UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:
INGENIERÍA ELECTRÓNICA

Trabajo de titulación previo a la obtención del título de:

INGENIEROS ELECTRÓNICOS

TEMA:
ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO

AUTORES:
WILSON GABRIEL ORDÓÑEZ ALARCÓN

WELLINGTON ENRIQUE PINO NOGUERA

TUTOR:
CARLOS GERMÁN PILLAJO ANGOS

Quito, septiembre de 2021

CESIÓN DE DERECHOS DE AUTOR

Nosotros Wilson Gabriel Ordóñez Alarcón y Wellington Enrique Pino Noguera, con documentos de identificación N° 1900614775 y Nº 1721350971 respectivamente, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación intitulado: **ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO**, mismo que ha sido desarrollado para optar por el título de: INGENIEROS ELECTRÓNICOS, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autores nos reservamos los derechos morales de la obra antes citada. En concordancia, suscribimos este documento en el momento que hacemos entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

.....

Wilson Gabriel Ordóñez Alarcón

Cédula: 1900614775

.....

Wellington Enrique Pino Noguera

Cédula: 1721350971

Quito, septiembre de 2021.

DECLARATORIA DE COAUTORÍA DEL DOCENTE TUTOR

Yo declaro que bajo mi dirección y asesoría fue desarrollado el trabajo de titulación ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO realizado por Wilson Gabriel Ordóñez Alarcón y Wellington Enrique Pino Noguera, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana para ser considerado como trabajo final de titulación.

Quito, septiembre de 2021.

CARLOS GERMÁN PILLAJO ANGOS

C. I: 1709255119

DEDICATORIA

La elaboración de esta Tesis va principalmente dedicada a mis padres, quienes estuvieron siempre acompañándome a la distancia y apoyándome en todo sentido a lo largo de mi vida; a mi hermana quien me acompaño en toda mi vida universitaria, de la que he aprendido mucho y a la que considero mi segunda madre; y mi hija quien ahora es el motor que mueve todo en mi vida.

Wilson Ordóñez

Este trabajo quiero dedicarlo primero a Dios, a quien siempre he tenido presente en mi vida, a mi madre quien incondicionalmente me ha brindado su apoyo en todo el trayecto de mi vida, a mi padre que desde el cielo me acompaña, a mis abuelos María Elena y Vicente por su apoyo y aliento durante el desarrollo de este trabajo y en la vida.

Wellington Pino

AGRADECIMIENTO

Este trabajo va en muestra de agradecimiento especialmente, a toda la entrega y paciencia de mis padres; el apoyo y cuidados de parte de mi hermana; al esfuerzo y dedicación de mi compañero Wellington y mío, con quien he forjado una gran amistad; y a la guía de parte de nuestro tutor Carlos Pillajo, quien ha estado presente a lo largo de nuestra vida universitaria siendo un profesional ejemplar en la enseñanza nuestra y de la comunidad salesiana.

Wilson Ordóñez

Agradezco a todas las personas que me han acompañado en este viaje llamado vida dejando una huella significativa en mi memoria, sobre todo a mi madre, por su apoyo, a mis amigos Aldair, Majo, Marcelo, Mauri, Michelle, a quienes les tengo especial gratitud por su apoyo en diferentes etapas y formas, a Wilson a quien considero un hermano y un amigo, a nuestro tutor Carlos Pillajo un docente excepcional, que ha mostrado entrega a la enseñanza y a la formación de excelentes profesionales, y ciudadanos.

Wellington Pino

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

En el presente trabajo se muestra el procedimiento llevado a cabo para realizar la implementación de los controladores LQG, LQI y PID a través de WNCS, aplicados a un levitador neumático, y la comparación del tiempo de estabilización entre los mismos, cuya planta consta de una base de madera con un orificio en el medio, y en esta se ubica por la parte inferior un ventilador, el cual se encarga de enviar aire hacia un embudo colocado en la parte superior, que concentra el aire y lo expulsa a través de una rejilla hacia un tubo plástico para elevar una pieza de cartón prensando ubicada en su interior hasta la altura indicada por el usuario en una interfaz desarrollada por medio de una App de la plataforma de Blynk, que cuenta con: una entrada de texto numérico para determinar el setpoint deseado; un medidor de porcentaje del valor de PWM que controla el ventilador; valor que es entregado por el tipo de controlador seleccionado; y un histograma que refleja el setpoint ingresado y la variación de la altura de la pieza de cartón a lo largo del tiempo. El sistema se implementa sobre una Raspberry Pi 4, en donde se realiza todo el procesamiento de: los algoritmos de control, el flujo de datos establecidos con una tarjeta de desarrollo NODE MCU, con la que se comunica la planta, y con la aplicación de Blynk.

ABSTRACT

This work shows the procedure carried out to implement the LQG, LQI and PID controllers through WNCS applied to a Pneumatic Levitator, and the comparison of the stabilization time between them, whose plant consists of a base made of wood with a hole in the middle, and in this a fan is located at the bottom part, which is responsible for sending air to a funnel placed in the upper part, which concentrates the air and expels it through a grill towards a plastic tube to raise a pressed paperboard piece located inside it to the height indicated by the user in an interface developed through an App of the Blynk platform, which has: a numerical text input to determine the desired setpoint ; a percentage meter of the PWM value that controls the fan; value that is delivered by the selected controller type; and a histogram that reflects the entered setpoint and the variation in the height of the paperboard piece over time. The system is implemented on a Raspberry Pi 4, where to realize all the processing of: the control algorithms; the data flow established with a NODE MCU development card, with which the plant communicates, and with the Blynk application.

INTRODUCCIÓN

En las industrias existen plantas que necesitan controlar sus procesos mediante la implementación de un algoritmo de control que corrija en tiempo real su funcionamiento para que los procedimientos llevados por la planta sean precisos. En la actualidad con la innovación del IIoT (Industrial Internet of Things) o industria 4.0, los procesos de una industria pueden ser observados y manipulados a través de internet desde cualquier parte del mundo por medio de una red WAN o dentro de la misma industria, sin necesidad de estar junto a la planta, por medio de una red WLAN, motivo por el cual se propone la implementación de un sistema WNCS comunicado mediante una red WLAN y cuya planta es un Levitador Neumático, la cual será manipulada por 3 diferentes algoritmos de control, 2 controles óptimos y 1 no óptimo, alojados en una tarjeta Raspberry Pi 4. Se implementaron estos controladores con el fin de determinar cuál es mejor control en base al tiempo de establecimiento que presenta la respuesta de la planta.

En el primer capítulo se presentan los antecedentes, entre los cuales se encuentran el planteamiento del problema, la justificación, y los objetivos, general y específicos.

El segundo capítulo trata el marco teórico necesario para la comprensión e implementación de los algoritmos de control desarrollados para manipular la planta.

En el tercer capítulo se habla acerca del proceso de diseño de la planta, los controladores, el sistema de comunicación y la interfaz de control.

El cuarto capítulo contiene las pruebas realizadas en la planta real, para evaluar los 3 controladores implementados, en base al tiempo de establecimiento que tuvo la planta con diferentes setpoints, con el fin de determinar el mejor.

En el quinto capítulo se exponen las conclusiones y recomendaciones determinadas por los autores del presente proyecto, a partir de los resultados mostrados en el cuarto capítulo.

CAPÍTULO 1
ANTECEDENTES

1.1 Planteamiento del problema

Las industrias requieren que sus procesos sean precisos, repetitivos, y adaptativos, puntos resueltos mediante controladores programables como PLC o tarjetas de desarrollo que cumplan con características de robustez propias de ambientes industriales apostando por la precisión y resistencia al ruido que los algoritmos de control como Fuzzy, LQI, LQG, entre otros, pueden brindar (Banerjee & Pal, 2018). Esto añadido el avance en redes para procesos de control, genera un campo abierto a la aplicación e incorporación de controladores como los controladores óptimos y su adaptación a un medio compartido que permita su gestión y administración empleando una red inalámbrica a la par con el avance tecnológico y beneficios que las herramientas IoT proveen (Toro, Sánchez, Strefezza, & Granado, 2017).

Siendo de esta manera necesaria la comparación entre diferentes controladores para aportar a la base de conocimiento de la comunidad a la que va orientada esta investigación, la cual concierne a desarrolladores o estudiantes en el área de teoría de control con vistas a mejorar procesos que requieren la aplicación de sistemas de control basada en la nube o sobre una red wireless. Y así, se busca ampliar y brindar un panorama más claro de esta tecnología aplicada en la industria por medio de la elaboración de un prototipo.

Por tal motivo se pretende implementar los controladores óptimos LQG y LQI, y el controlador PID sobre un sistema WNCS para el control del levitador neumático, y comparar sus tiempos de establecimiento al llevarlos desde condiciones iniciales iguales hasta una misma referencia y variándola sobre la marcha, con el fin de determinar el controlador que presenta el mejor desempeño en términos de tiempo, y cómo influye en este, la implementación sobre una red inalámbrica local establecida para permitir la interacción con el usuario mediante una App (aplicación móvil).

1.2 Justificación

En la actualidad gracias al avance tecnológico y a las herramientas que de este se derivan, se ha concebido una nueva área de estudio y aplicación para los dispositivos controlados sobre una red inalámbrica. Esta área empezó con la idea de ir un paso más allá de la ya existente domótica, la cual se basa en la automatización de los implementos de los hogares para crear hogares inteligentes, por lo que el siguiente paso fue la automatización de los hogares por medio de comunicaciones en red local o remota, lo cual se conoce como IoT (Internet of Things) (Madakam, 2015). Una aplicación que nació de esta tecnología fue crear una industria inteligente que sea capaz de manejarse con la menor intervención humana posible mediante una conectividad wireless, denominada IIoT (Industrial Internet of things), ya que se emplean sensores y actuadores inteligentes con la capacidad de conectarse a una red, permitiendo expansión en el área de control (Sreenivasulu & Chalamalasetti, 2019).

Con la llegada del IIoT, y los controladores basados tecnología wireless, se pueden optimizar dos factores claves: la complejidad de implementación, monitorización y administración de los sistemas implementados in situ mediante herramientas IoT/IIoT; y el tiempo muerto en caso de fallo, dado que se puede pasar el algoritmo de control a un dispositivo con capacidad de conectarse a una red local o nube, como un servidor, permitiendo respaldar el mismo, y al tener mayor capacidad de procesamiento debido a la escalabilidad propia de los ambientes wireless (Al-Dabbagh & Chen, 2016), en comparación a los sistemas embebidos tradicionales que emplean medios cableados, estos pueden ser reemplazados, simplemente, por tarjetas DAQ con capacidad de transmitir y recibir datos por medio de una red LAN (Sreenivasulu & Chalamalasetti, 2019), representando un ahorro para la empresa, debido al bajo costo de su instalación y manejo en campo frente a su contraparte cableada. A su vez, disminuye la complejidad de mantenimiento, puesto que el algoritmo de control se encuentra respaldado en la nube o medio compartido y el hardware empleado se reduce, limitando esto en su mayoría a problemas de software (Park, Ergen, Fischione, & Lu, 2018).

1.3 Objetivos

1.3.1 Objetivo general

Implementar algoritmos de control óptimo mediante un WNCS para el manejo de un levitador neumático.

1.3.2 Objetivos específicos

- Revisar el estado del arte referente a controles óptimos y su aplicación sobre WNCS por medio de artículos y libros relacionados para la familiarización con este tipo de sistemas de control.

- Realizar la planta de un levitador neumático mediante un sistema embebido para la obtención de su función de transferencia.

- Desarrollar una App mediante herramientas IoT para la monitorización del estado del sistema y la asignación de la referencia de control.

- Elaborar una tabla comparativa de algoritmos de control mediante la medición variables del sistema para la determinación del mejor controlador.

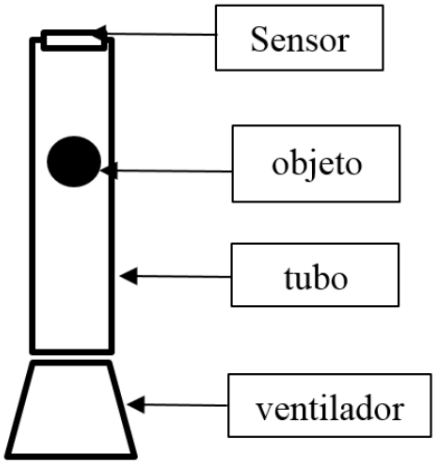
CAPÍTULO 2
MARCO CONCEPTUAL

En este capítulo se abordan conceptos necesarios para el desarrollo del presente proyecto, en los cuales se abarcan los algoritmos de control PID, LQG, LQI, los conceptos relacionados con IoT, y WNCS.

2.1 Levitador Neumático

Un levitador neumático es un dispositivo electromecánico cuyo propósito es elevar un objeto a partir de la variación del flujo de aire inducido mediante un ventilador (Colín Rivas, García Mejía, & Flores Fuentes, 2020). Como se muestra en la Figura 2.1, la arquitectura de un levitador neumático está constituida por un sensor de distancia, el objeto a levitar, un tubo por el cual se desplazará el flujo de aire, un ventilador, y un sistema embebido que controla el proceso.

Figura 2.1. Levitador Neumático.



Arquitectura básica de un Levitador Neumático, Fuente: (Colín Rivas, García Mejía, & Flores Fuentes, 2020)

2.2 Sistema de control

Para definir lo que es un sistema de control es necesario determinar el “control” como la manipulación de una variable para que otro alcance un valor deseado, de esta forma se puede definir a un sistema de control como un sistema en el cual al ingresar objetivos determinados responderá con una serie de actuaciones para alcanzar dichos objetivos (Ñeco Garcia, Reinoso García, García Aracil, & Aracil Santonja, 2003).

2.3 Controlador PID

El controlador PID (Proporcional Integral Derivativo) es un método de control en lazo cerrado de uso extendido a nivel industrial, este controlador logra corregir el error existente en estado estacionario a lo largo del tiempo, mediante el cálculo del error entre el valor medido y el valor deseado (Cova, CONTROL PID UN ENFOQUE DESCRIPTIVO, 2005).

La componente proporcional del controlador logra que su salida oscile alrededor del valor deseado, sin embargo, no corrige el error en estado estacionario. La componente integral logra corregir el error en estado estacionario mediante la suma de los errores anteriores, logrando que el error actual disminuya en el tiempo. La componente derivativa actúa como una predicción del error permitiendo que la acción de control tenga una mayor rapidez (Cova, 2008).

2.4 Controlador LQG

El controlador LQG (Lineal Cuadrático Gaussiano) mostrado en la figura 2.2 es una técnica de control óptimo cuya función es minimizar un índice de desempeño cuadrático al igual que el coste de esfuerzo del controlador en presencia de ruido y desviaciones del modelo, este controlador surge de la combinación de un regulador cuadrático lineal (LQR) junto a un estimador óptimo cuya función es reconstruir el vector de estado a partir de medidas distorsionadas con ruido (Beauchamp Báez & Batista, 2016).

