del controlador balancin, por C. infante y J. Navarrete.

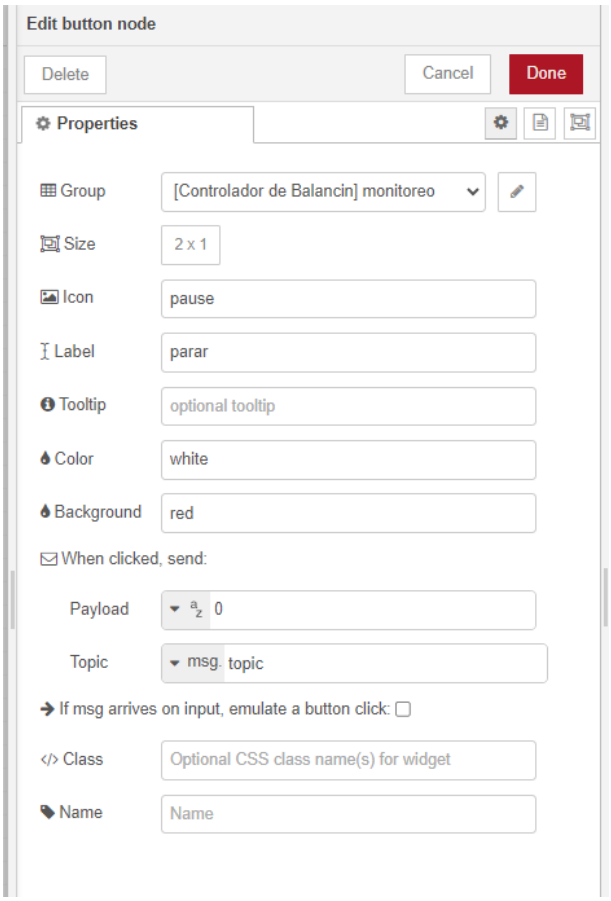


Figura 92. Accionar del encendido y apagado via remota al robot, por C. infante y J. Navarrete.

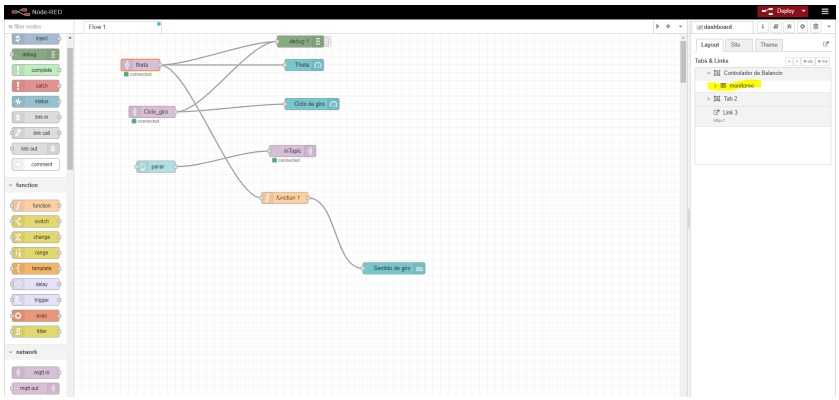


Figura 93. Comunicación del node red hacia el microprocesador ESP32, por C. infante y J. Navarrete.

Figura 94. Monitorio del robot mediante la node red, por C. infante y J. Navarrete.

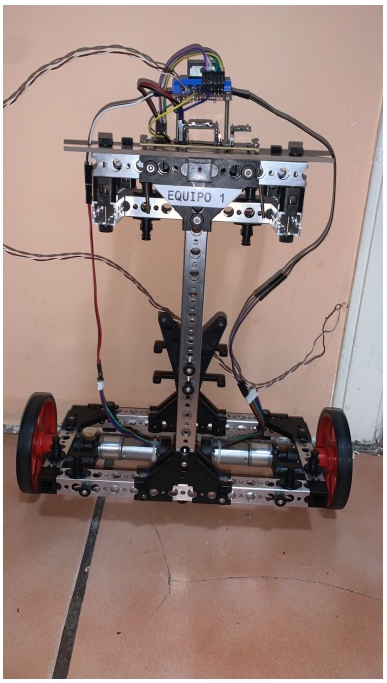


Figura 95. Robot balancin, por C. infante y J. Navarrete.

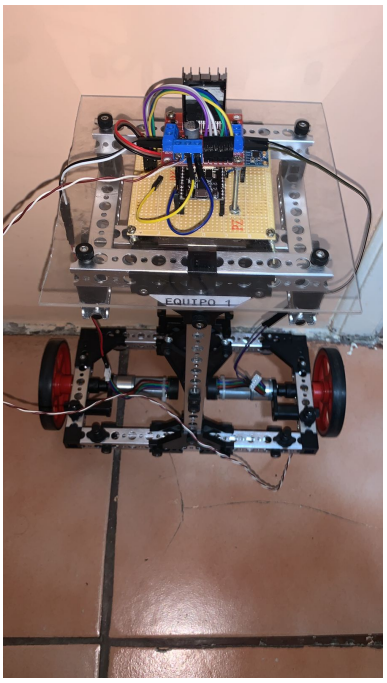


Figura 96. Robot balancin, por C. infante y J. Navarrete.