Figura 2.2. Controlador LQG

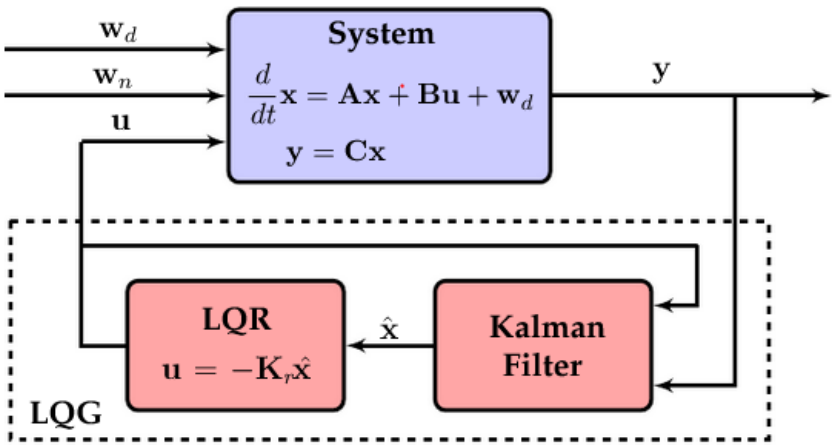


Diagrama de bloques del controlador LQG, Fuente: (Brunton & Kutz, 2019)

2.4.1 Regulador cuadrático lineal (LQR)

El regulador cuadrático lineal (LQR) es un método de control con el cual se asegura la estabilidad de un sistema en lazo cerrado por medio de ganancias de realimentación, buscando minimizar la función de costo definida como la sumatoria de desviaciones de los valores reales medidos respecto a los ideales, para mediante este proceso encontrar las ganancias del controlador que minimicen las desviaciones no deseadas (González Uribe, 2016).

2.4.2 Filtro de Kalman

El algoritmo de filtro de Kalman es un estimador lineal cuadrático (LQE), que se utiliza para la actualización de la proyección lineal de un sistema variable en el tiempo, mediante la observación del conjunto de información existente en cada instante, mientras se va actualizando (Novales, 2017). Este filtro es empleado para controlar sistemas sometidos a ruido blanco aditivo, resultando estadísticamente óptimo a comparación de otras funciones cuadráticas para estimación de errores, para tener una mayor precisión de la estimación se debe tener un modelo de la planta coherente con el comportamiento real de la planta (Dorado & Ruíz).

El filtro de Kalman sigue dos pasos para la estimación de la variable mediante la información de los datos medidos, primero realiza la predice el estado del sistema, y posteriormente incorpora las observaciones recogidas ya corregidas, siendo un algoritmo recursivo, no necesita reformularse para añadir nuevas observaciones (Munuera Raga, 2018).

2.5 Controlador LQI

El controlador LQR presenta la desventaja de necesitar que el modelado de la planta sea perfecto con el fin de satisfacer la dinámica del proceso, por lo que en caso de no lograr un modelado que cumpla dicha característica se producirán errores de posición o de estado estacionario, para solucionar estos errores se debe agregar una realimentación integral creando un estado dentro del controlador que calcule la integral de la señal del error para utilizarla como un término de realimentación, como se muestra en la figura 2.3 (Cargua Abril & Gallegos Herrera, 2017).

Figura 2.3. Controlador LQI

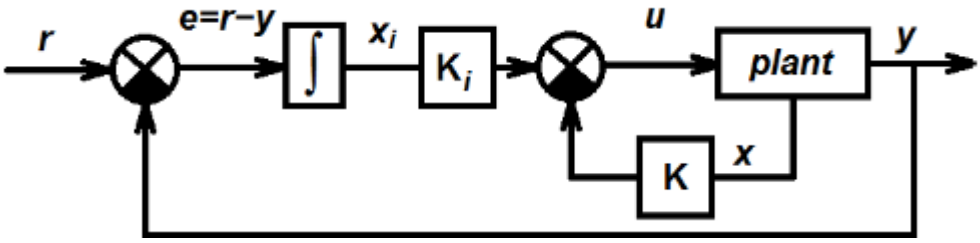


Diagrama de bloques del controlador LQI, Fuente: (Kisszölgyémi, Beneda, & Faltin, 2017)

2.6 WNCS

Los sistemas de control necesitan de un medio sobre el cual desarrollar su comunicación para lograr el envío de los datos del proceso obtenidos por los sensores hacia el controlador y a su vez enviar los valores emitidos por el controlador hacia el actuador para el funcionamiento de la planta, por lo que, con el desarrollo de protocolos de comunicación inalámbrica cada vez más robustos, se ha optado por utilizar medios de comunicación inalámbrica para el intercambio de datos dentro del sistema de control, por lo tanto se conoce como Wireless Network Control Systems (WNCS) a los sistemas de control en lazo cerrado que utilizan redes inalámbricas para la transmisión de datos, logrando de esta forma sortear limitaciones de distancia y accesibilidad (Pillajo & Hincapié, 2018).

2.7 NODE-RED

Node-RED es una herramienta de programación basada en flujos que fue diseñada originalmente por el equipo de desarrollo de IBM en 2013, sin embargo, en la actualidad es parte de la fundación JS. La programación basada en flujos es una forma para describir el comportamiento de una aplicación como una red de nodos, en la cual cada nodo tiene un propósito definido, por lo tanto, una vez se entrega información al nodo este realiza su función con esa información y la envía hacia el siguiente nodo, esta herramienta es ampliamente utilizada en aplicaciones web, IIoT, entre otros, utilizando diferentes protocolos de comunicación, como por ejemplo MQTT (OpenJS Foundation, 2021).

2.8 Blynk

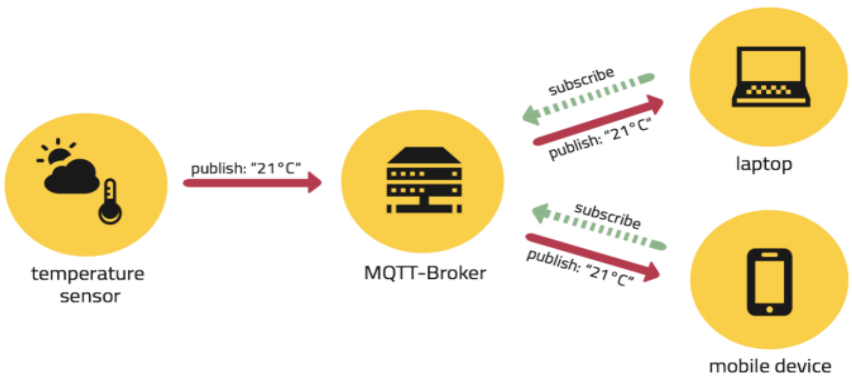
Blynk es un conjunto de software creado para realizar prototipos, implementar y administrar de manera remota los dispositivos electrónicos conectados, permite la

conexión de hardware con la nube y la creación sin necesidad de código de aplicaciones web y móviles tanto para Android como IOS, para analizar la información proveniente de los dispositivos en tiempo real e información histórica, puede configurar permisos y roles para los usuarios dentro de la aplicación. Las aplicaciones realizadas con Blynk pueden ser utilizadas directamente por el usuario final (Blynk, 2021).

2.9 Protocolo MQTT

El protocolo MQTT (Message Queing Telemetry Transport o Transporte de telemetría de cola de mensajes) mostrado en la Figura 2.4 es un estándar de OASIS (Organización para el Avance de los Estándares de Información Estructurada), diseñado por Andy Stanford y Arlen Nipper en 1999, para la conectividad IoT. Es un protocolo de mensajería de publicación/suscripción por "topic" o tema, los cuales presentan jerarquías para agrupar y seccionar los objetivos de lectura y escritura, que busca ser extremadamente simple y liviano, pensado para dispositivos con limitaciones de recursos, y redes con ancho de banda bajo, latencia alta, o inestables. Los principios antes mencionados a su vez buscan garantizar la confiabilidad y entrega de información, por lo que estos principios lo convierten en el protocolo ideal para el mundo de los dispositivos en el internet de las cosas (MQTT, 2020).

Figura 2.4. Arquitectura protocolo MQTT



Arquitectura de la comunicación mediante protocolo MQTT, Fuente: (Hita Albarracín, 2020).

2.9.1 Mosquitto Broker

Mosquitto es un bróker de mensajería open source que implementa el protocolo MQTT, es liviano y apropiado para utilizarlo en dispositivos desde computadoras de placa simple de bajo poder, hasta servidores, también provee una librería en C para implementar clientes MQTT. Mosquitto es parte de la fundación Eclipse, y es un proyecto IoT financiado por cedalo.com (Eclipse Foundation, 2019).

CAPÍTULO 3
DISEÑO E IMPLEMENTACIÓN

En el presente capítulo se muestran el diseño e implementación de todos los componentes del sistema utilizado en el proyecto, tanto hardware como software, siguiendo el diagrama de la Figura 3.1, en el cual se indica el funcionamiento del sistema con las diferentes partes que lo conforman.

Figura 3.1. Diagrama de funcionamiento del sistema.

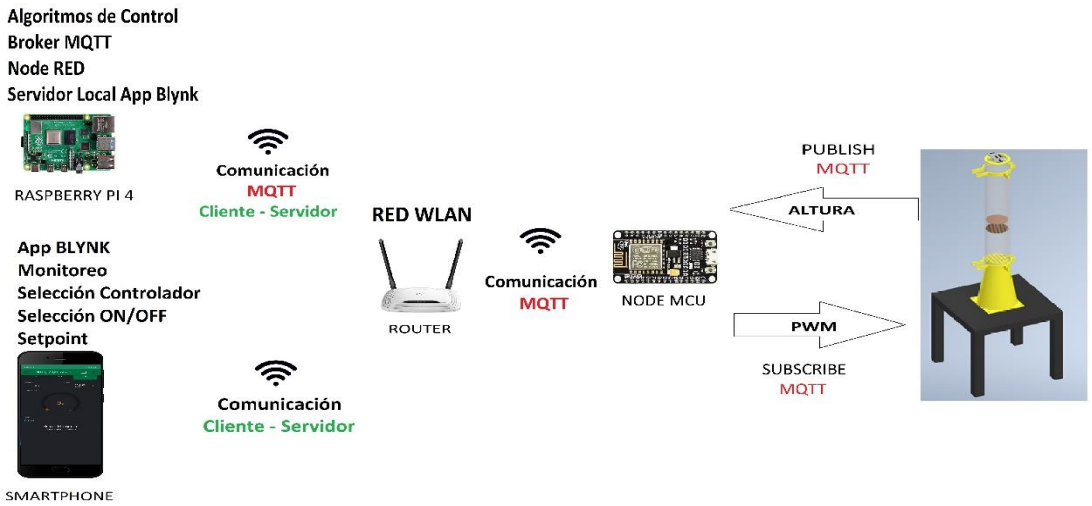
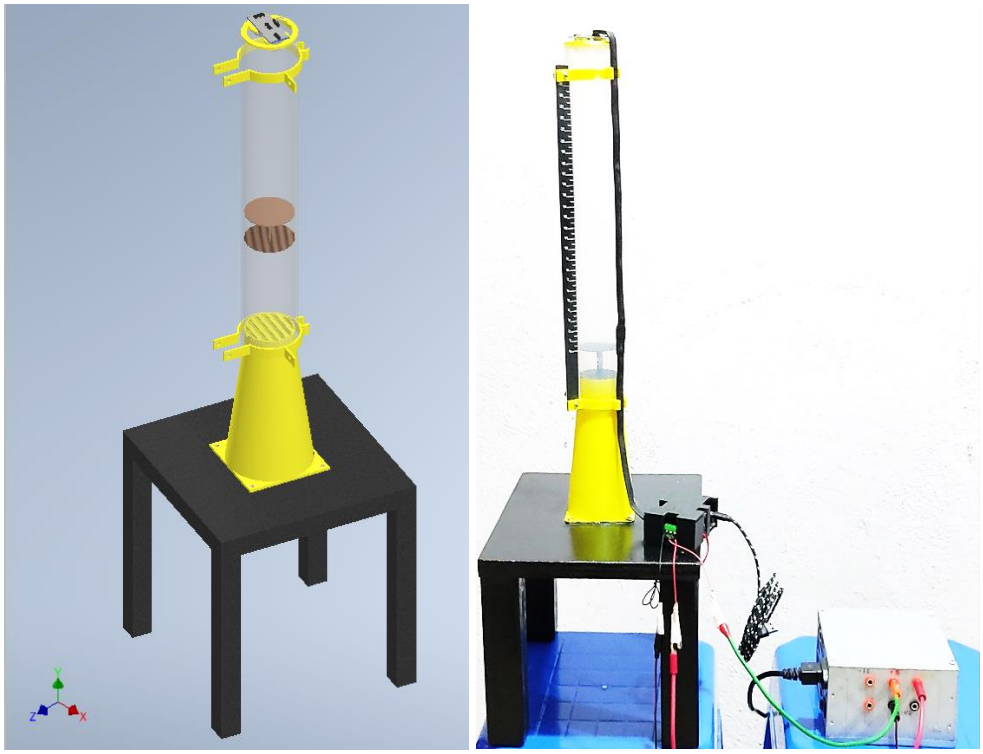


Diagrama explicativo del funcionamiento del sistema, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1 Diseño de la planta

La planta diseñada, que se muestra en la Figura 3.2, consta de una base en la que un ventilador colocado en su cara inferior sea suficientemente abastecido de aire, que, a través de un embudo contiguo y una rejilla, concentre el flujo de aire y se logre elevar una pieza de cartón sin turbulencia dentro de un tubo embonado. El diseño y medidas de cada pieza que conforman únicamente la planta se presentan del ANEXO I al ANEXO VIII.

Figura 3.2. Levitador Neumático implementado.



a) b)

a) Levitador Neumático diseñado en el software Inventor.

b) Planta de Levitador Neumático real, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.1 Mesa o base

La planta se sostiene por una mesa elaborada de madera con forma cuadrada, la cual se puede apreciar en la Figura 3.3. Cuenta con una cavidad circular central equivalente al diámetro interno del ventilador y orificios para la fijación tanto del embudo como del ventilador por medio de tornillos.

Figura 3.3. Base de la planta.

Base construida con madera para Levitador Neumático, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.2 Embudo

En la Figura 3.4 se presenta el embudo, el cual que se encarga de concentrar el flujo de aire proveniente del ventilador para lograr la suficiente presión de aire que permita elevar el objeto a levitar en el tubo.

Figura 3.4. Embudo de PLA impreso.

Embudo diseñado en software Inventor, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.3 Rejilla

El propósito de esta pieza, mostrada en la Figura 3.5, es disminuir la turbulencia generada dentro del embudo para que el flujo de aire que ingrese al tubo logre elevar la pieza con un flujo laminar y no la desplace de manera caótica.

Figura 3.5. Rejilla de PLA impresa.

Rejilla diseñada en software Inventor, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.4 Tubo

Se consideró un tubo plástico transparente, adquirido comercialmente, como elemento limitante, debido a que, en base a este elemento, mostrado en la Figura 3.6, se determinaron el rango de trayectoria máximo del objeto a levitar y las dimensiones para los demás elementos de la planta.

Figura 3.6. Tubo plástico transparente.

Tubo plástico transparente adquirido, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.5 Abrazadera

Como una forma de medir de manera física la altura del objeto a levitar dentro del tubo, se diseñó la abrazadera mostrada en la Figura 3.7, la cual dispone de un alargamiento lateral con un orificio para colocar una cinta métrica impresa, un saliente en forma de “C” para mantener derecho el cableado de sensor y un pico de sujeción.

Figura 3.7. Abrazadera de PLA impresa.

Abrazadera diseñada en software Inventor para sujeción de cinta métrica y cables de sensor, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.6 Soporte para sensor

Dado que el tubo utilizado para la planta tiene un diámetro interno mayor a la longitud del sensor utilizado en el proyecto y con el fin de mantenerlo fijo y evitar desviaciones, se diseñó un soporte, presentado en la Figura 3.8, que, internamente, se ajusta al emisor y receptor del sensor, y, en su periferia, encaja en el contorno de la pared del tubo.

Figura 3.8. Soporte de PLA para sensor ultrasónico impreso.

Soporte para sensor ultrasónico HC-SR04 diseñado en software Inventor, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.7 Objeto a levitar

Se decidió elaborar este elemento con cartón corrugado, como se muestra en la Figura 3.9, ya que se necesita que sea liviano y ligeramente flexible, de manera que abarque todo el diámetro interno del tubo y tenga la capacidad de deslizarse por sus paredes sin atorarse, además, debe poseer una superficie plana en su parte superior para evitar errores de medición.

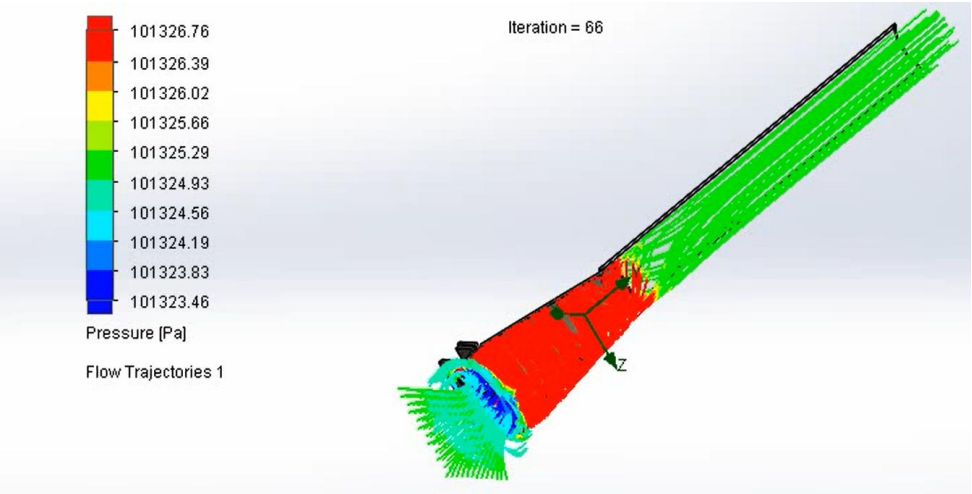
Figura 3.9. Objeto a levitar de cartón corrugado.

Objeto a levitar diseñado en software Inventor, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.8 Simulación del comportamiento del flujo de aire en planta

Se tomó el diseño de los diferentes elementos que conforman la planta, para realizar una simulación del flujo de aire en la misma, mediante el software SolidWorks, teniendo en cuenta las especificaciones del ventilador empleado, para determinar si las piezas ensambladas cumplen con el comportamiento esperado para el desarrollo del proyecto. Esta simulación se muestra en la siguiente figura.

Figura 3.10. Simulación en SolidWorks del flujo de aire en la planta.



Visualización del flujo y presión que genera el aire en la planta en el software SolidWorks, Elaborado por: Ordóñez Wilson y Pino Wellington.

Como se puede apreciar en la Figura 3.10, el aire impulsado por el ventilador logra una mayor presión dentro del embudo, sin embargo, presenta turbulencia. Y al pasar a través de la rejilla, el flujo de aire pierde presión, pero esta sigue siendo mayor a la generada en el ventilador, a la vez que el flujo de aire se vuelve laminar al ascender por el tubo plástico. Es decir, el diseño del embudo permite al mismo comportarse como un contenedor de presión debido a su alargada figura, presión que es suficiente para impulsar la pieza hacia arriba. Y la rejilla, por medio de sus rejas, reduce la presión y le da forma al flujo de aire, convirtiendo el flujo de turbulento a laminar inmediatamente.

3.1.9 Sensor ultrasónico HC-SR04

La medición de la altura del objeto a levitar se realizó a través de un sensor ultrasónico HC-SR04 mostrado en la Figura 3.11, el mismo que posee las características de la Tabla 3.1.

Tabla 3.1. Características sensor HC-SR04.

Característica	**Valor**
Voltaje de trabajo	3.3V~5V
Corriente de trabajo	15mA
Frecuencia de trabajo	40KHz
Rango máximo de medición	4m
Rango mínimo de medición	2cm
Pines	4 (VCC, Trig, Echo, GND)
Ángulo efectivo de medición	15°
Resolución	0.3cm
Dimensiones	45mm*20mm*15mm

Características de funcionamiento del sensor ultrasónico HC-SR04, Fuente: (ELECFreaks, 2013)

Figura 3.11. Sensor ultrasónico HC-SR04.

Sensor de distancia ultrasónico HC-SR04, Fuente: (Mouser Electronics, 2019)

3.1.10 Ventilador

Para generar el flujo de aire necesario para elevar la pieza dentro del tubo se eligió el ventilador que se muestra en la Figura 3.12, con las características de la Tabla 3.2.

Tabla 3.2. Características ventilador VN-351 DC.

Característica	**Valor**
Voltaje de operación	12V
Corriente de operación	180mA
Velocidad máxima	2800rpm
Dimensiones	80mm*80mm*2.5mm

Características de funcionamiento del ventilador modelo VN-351, Fuente: (PCJAM20, 2021)

Figura 3.12. Ventilador DC modelo VN-351.

Ventilador DC modelo VN-351, Fuente: (PCJAM20, 2021)

3.1.11 NODE MCU V2 (ESP-12E module)

En la planta se tiene una tarjeta de desarrollo NODE MCU V2 (ESP-12E module) mostrada en la Figura 3.13, que posee el chip ESP8266 orientado a conexión wifi. Esta tarjeta está encargada de: obtener la altura de la pieza levitante, aplicar la señal PWM para el control del ventilador y comunicarse con Node-RED, para ello cuenta con las especificaciones de la Tabla 3.3.

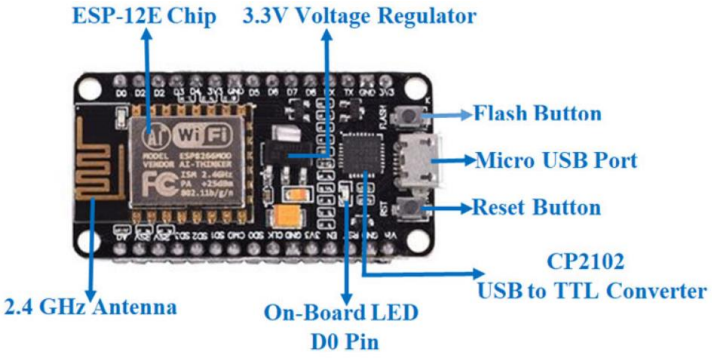
Tabla 3.3. Características tarjeta NODE MCU V2 (ESP-12E module).

Característica	**Valor**
Microcontrolador	Tensilica Xtensa LX106 RISC 32-bit
Voltaje de operación	3.3V
Voltaje de alimentación	5V-12V
Timers	1 FRC1 para modulación PWM 1 FRC2 para comunicación wifi
Salidas PWM	4
Resolución PWM	10bits
Pines I/O Digitales	16
Memoria ROM	64KB
Memoria Flash	4MB
Memoria SRAM de instrucciones	64KB

Velocidad de reloj	80MHz
Antena PCB	1
Ancho de banda conexión wifi	2.4GHz
Puerto USB (Alimentación/Programación)	1
Dimensiones	49mm*26mm*12mm

Características de la tarjeta NODE MCU V2 (ESP-12E module), Fuente: *(Espressif Inc, 2020)*

Figura 3.13. Tarjeta de desarrollo NODE MCU V2 (ESP-12E module).



Tarjeta de desarrollo NODE MCU V2 (ESP-12E module), Fuente: (Components 101, 2020)

3.1.12 Raspberry PI 4

Como servidor local para los algoritmos y la interfaz de control, y para establecer comunicación entre dispositivos, se escogió una Raspberry Pi 4 modelo B mostrada en la Figura 3.14, la misma que es un ordenador de placa simple (SBC) de bajo costo. En ella se alojaron los algoritmos de control dentro de la plataforma Node-RED, se instaló el bróker de comunicación MQTT y un servidor local para la comunicación con la App desarrollada en Blynk. Esta microcomputadora tiene una amplia gama de aplicaciones debido a sus características mostradas en la Tabla 3.4.

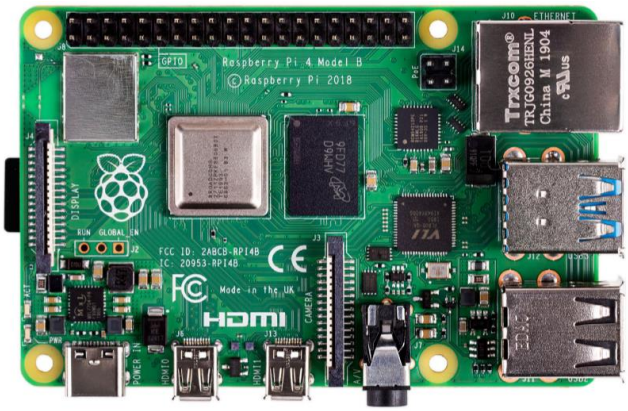
Tabla 3.4. Características Raspberry PI 4.

Característica	**Valor**
Procesador	Quad core Cortex-A72 (ARM v8) 64-bit
Velocidad de reloj	1.5GHz
Memoria SDRAM	4GB
Ancho de banda conexión wifi	2.4GHz/5GHz
Puertos USB 2.0/3.0	2/2

Puertos micro HDMI	2
Puerto micro USB tipo C (alimentación)	1
Puerto microSD (almacenamiento y OS)	1
Puerto Gigabit Ethernet	1

Características de trabajo Raspberry PI 4 modelo B, Fuente: (Raspberry Pi Trading Ltd, 2021)

Figura 3.14. Raspberry PI 4 modelo B.

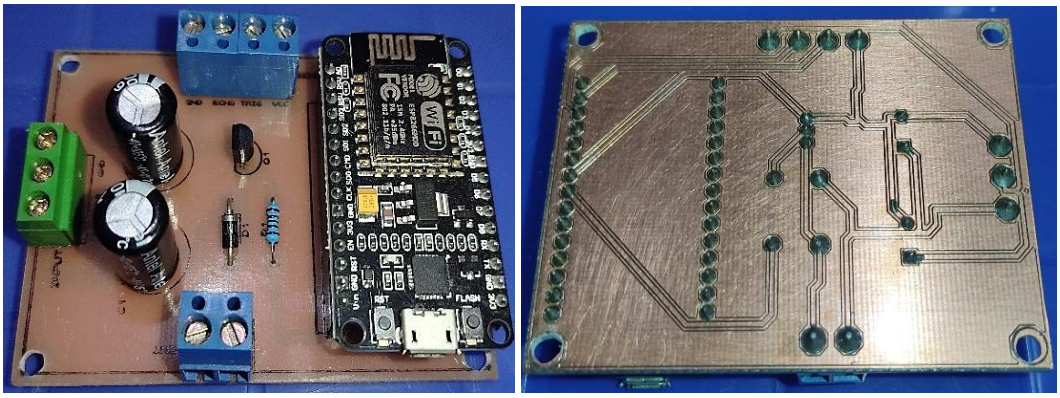


Computador Raspberry PI 4 modelo B, Fuente: (Raspberry Pi Trading Ltd, 2021)

3.1.13 PCB de circuitería local

El desarrollo de la circuitería para la ejecución de las acciones locales en la planta, se implementó mediante una tarjeta electrónica PCB, mostrada en la Figura 3.15, que consta de: dos condensadores para filtración del voltaje, un transistor 2N2222 para controlar el voltaje medio en el ventilador, un diodo 1N4007 en configuración flyback, una resistencia de 1kΩ para limitar la corriente desde el pin PWM del NODE MCU hacia la base del transistor, un sócalo para la colocación del NODE MCU, y borneras para la alimentación de voltaje, conexión del sensor HC-SR04 y ventilador DC modelo VN-351. El diseño de esta placa PCB se muestra en el ANEXO IX.

Figura 3.15. Tarjeta PCB.



a) b)

a) Vista superior de elementos de PCB en físico

b) Vista inferior de pistas de PCB en físico, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.14 Carcasa para PCB

Debido a las vulnerabilidades a las que se encuentra expuesta la placa PCB, se necesitó implementar una carcasa de protección que esté acorde a los espacios de conexión del PCB, como se muestra en la Figura 3.16. El diseño de está carcasa se presenta en el ANEXO X.

Figura 3.16. Carcasa para placa PCB.

Carcasa diseñada en software Inventor para protección de la placa PCB, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.1.15 Fuente de alimentación

La alimentación de voltaje del sensor y ventilador de la planta viene dada por una fuente ATX genérica de gama baja, con las especificaciones de la Tabla 3.5.

Tabla 3.5. Características fuente ATX genérica.

Característica	**Valor**
Voltajes suministrados	±12VDC, +5VDC, +3.3VDC, GND
Corriente suministrada	12VDC – 24A, 5VDC – 40A, 3.3VDC – 30A
Potencia de la fuente	650 W

Especificaciones de la fuente de alimentación ATX utilizada en la planta, Elaborado por: Ordoñez Wilson y Pino Wellington.

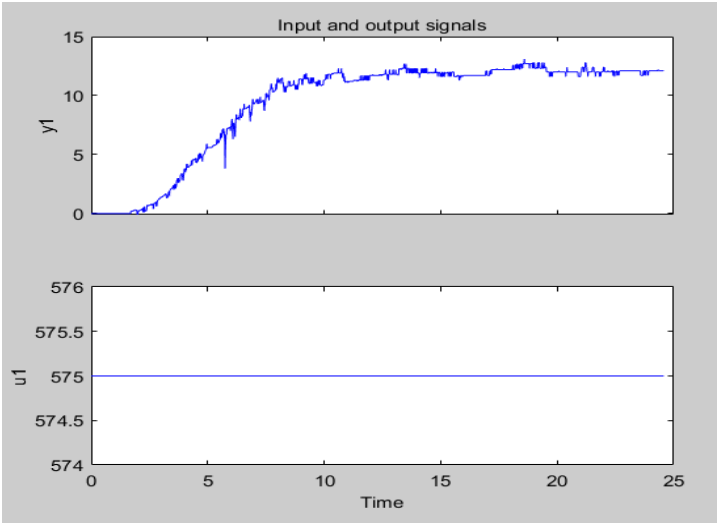
3.2 Obtención de la función de trasferencia

Para el desarrollo del presente proyecto se requiere la obtención de la función de transferencia que determina el comportamiento de la planta, para lo cual se aplicó el método de caja negra, que consiste en colocar un valor a la entrada, observando el comportamiento de la salida de la planta, para de esta forma, y con la utilización de la herramienta System Identification de Matlab determinar una función de transferencia que describa con mayor aproximación el funcionamiento de la planta. Esta función debe tener una aproximación mínima del 85% con respecto a la curva observada a la salida de la planta (Chalán Padilla, 2020).

Siguiendo este método se tiene como entrada una señal PWM que varía el voltaje medio en el ventilador de la planta controlando su velocidad, mientras que la salida es el desplazamiento de altura del objeto a levitar en el interior del tubo conectado al ventilador. Para lo cual se utilizó una entrada constante de 575, equivalente al 56.21% del valor del ciclo de trabajo máximo de la señal PWM que entrega el NODE MCU ya que tiene una resolución de 1023.

A continuación, se ingresaron los datos de entrada y salida obtenidos, en la herramienta System Identification de Matlab, con un tiempo de muestreo de 13ms, el cual es el mismo con el que se obtuvieron los datos de altura de la planta. Obteniendo así las curvas mostradas en la Figura 3.17.

Figura 3.17. Graficas de entrada y salida de la planta.

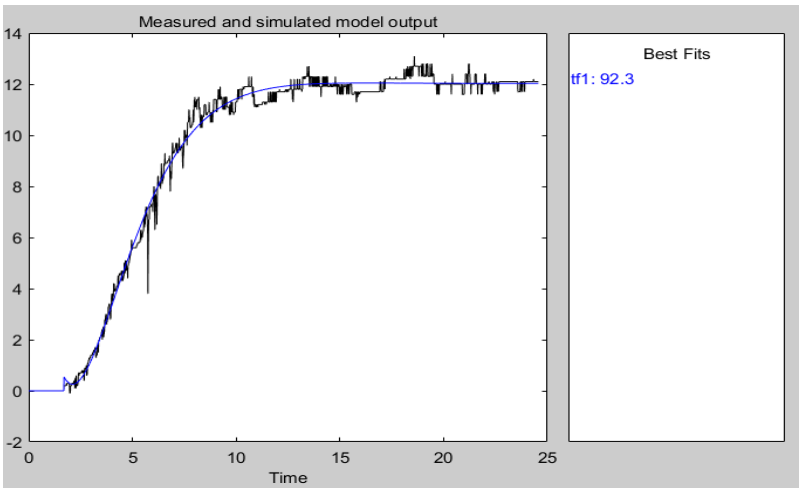


Graficas de salida y entrada obtenidas de la planta e ingresadas en la herramienta System Identification de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

Posteriormente, se procede con la estimación de la función de trasferencia con la opción "Transfer Function Models" de la herramienta System Identification, colocando dos polos, dos ceros, y un retardo de 1.7seg, para obtener el mayor porcentaje de aproximación posible a la curva de salida.

Observando los resultados que arroja la estimación, se tiene una aproximación de 92.3%, tal y como se muestra en la Figura 3.18. Y se presenta la función de transferencia estimada en la Figura 3.19.

Figura 3.18. Curva real y de estimación.



Curva real de la planta vs Curva de estimación obtenida por medio de la herramienta System Identification de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

Figura 3.19. Función de Transferencia.

```
From input "u1" to output "y1":
                0.0009489 s^2 - 0.001735 s + 0.005401
exp(-1.7*s) * -------------------------------------
                        s^2 + 0.9158 s + 0.2582

Name: tf1
Continuous-time identified transfer function.
```

Función de transferencia identificada por medio de la herramienta System Identification de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

Cabe mencionar que se realizó la medición de la latencia dentro de la red WLAN utilizada en la comunicación entre los dispositivos que intervienen en el proceso de control, que, sumada a los retardos propios de los sistemas embebidos, dio como resultado un retardo de 4ms. Y debido a que se implementó por programación un filtro de promedio de muestras para minimizar el error de medida del sensor ultrasónico, finalmente se empleó un tiempo de muestreo de 102 milisegundos para el diseño de los controladores y para el funcionamiento del sistema en general.

3.3 Diseño de controlador PID

Con la función de transferencia obtenida previamente, se procede a realizar la simulación del controlador PID mediante la herramienta Simulink de MATLAB, desarrollando el siguiente diagrama de bloques mostrado en la Figura 3.20.

Figura 3.20. Diagrama de Bloques de control PID.

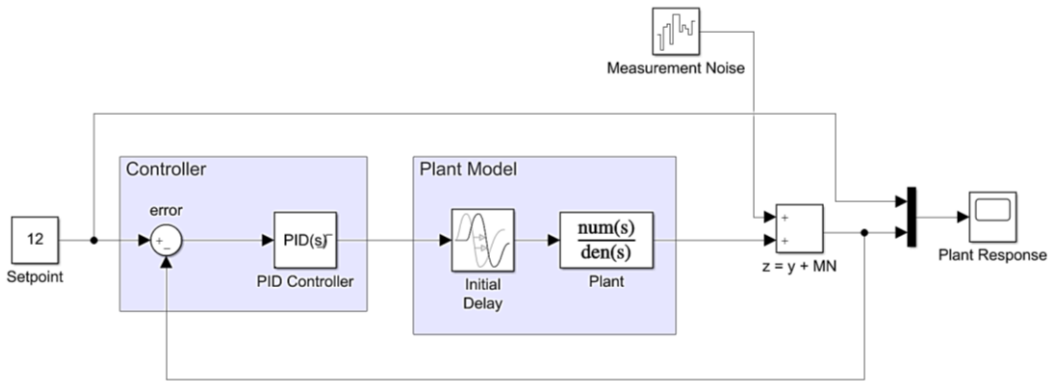
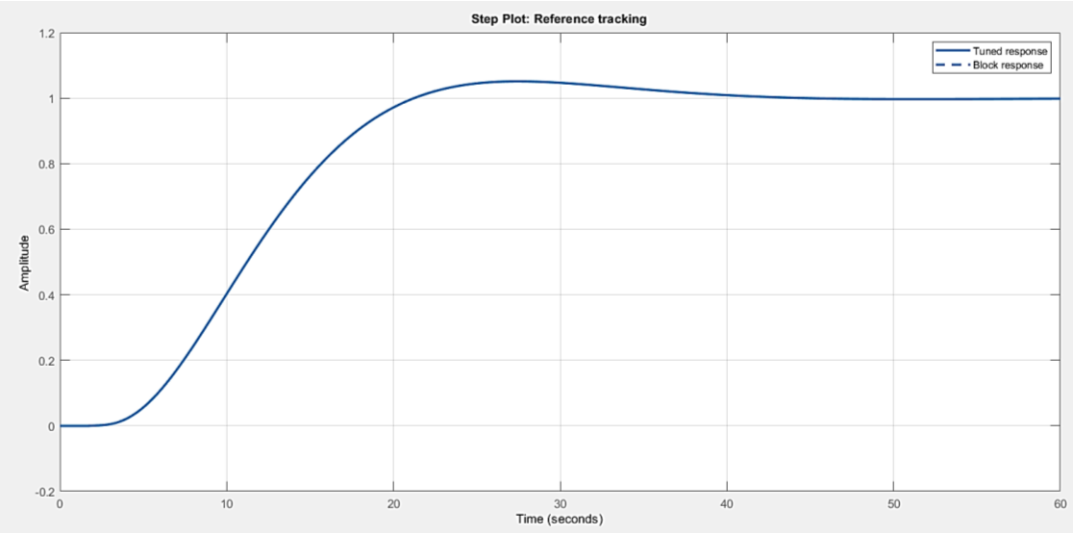


Diagrama de Bloques de control PID en Simulink de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

En este diagrama de bloques se tiene un bloque PID controller, y un bloque de retardo o “delay” en serie con un bloque que contiene la función de transferencia, que en

conjunto representan a la planta. Al desplegar la pantalla con las propiedades del bloque PID Controller se puede acceder a la herramienta PID Tunner, la misma que tomará automáticamente el diagrama de bloques de la simulación para realizar la autosintonización del controlador PID, obteniendo la respuesta que se observa en la Figura 3.21.

Figura 3.21. Respuesta de la planta a control PID.



Respuesta de la planta al escalón unitario para el control PID en la herramienta PID Tunner de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

Validando en la planta real los parámetros que arrojó la herramienta PID Tunner de MATLAB como punto de referencia, se realizó un ensayo de prueba y error para obtener la mejor respuesta en la planta real, dando como resultado las siguientes constantes para el controlador PID: Kp = 15, Ki = 0.1, Kd = 80.

3.4 Diseño de controlador LQG

Para el desarrollo del controlador LQG, se debe tomar la función de transferencia obtenida previamente, y mediante el uso del software MATLAB, realizar un script que determine las matrices de espacio de estados, ingresando la función de transferencia como numerador y denominador, para convertirla a espacio de estados y posteriormente, discretizar el resultado, el mismo del que se extraen las matrices de estado A, B, C, D, como se muestra en la Figura 3.22 y 3.23.

Figura 3.22. Script para obtener matrices de espacio de estados.

```
clc;
% Transfer function
num = [9.489496760496952e-04,-0.001735203602620,0.005401218766916];
den = [1,0.915775407922508,0.258234581040331];
tf2 = tf(num,den);

% Transfer function to space states
ss1 = ss(tf2);
sysd = c2d(ss1,0.102);
[A,B,C,D] = ssdata(sysd);
```

Script desarrollado en MATLAB para el cálculo de las matrices de espacio de estados, Elaborado por: Ordóñez Wilson y Pino Wellington.

Figura 3.23. Matrices de espacio de estados.

```
A =

    0.9096   -0.0503
    0.0487    0.9987

>> B

B =

    0.0122
    0.0003

>> C

C =

   -0.0208    0.0825

>> D

D =

   9.4895e-04
```

Matrices de Espacio de Estados obtenidas en script de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

A partir de este punto, se establece una matriz identidad como matriz Q de igual dimensiones que la matriz de estado A, y un valor de referencia para la matriz R, la cual en este caso es una matriz de un solo elemento debido a la cantidad de entradas del sistema. Para la obtención de estas matrices no existe un método definido como

menciona (Solórzano Peñafiel, 2018), por lo cual, se realizó la simulación mostrada en la Figura 3.24 y 3.25, siguiendo la arquitectura del control LQG.

Figura 3.24. Diagrama de Bloques de control LQG.

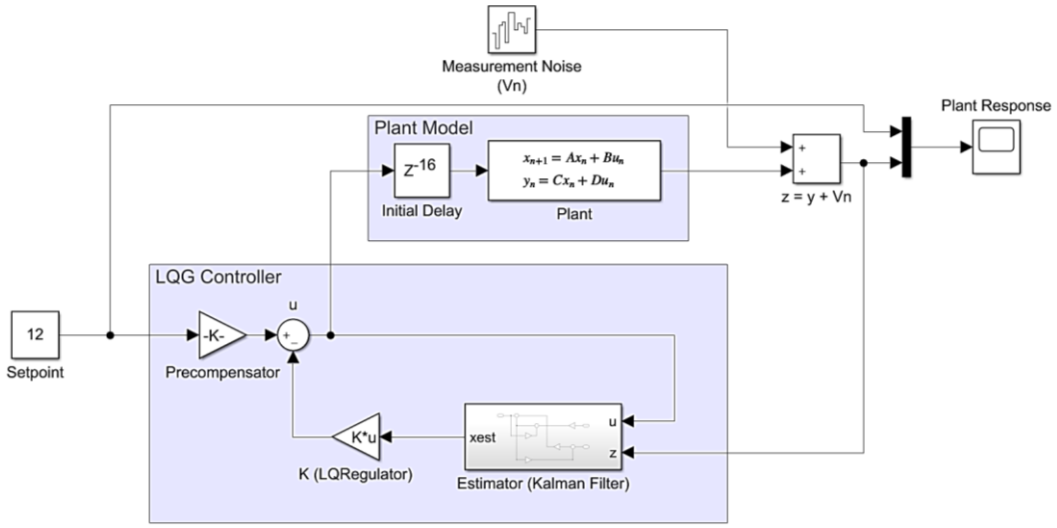
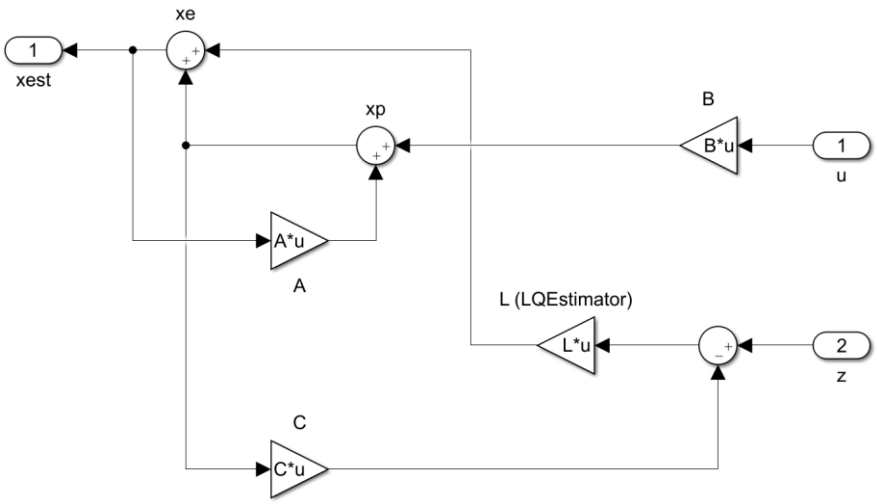


Diagrama de Bloques del control LQG en Simulink de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

Figura 3.25. Subsistema “Estimator (Kalman Filter)”.



Bloque “Estimator (Kalman Filter)” en Simulink de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

En la que, a base de prueba y error, se variaron los valores para las matrices Q y R al momento de introducirlos como parámetros dentro de la función *lqr(sys,Q,R)* de MATLAB empleada en el script desarrollado, como se muestra en la Figura 3.26, con el fin de determinar los valores de la matriz K del regulador LQR que cumplan con la

respuesta deseada de la planta. Cabe destacar que esta función se puede utilizar tanto en tiempo continuo como en discreto.

Figura 3.26. Script para la obtención de Matrices Q, R y K – Control LQG.

```
% Q and R matrix
Q = [1 0;
    0 0.0001];
R = 2;

% LQR regulator
[K,S,CLP] = lqr(sysd,Q,R);
```

Script desarrollado en MATLAB para la obtención de las matrices de costo Q y R, y la matriz K del regulador LQR, Elaborado por: Ordóñez Wilson y Pino Wellington.

Y para la obtención de la matriz L del estimador LQE se emplean constantes de ruido de medida (Wn) y ruido de proceso (Vn), como parámetros de la función *dlqe(A,Gc,C,Wn,Vn)* de MATLAB utilizada en el script desarrollado, como se observa en la Figura 3.27, la cual trabaja únicamente en tiempo discreto.

Figura 3.27. Script para la obtención de la matriz L – Control LQG.

```
% LQE estimator
vn = 0.09;
wn = 0.025;
Gc = [0;1];
[L,P,aut] = dlqe(A,Gc,C,wn,vn);
```

Script desarrollado en MATLAB para la obtención de la matriz L del regulador LQR, Elaborado por: Ordóñez Wilson y Pino Wellington.

En estas constantes de ruido no se aplica una regla específica para su determinación, por lo cual, se ponderan sus valores observando cuanto influye su variación, en términos de estabilidad, en la gráfica de la respuesta de la planta en la simulación.
La matriz K del regulador LQR y la matriz L del estimador LQE del filtro de Kalman que se obtuvieron por medio de la simulación son mostradas en la Figura 3.28.

Figura 3.28. Matrices K y L – Control LQG.

```
>> K

K =

    0.0324   -0.0009

>> L

L =

   -0.1257
    0.3645
```

Matrices K del regulador LQR y L del estimador LQE del filtro de Kalman obtenidas por medio de la simulación en Simulink de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

Para el cálculo del precompesador se evaluó la planta en la simulación con un setpoint o referencia y un precompensador igual a 1, y sin ruido de medida, ya que, de esta manera, se puede calcular fácilmente el valor de ganancia necesario para alcanzar el punto de consigna al visualizarse la amplitud de estabilización dentro de un rango de 0 a 1. En este punto, la planta presenta una respuesta sin oscilaciones ni sobreimpulso pero con una ganancia baja como se observa en la Figura 3.29.

Figura 3.29. Respuesta de la planta al escalón unitario sin precompensador – Control LQG.

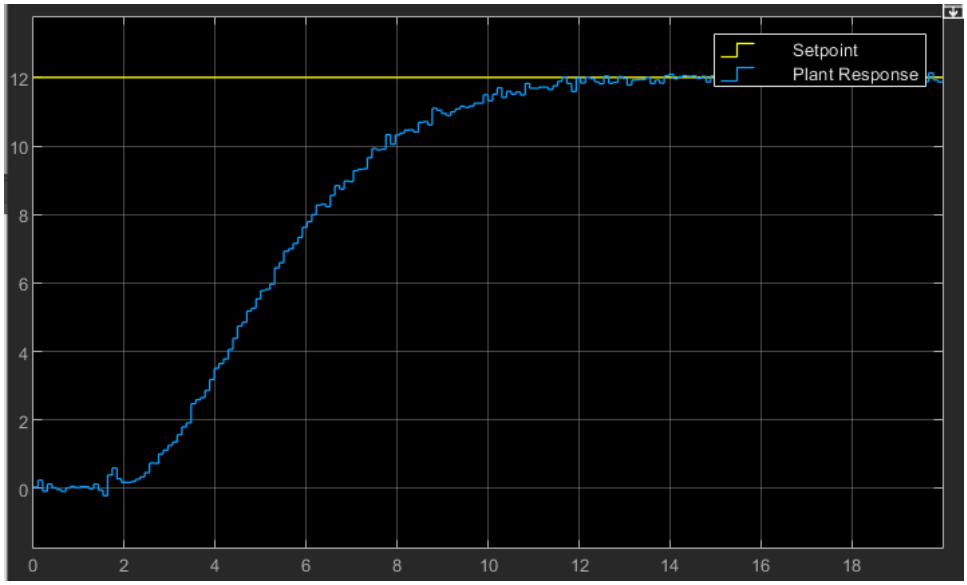
Respuesta de la planta sin precompensador al escalón unitario en Simulink de MATLAB aplicando el control LQG, Elaborado por: Ordóñez Wilson y Pino Wellington.

Motivo por el cual, se realizó el cálculo de la ganancia del precompensador mediante la ecuación 3.1, una vez obtenido el valor de estabilización mostrado en la Figura 3.29.

$$K_{pre} = \frac{1}{0.02093} = 47.7783 \qquad \text{Ec. (3.1)}$$

A partir de ello, la respuesta de la planta muestra que se alcanza el punto de consigna sin sobre impulso ni oscilaciones, para cualquier setpoint dentro del rango de trabajo de la planta como se aprecia en la Figura 3.30.

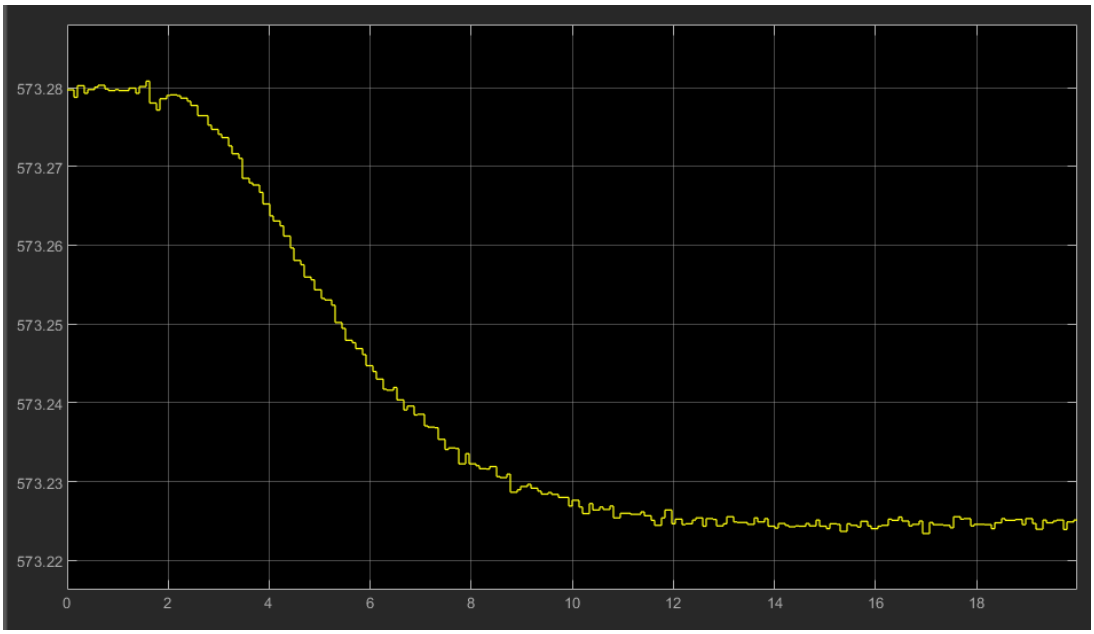
Figura 3.30. Respuesta de la planta a un setpoint de 12cm con precompensador en Simulink – Control LQG.



Respuesta de la planta con precompesador para un setpoint de 12cm en Simulink de MATLAB aplicando el control LQG, Elaborado por: Ordóñez Wilson y Pino Wellington.

Se determinó por simulación, que el control LQG no aplica para un setpoint diferente a 12cm. Esto debido a la dinámica que presenta la planta, dado que se requiere un control de rápida reacción, a factores físicos propios de la planta, que hacen que en cada prueba la planta se comporte de distinta manera, motivo por el cual, una acción Integral en el controlador es de vital importancia, ya que ayuda a minimizar el error en estado estable y a corregir rápidamente el error en estado transitorio. Además, sumado a estos inconvenientes, se observó un comportamiento erróneo en la acción del control, ya que se ve limitada por los 10 bits de resolución que se dispone en el NODE MCU, generando valores decimales, como se observa en la Figura 3.31.

Figura 3.31. Salida de controlador LQG limitada por resolución de NODE MCU en Simulink.



Salida de controlador LQG que presenta valores decimales limitando el alcance del control por la resolución de NODE MCU en Simulink de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.5 Diseño de controlador LQI

El controlador LQI al igual que el controlador LQG, es un algoritmo de control optimo, cuya principal diferencia es que no necesita un estimador ya que posee una acción integral con la cual se realimenta el estado actual de la planta, minimizando el error en estado estable.

Para el diseño del controlador LQI se realizó la simulación mostrada en la Figura 3.32, de igual forma que en el diseño del controlador LQG se creó un script, como se observa en la Figura 3.33, del cual se obtienen los valores de la matriz K mediante la función *lqi(sys,Q,R)* de MATLAB, misma que tiene como parámetros a las matrices Q y R, que para su obtención se varían sus valores evaluando la respuesta de la planta hasta obtener una respuesta deseada aproximada en la simulación.

Figura 3.32. Diagrama de Bloques de control LQI.

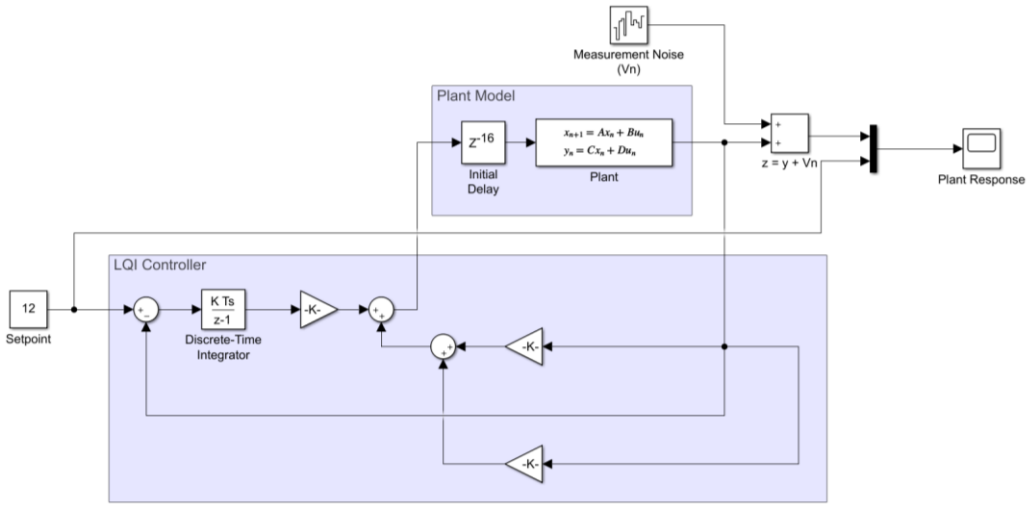


Diagrama de Bloques de control LQI en Simulink de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

Figura 3.33. Script para la obtención de las matrices Q, R y K – Control LQI.

```
clc;
% Transfer function
num = [9.489496760496952e-04,-0.001735203602620,0.005401218766916];
den = [1,0.915775407922508,0.258234581040331];
tf2 = tf(num,den);

% Transfer function to space states
ss1 = ss(tf2);
sysd = c2d(ss1,0.102);
[A,B,C,D] = ssdata(sysd);

% Q and R matrix
Q = [6 0 0;
    0 0.05 0;
    0 0 0.08];
R = 0.001;

% LQI estimation
[K,S,e] = lqi(sysd,Q,R);
```

Script desarrollado en MATLAB para la obtención de las matrices Q y R, y la matriz K del controlador LQI, Elaborado por: Ordóñez Wilson y Pino Wellington.

La matriz K del controlador LQI obtenida a través de la simulación en Simulink se presenta en la Figura 3.34.

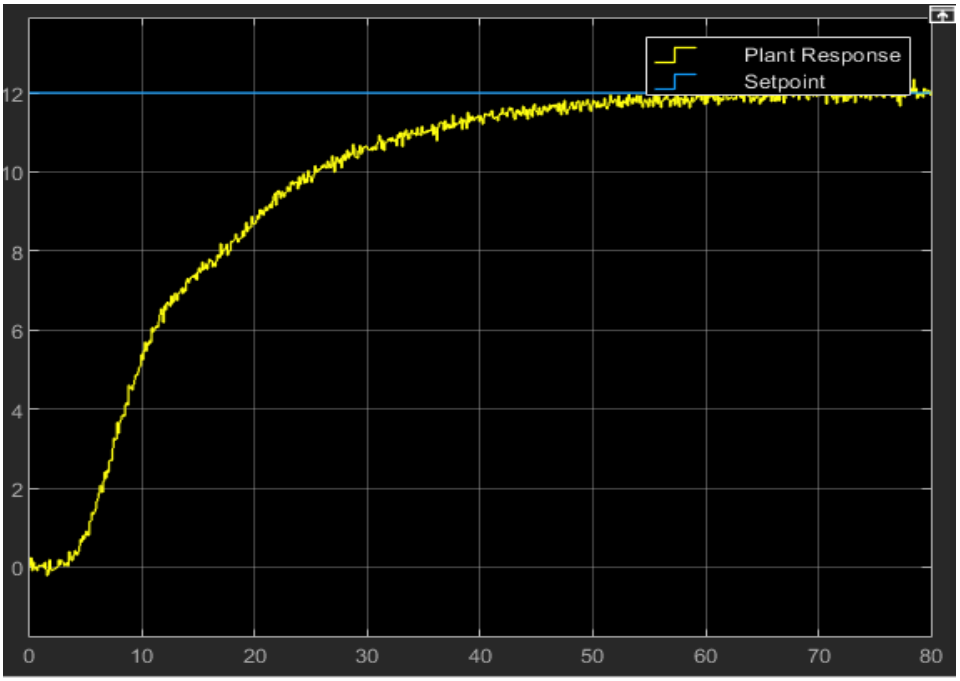
Figura 3.34. Matriz K – Control LQI.

```
>> K

K =

   43.6557    7.4674   -5.7868
```

Matriz de realimentación K del controlador LQI obtenida a través de Simulink de MATLAB, Elaborado por: Ordóñez Wilson y Pino Wellington.

La respuesta de la planta aproximada que se obtuvo en Simulink es la que muestra en la Figura 3.35.

Figura 3.35. Respuesta de la planta a un setpoint de 12cm – Control LQI.



Respuesta de la planta para un setpoint de 12cm en Simulink de MATLAB aplicando control LQI, Elaborado por: Ordóñez Wilson y Pino Wellington.

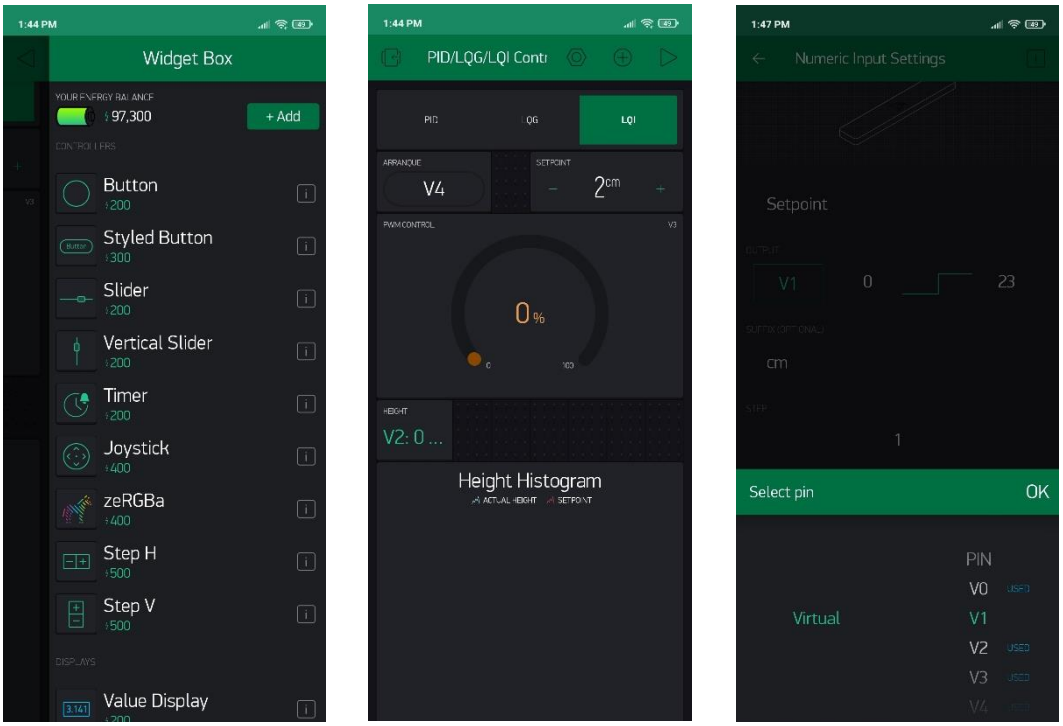
3.6 Diseño de Interfaz de Control

El control y monitoreo del sistema, se desarrolló a través de una interfaz gráfica mediante la herramienta Blynk. Se eligió esta herramienta ya que está orientada al desarrollo de proyectos y aplicaciones IoT y es soportada por la plataforma Node-RED facilitando la comunicación e interacción entre la App móvil y Node-RED. De igual manera, se presenta un menú para la configuración general de la App en el smartphone, en donde, además de varias funcionalidades, se elige un dispositivo a vincular

previamente configurado en otra pantalla de configuración como se muestra en el ANEXO XI. En este caso se seleccionó el modelo de dispositivo “Raspberry Pi 3B”, ya que es el que tiene las características que más se asemejan respecto al modelo Raspberry Pi 4, que es el implementado. También permite establecer directamente conexión al servidor en cloud de la plataforma o a un servidor local. Además, para la autenticación del smartphone que hace uso de la App, se genera un token único.

Y también muestra un panel para la integración de widgets que ofrece una configuración intuitiva, y que, además, interactúan con los dispositivos vinculados mediante pines virtuales, como se muestra en la Figura 3.36.

Figura 3.36. Interfaz y panel de widgets y configuración de pines virtuales en Blynk.



a) b) c)

a) Panel de selección de widgets, Fuente: Blynk Inc.

b) Interfaz de trabajo en base a widgets, Elaborado por: Ordóñez Wilson y Pino Wellington.

c) Ejemplo de configuración de pin Virtual, Fuente: Blynk Inc.

3.7 Desarrollo de programación en NODE MCU

Dentro de la tarjeta de desarrollo NODE MCU V1.0 (ESP8266) y mediante el entorno de desarrollo Arduino IDE, se implementó la comunicación con Node-RED mediante el protocolo MQTT, con las librerías “PubSubClient” y “ESP8266WiFi” que brindan

de una variedad de funciones para conexión wifi y con el bróker MQTT. En la Figura 3.37, se llaman a las librerías antes mencionadas, se ingresan las credenciales correspondientes para la conexión wifi, la dirección IP del bróker MQTT, y se crea un cliente wifi y un cliente MQTT.

Figura 3.37. Ingreso de credenciales, bróker y creación de clientes en NODE MCU.

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

const char* ssid = "System_Control";
const char* password = "1234678";
const char* mqtt_server = "192.168.0.52";

WiFiClient espClient;
PubSubClient client(espClient);
```

Ingreso de credenciales wifi y dirección IP de bróker MQTT, y creación de cliente wifi y MQTT para NODE MCU sobre Arduino IDE, Elaborado por: Ordóñez Wilson y Pino Wellington.

Para establecer conexión por primera vez o en caso de pérdida de conexión con el bróker MQTT se llama a la función “reconnect()”, y para subscribirse a uno o varios “topic”, se debe llamar a la función “client.subscribe(“topic_name”)” que se encuentra dentro de la función “reconnect()”, como se indica en la Figura 3.38.

Figura 3.38. Funciones para subscrición, conexión y reconexión en NODE MCU.

```
void reconnect() {
  // Loop until reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("ESP8266Client")) {
      Serial.println("connected");
      // subscribing to PWM mqtt topic
      client.subscribe("PWM");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
```

Función para subscribirse a un “topic”, dentro de la función “reconnect()”, para establecer la primera conexión y en caso de pérdida de conexión MQTT para NODE MCU sobre Arduino IDE, Elaborado por: Ordóñez Wilson y Pino Wellington.

Para la recepción de datos de uno o varios “topics” se utiliza la función “callback” presentada en la Figura 3.39, misma que se encuentra a la espera de datos provenientes de algún “topic”.

Figura 3.39. Función para recepción de datos de “topic” en NODE MCU.

```
void callback(char* topic, byte* payload, unsigned int length) {
  if (control == 1) {
    for (int i = 0; i < length; i++) {
      input += (char)payload[i];
    }
    ventilador = input.toInt();
    analogWrite(pwmPin, ventilador);     // write PWM output
    input = "";
    control = 0;
  }
}
```

Función “callback()” para recibir datos de uno o varios “topics” para NODE MCU sobre Arduino IDE, Elaborado por: Ordóñez Wilson y Pino Wellington.

Y para la publicación en un “topic” se ejecuta la siguiente función presentada en la Figura 3.40.

Figura 3.40. Función para publicar en “topic” en NODE MCU.

```
client.publish("Height", msg);
```

Función para publicar datos en un “topic” para NODE MCU sobre Arduino IDE, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.8 Integración de sistemas en Node-RED

El punto en el que se conecta todo el sistema de control para cualquier caso, PID, LQI o LQG, es en Node-RED, debido a que esta plataforma provee de una sistema robusto y dinámico de control sobre el que se pueden ejecutar los algoritmos. desarrollados, además que permite controlar la comunicación entre dispositivos y el tráfico de datos, todo ello basado en flujos, lo que brinda una visión clara de cómo interactúan los dispositivos que intervienen en el proceso de control.

La implementación de Node-RED se empleó sobre la tarjeta de desarrollo Raspberry Pi 4, ya que posee la capacidad de procesamiento y memoria suficiente para ejecutar esta plataforma, tomando en cuenta: los algoritmos de control, la comunicación entre los dispositivos, el alojamiento de servidores y el manejo de datos.

Una vez instalado el sistema operativo Raspbian Stretch en la Raspberry se la conecta a Internet para actualizar todos sus paquetes de software con los comandos "*sudo apt-get update*" y "*sudo apt-get upgrade*", y a su vez se descarga e instala el paquete de node.js y su administrador de paquetes "*npm*" desde su repositorio, ya que es el entorno de ejecución sobre el que corre Node-RED, ingresando los comandos en la Terminal de consola de Raspberry como se indica en el ANEXO XII.

A continuación, se realiza la instalación de Node-RED por medio del repositorio GitHub al ingresar los comandos en la Terminal de consola de Raspberry como se muestra en el ANEXO XII.

3.8.1 Conexión a NODE MCU por MQTT

A partir de esto, se incorpora la comunicación con el NODE MCU por medio del protocolo de comunicación MQTT, en el que, una vez realizada su programación, se instala el bróker "Mosquitto Broker" en la Raspberry ingresando los comandos en la Terminal de consola de Raspberry mostrados en el ANEXO XII.

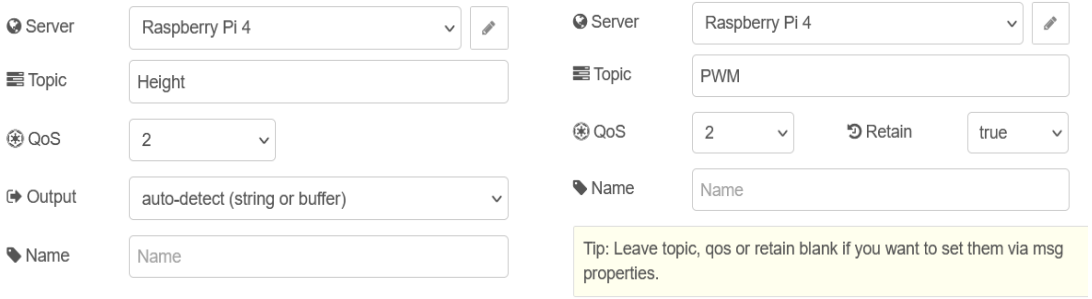
Ahora, para establecer la comunicación MQTT con el Mosquitto Broker, se accede a Node-RED, para lo cual se abre en la Raspberry un navegador cualquiera y en la barra de dirección se ingresa la dirección IP que asigna el router a la Raspberry o simplemente "localhost", ya que esta plataforma se ejecuta como local a través del puerto 1880, por defecto. Ya en la página de programación de Node-RED se crea el flujo para la comunicación MQTT mediante los nodos MQTT de recepción y envío de para los diferentes "topics", mismos que se registran en el Mosquitto Broker automáticamente. En la ventana de configuración del servidor MQTT, en el parámetro de servidor se ingresa la dirección asignada a la Raspberry y el puerto 1883 que es el puerto por defecto de MQTT, ya que se ejecuta el bróker localmente, como se indica en la Figura 3.41.

Figura 3.41. Configuración de servidor MQTT, bloques MQTT e Interfaz de Node-RED.

Configuración de servidor MQTT en bloques de conexión MQTT e ingreso a interfaz de Node-RED por navegador, Elaborado por: Ordóñez Wilson y Pino Wellington.

A continuación, en la Figura 3.42, se muestran las configuraciones de cada bloque de comunicación MQTT para recepción y envió de datos en sus diferentes “topics”.

Figura 3.42. Configuración de nodos de conexión MQTT en Node-RED.



a) b)

a) Configuración de nodo de recepción MQTT de “topic” “Height” en Node-RED.

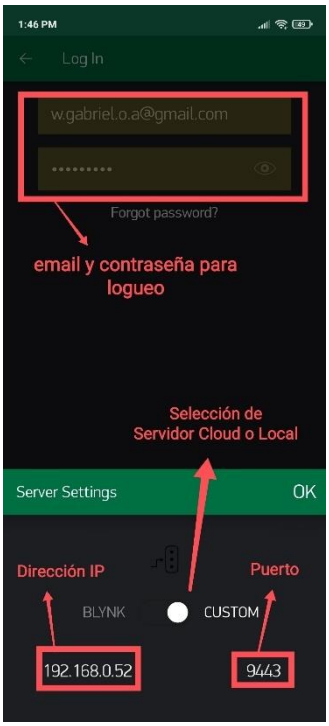
b) Configuración de nodo de envío MQTT de “topic” “PWM” en Node-RED.

3.8.2 Conexión para App de Blynk por servidor local

Para establecer la conexión con la App desarrollada en la plataforma IoT Blynk, se optó por la instalación de un servidor local en la Raspberry debido a que la red de

conexión wifi es WLAN sin conexión a internet. Para ello se ingresan los siguientes comandos en la Terminal de consola de Raspberry, como se indica en el ANEXO XIII. Una vez realizado este proceso, en la App se realiza un log-in con un email y contraseña. Posteriormente, se configura como servidor local y se ingresa la dirección IP del servidor local, que en este caso es la Raspberry y el número de puerto, el cual es 9443, que se usa por defecto para comunicación cliente-servidor en la App de Blynk como se observa en la Figura 3.43.

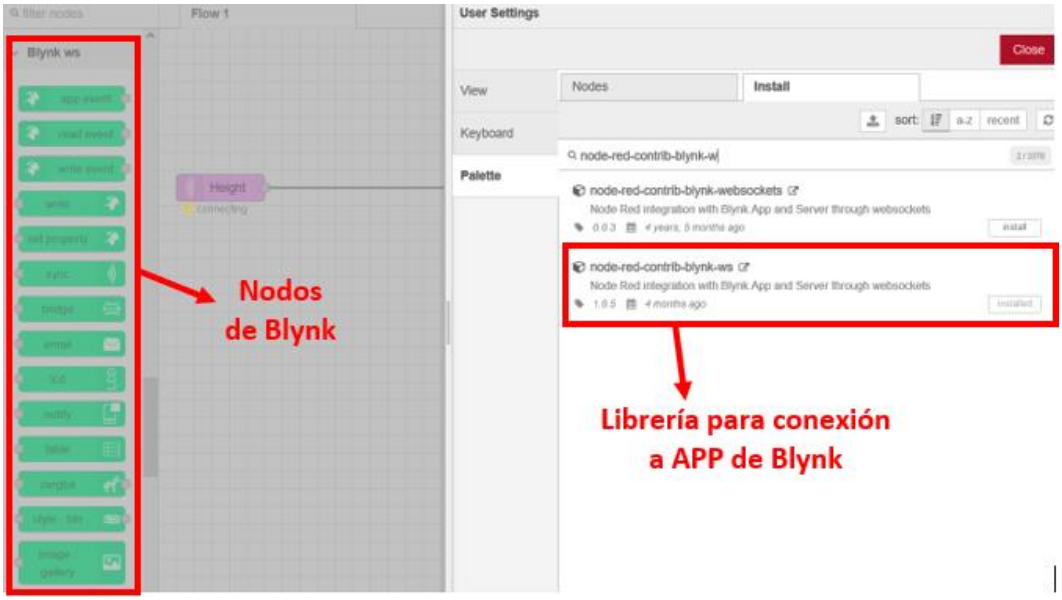
Figura 3.43. Log-in y configuración de servidor en App de Blynk.



Logueo con email y contraseña, y configuración de servidor local: dirección IP y puerto, en App de Blynk, Elaborado por: Ordóñez Wilson y Pino Wellington.

Una vez establecidos estos parámetros de servidor, creada la interfaz de control en la App y asignados los pines virtuales para las variables de cada widget, se realiza el flujo en Node-RED para establecer la conexión con la App de Blynk, motivo por el cual en el Manage Palette de Node-RED se descarga e instalación de una librería llamada "node-red-contrib-blynk-ws" como se indica en la Figura 3.44, que incorpora en Node-RED varios nodos que permiten interactuar con la App.

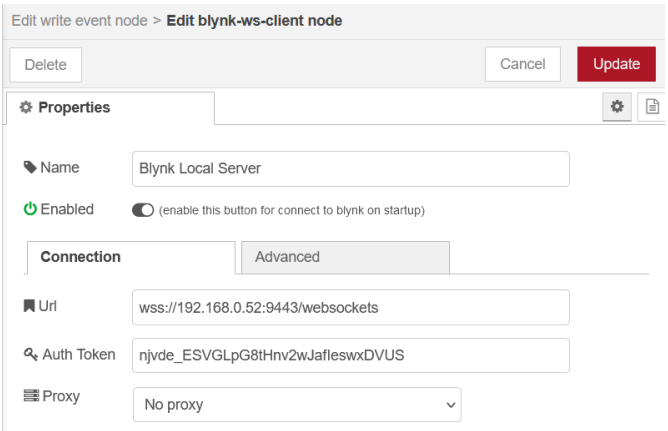
Figura 3.44. Librería y nodos de Blynk en Node-RED.



Descarga de librería y presentación de palette de nodos de Blynk en Node-RED, Elaborado por: Ordóñez Wilson y Pino Wellington.

Con la librería ya instalada, se eligen los nodos “write event” para recepción y “write” para envío de datos entre Blynk y Node-RED. En los cuales se configura el servidor con los mismos datos de la App y con el servicio websockets, que proporciona una comunicación full-duplex basada en eventos para aplicaciones cliente/servidor (MDN contributors, 2021), además de ingresar el token único de autenticación que se generó automáticamente en la App, como se muestra en la Figura 3.45.

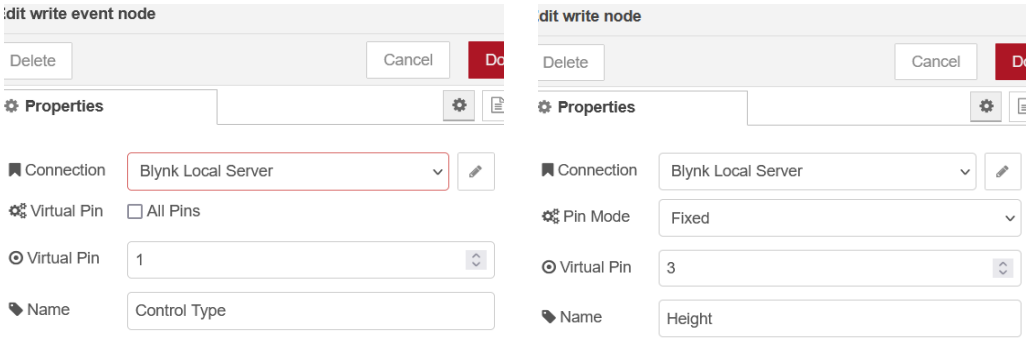
Figura 3.45. Configuración de servidor de Blynk en Node-RED.



Configuración de servidor local de Blynk en Node-RED para nodos de envío y recepción de datos, Elaborado por: Ordóñez Wilson y Pino Wellington.

Y se establece el pin virtual al que se encuentra vinculada la variable en la App y la que maneja el nodo, como se indica en la Figura 3.46.

Figura 3.46. Configuración de nodo “write event” y “write” en Node-RED.



a) b)

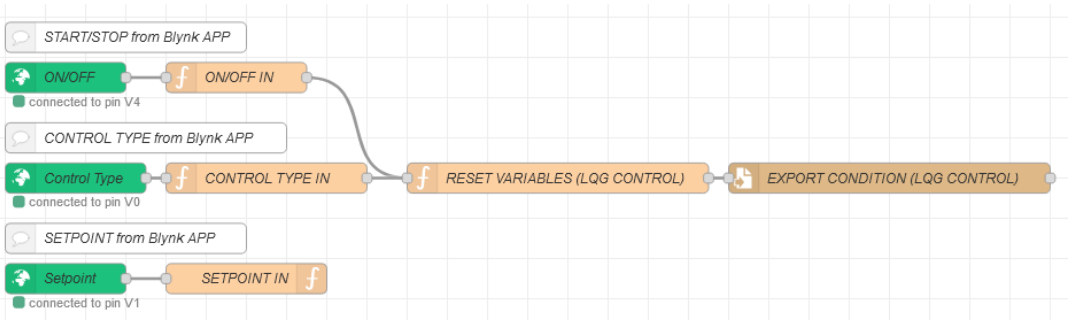
a) Se elige el servidor configurado para el nodo “write event” y se ingresa el pin virtual vinculado con variable en App y con el nodo de Blynk en Node-RED.

b) Se elige el servidor configurado para el nodo “wirte” y se ingresa el pin virtual vinculado con variable en App y con el nodo de Blynk en Node-RED, Elaborado por: Ordóñez Wilson y Pino Wellington.

3.8.3 Incorporación de flujos en Node-RED

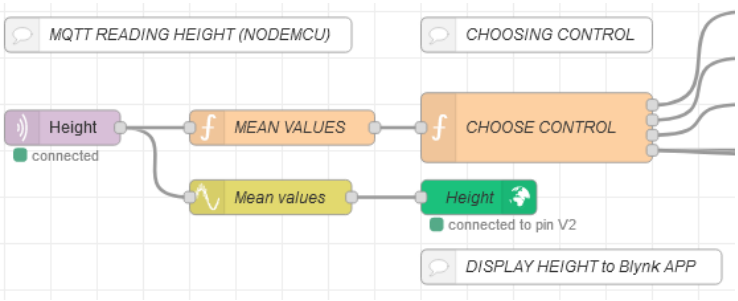
Teniendo en cuenta todas estás configuraciones, se incorporan todos los flujos en uno solo, que permita: la interacción entre todos los dispositivos que intervienen en el proceso de control; y que maneje el tráfico de datos según se lo requiera, como se muestra en la Figura 3.47, 3.48, y 3.49.

Figura 3.47. Flujos de datos desde la App de Blynk.



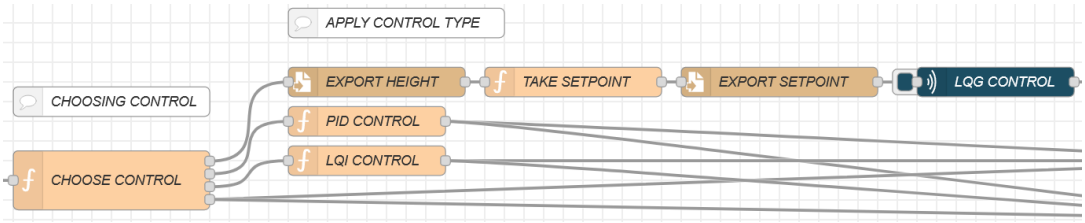
Flujos de datos de arranque, tipo de control y setpoint, provenientes de la App de Blynk, Elaborado por: Ordóñez Wilson y Pino Wellington.

Figura 3.48. Flujos de recepción por MQTT, elección de control y envío hacía la App de Blynk.



Flujos de recepción de dato de Altura por MQTT desde NODE MCU, elección de control y envío hacía la App de Blynk, Elaborado por: Ordóñez Wilson y Pino Wellington.

Figura 3.49. Aplicación de control seleccionado.

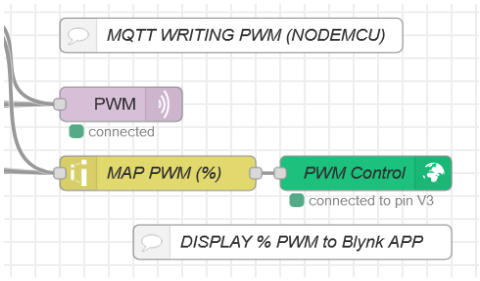


Se aplica el control seleccionado desde la App de Blynk, Elaborado por: Ordóñez Wilson y Pino Wellington.

En la Figura 3.49 se puede observar, que, para el control LQG se siguió otro procedimiento, debido al manejo complejo de matrices, ya que existe falta de recursos en programación que presentan los nodos de función de Node-RED, en los cuales se desarrolla código en lenguaje JavaScript. Por tal motivo, se optó por descargar una librería que contiene un nodo que permite ejecutar un script de Python externo, ya que en ese entorno de programación si se cuentan con los recursos necesarios para el desarrollo del algoritmo de control LQG.

Finalmente, se envía el dato PWM de salida de los controladores hacía NODE MCU y a la App de Blynk, como se indica en la Figura 3.50.

Figura 3.50. Salida de controladores por MQTT y a la App de Blynk.



Se envía el dato de PWM de la salida de los controladores hacía NODE MCU por MQTT y a la App de Blynk, Elaborado por: Ordóñez Wilson y Pino Wellington.

CAPÍTULO 4
PRUEBAS Y RESULTADOS

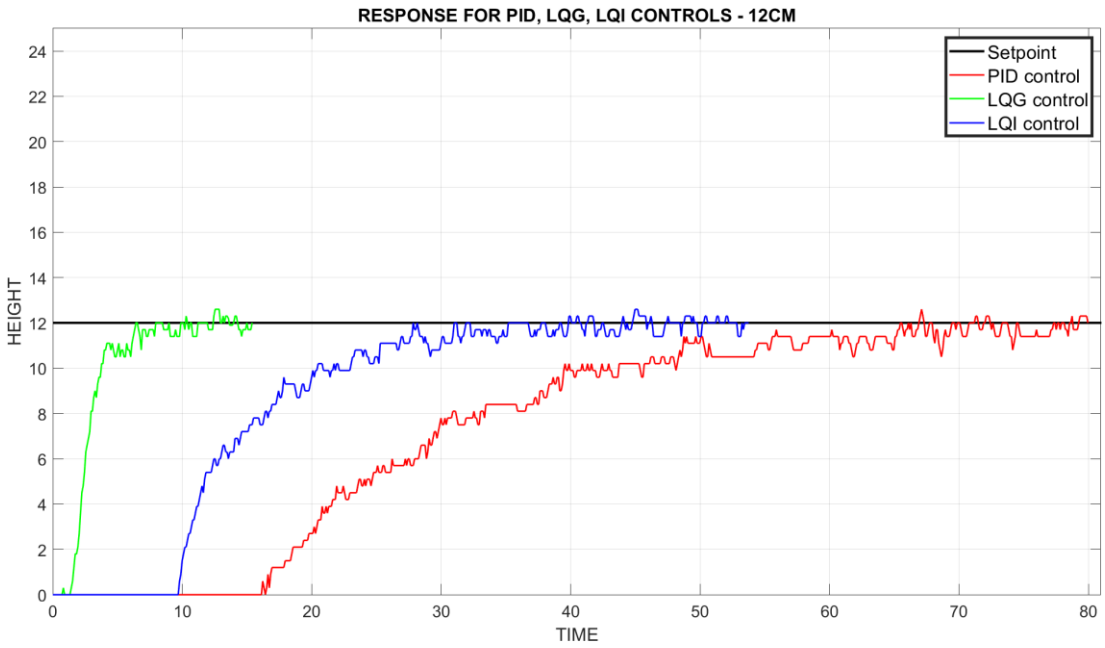
Con el fin de evaluar el funcionamiento de la planta con el sistema WNCS implementado para los 3 controladores, en donde, se determinó como punto de referencia a los 12cm en los que se estabilizó la planta a lazo abierto y debido a que es el punto medio del rango de medidas, se realizaron 6 pruebas que se detallan a continuación:

a) **1ra Prueba:** Se mide el tiempo de establecimiento desde el reposo con los 3 tipos de controles implementados, para un setpoint igual a 12cm.
b) **2ra Prueba:** Se mide el tiempo de establecimiento en marcha con 2 tipos de controles implementados, LQI y PID, para un setpoint menor a 12cm.
c) **3ra Prueba:** Se mide el tiempo de establecimiento en marcha con 2 tipos de controles implementados, LQI y PID, para un setpoint mayor a 12cm.
d) **4ta Prueba:** Se mide el tiempo de establecimiento en marcha con 2 tipos de controladores implementados, LQI y PID, en el cambio de setpoint de 23cm a 2cm, los cuales son los puntos de control críticos.
e) **5ta Prueba:** Se mide el tiempo de establecimiento en marcha con 2 tipos de controladores implementados, LQI y PID, en el cambio de setpoint de 2cm a 23cm, los cuales son los puntos de control críticos.
f) **6ta Prueba:** Se mide el tiempo de establecimiento en marcha con 2 tipos de controladores implementados, LQI y PID, sumando perturbación para setpoint de 12cm.

4.1 1ra Prueba

En esta prueba se asignó un setpoint de 12cm y se registró la gráfica de la respuesta de la planta desde el reposo hasta alcanzar la estabilidad, para todos los controladores implementados, LQG, LQI y PID, como se muestra en la Figura 4.1.

Figura 4.1. Gráfica aplicando los controles LQG, LQI y PID – Setpoint: 12cm.



Respuesta de la planta desde el reposo con los controles LQG, LQI y PID, para un setpoint de 12cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

Como resultado de obtuvieron los siguientes tiempos de establecimiento para cada caso.

Tabla 4.1. Tiempo de establecimiento de control LQG, LQI y PID – Setpoint: 12cm.

Control	**Tiempo de establecimiento [s]**
LQG	15.61
LQI	50.29
PID	78.03

Tiempo de establecimiento con los controles LQG, LQI y PID desde el reposo, para un setpoint de 12cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

De la Figura 4.1, se observa que los 3 controladores logran estabilizarse desde el reposo hasta un setpoint medio del rango de valores, en diferentes tiempos de establecimiento.

Como se indica en la Tabla 4.1, el control LQG es el controlador que presenta el menor tiempo de establecimiento, a diferencia del control PID, que presenta un tiempo 5 veces mayor al control LQG. Y el control LQI se muestra como un punto intermedio

entre estos 2 controladores, al necesitar un poco más del triple del tiempo de establecimiento del control LQG.

4.2 2da Prueba

En la Figura 4.2 se observa la respuesta del sistema en marcha aplicando los controles LQI y PID, para un setpoint de 6cm.

Figura 4.2. Gráfica aplicando los controles LQI y PID – Setpoint: 12cm a 6cm.

Respuesta de la planta con los controles LQI y PID sobre la marcha, para un setpoint de 6cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

En la Tabla 4.2 se tienen los resultados obtenidos aplicando los 2 controladores.

Tabla 4.2. Tiempo de establecimiento de control LQI y PID – Setpoint: 12cm a 6cm.

Control	Tiempo de establecimiento [s]
LQI	25.19
PID	7.45

Tiempo de establecimiento con los controles LQI y PID sobre la marcha, para un setpoint de 6cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

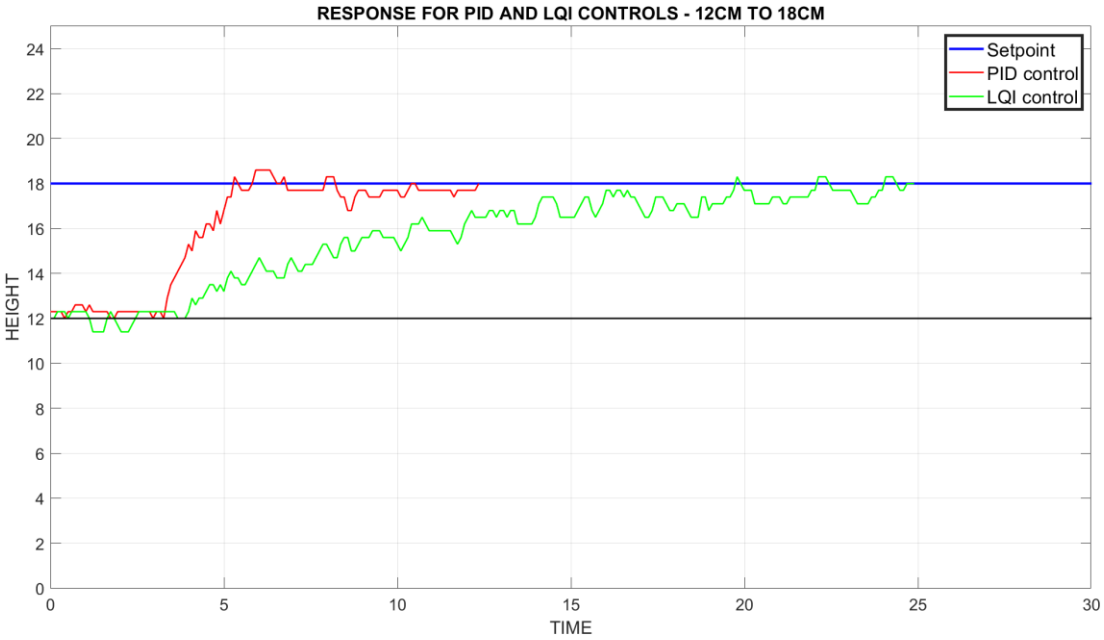
Como se puede apreciar en la Figura 4.2, los 2 controladores siguen correctamente el cambio en marcha de setpoint, en sentido descendente, sin ningún problema.

De la Tabla 4.2, se evidencia que el control LQI necesita un mayor tiempo de establecimiento que el control PID, un poco más del triple de tiempo, a diferencia de la 1ra Prueba, en la que se parte del reposo, al ser el control PID, el más lento en comparación con los otros 2 controles, en esa prueba.

4.3 3ra Prueba

En la Figura 4.3 se observa la respuesta del sistema en marcha aplicando los controles LQI y PID, para un setpoint de 18cm.

Figura 4.3. Gráfica aplicando los controles LQI y PID – Setpoint: 12cm a 18cm.



Respuesta de la planta con los controles LQI y PID sobre la marcha, para un setpoint de 18cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

En la Tabla 4.3 se tiene los resultados obtenidos aplicando los 2 controladores.

Tabla 4.3. Tiempo de establecimiento de control LQI y PID – Setpoint: 12cm a 18cm.

Control	Tiempo de establecimiento [s]
LQI	21.22
PID	9.29

Tiempo de establecimiento con los controles LQI y PID sobre la marcha, para un setpoint de 18cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

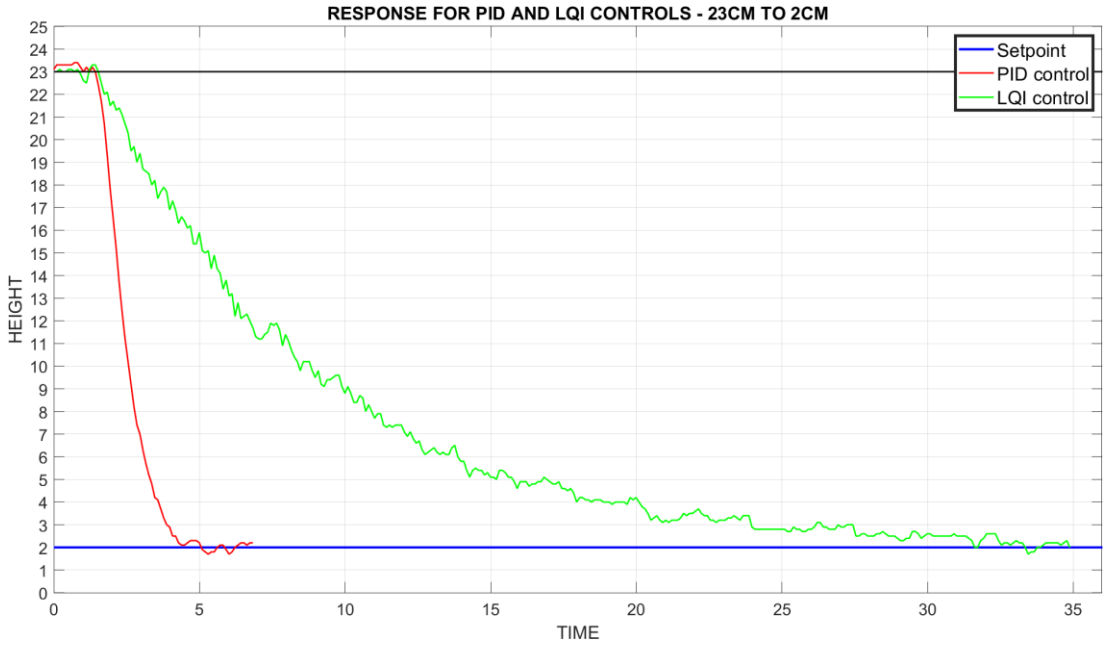
De la Figura 4.3, se observa que estos 2 controladores no pierden en el control frente a un cambio en marcha de setpoint en sentido ascendente.

Como se indica en la Tabla 4.3, el control PID sigue siendo el que necesita menor tiempo de establecimiento en comparación con el control LQI, el doble de tiempo menos en esta prueba.

4.4 4ta Prueba

En la Figura 4.4 se observa la respuesta del sistema en puntos críticos de control, con el cambio de un setpoint de 23cm a 2cm, aplicando los controles LQI y PID.

Figura 4.4. Gráfica aplicando los controles LQI y PID – Setpoint: 23cm a 2cm.



Respuesta de la planta con los controles LQI y PID desde un setpoint de 23cm a 2cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

En la Tabla 4.4 se tiene los resultados obtenidos aplicando los 2 controladores.

Tabla 4.4. Tiempo de establecimiento de control LQI y PID – Setpoint: 23cm a 2cm.

Control	Tiempo de establecimiento [s]
LQI	33.15
PID	5.3

Tiempo de establecimiento con los controles LQI y PID desde un setpoint de 23cm a 2cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

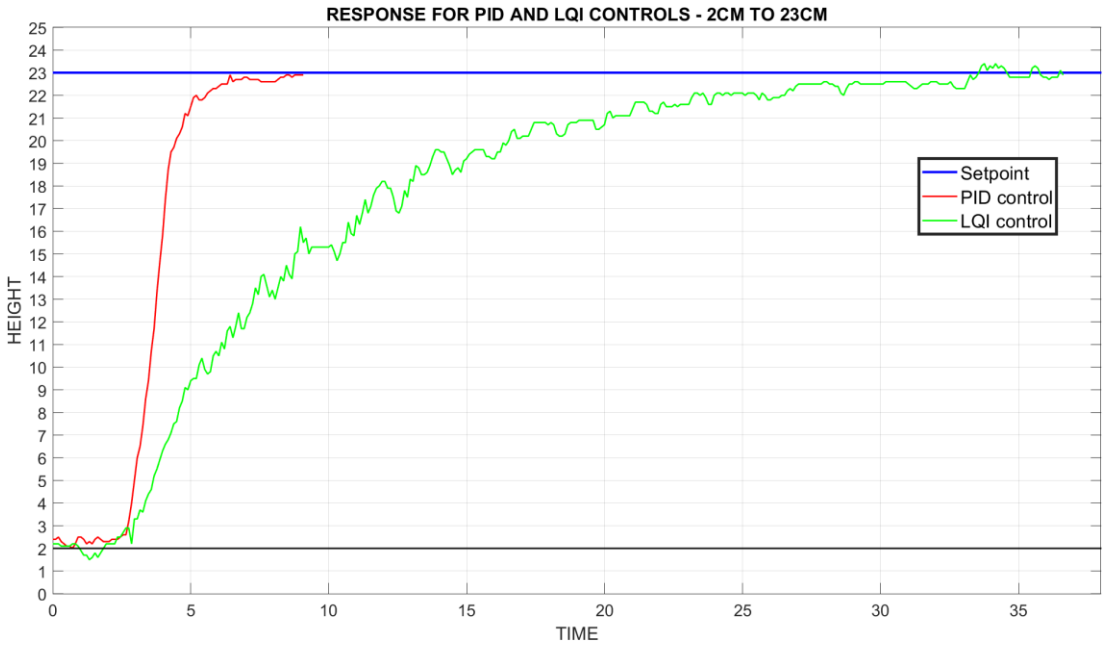
De la Figura 4.4 se evidencia que los controladores no se pierden frente a un drástico cambio de setpoint en marcha en los puntos críticos de control de manera descendente.

Y como se indica en la Tabla 4.4, se puede apreciar que el control PID continúa siendo el control con mejor tiempo de respuesta a los cambios de setpoint en comparación con el control LQI, al necesitar 6 veces menos tiempo de establecimiento en esta prueba.

4.5 5ta Prueba

En la Figura 4.5 se observa la respuesta del sistema en puntos críticos de control, con el cambio de un setpoint de 2cm a 23cm, aplicando los controles LQI y PID.

Figura 4.5. Gráfica aplicando los controles LQI y PID – Setpoint: 2cm a 23cm.



Respuesta de la planta con los controles LQI y PID desde un setpoint de 2cm a 23cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

En la Tabla 4.5 se tiene los resultados obtenidos aplicando los 2 controladores.

Tabla 4.5. Tiempo de establecimiento de control LQI y PID – Setpoint: 2cm a 23cm.

Control	Tiempo de establecimiento [s]
LQI	33.15
PID	6.94

Tiempo de establecimiento con los controles LQI y PID desde un setpoint de 2cm a 23cm, Elaborado por: Ordóñez Wilson y Pino Wellington.

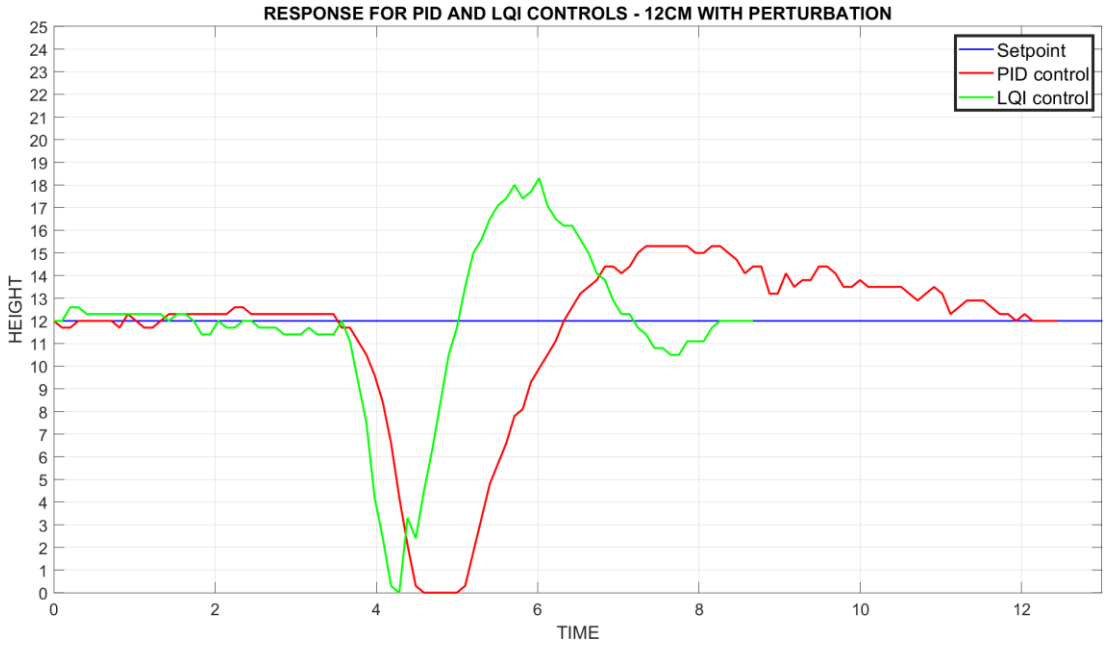
De la Figura 4.5, se puede observar que los controladores siguen de manera correcta el drástico cambio en marcha de setpoint en ascenso.

Como se indica en la Tabla 4.5, el control PID permanece como el control más rápido en comparación con el control LQI, dado que presenta 5 veces menos tiempo de establecimiento.

4.6 6ta Prueba

En la Figura 4.6 se observa la respuesta del sistema con perturbación, para un setpoint de 12cm, aplicando los controles LQI y PID.

Figura 4.6. Gráfica aplicando los controles LQI y PID – Setpoint: 12cm con perturbación.



Respuesta de la planta con los controles LQI y PID para un setpoint de 12cm con perturbación, Elaborado por: Ordóñez Wilson y Pino Wellington.

En la Tabla 4.6 se tiene los resultados obtenidos aplicando los 2 controladores.

Tabla 4.6. Tiempo de establecimiento de control LQI y PID – Setpoint: 12cm con perturbación.

Control	**Tiempo de establecimiento [s]**
LQI	5
PID	9

Tiempo de establecimiento con los controles LQI y PID para un setpoint de 12cm con perturbación, Elaborado por: Ordóñez Wilson y Pino Wellington.

De la Figura 4.6, se puede observar que los controladores siguen de manera correcta el setpoint con la presencia de perturbación.

Como se indica en la Tabla 4.6, el control LQI requiere menos tiempo de establecimiento frente a una perturbación en comparación con el control PID, que necesita 4 segundos más, en contraste con la 2da, 3ra, 4ta y 5ta Prueba.

CAPITULO 5
CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

Luego del análisis y estudio de la bibliografía se determinó que el método de la caja negra no es el apropiado para desarrollar el modelado matemático de esta planta, dado que no toma en cuenta factores internos que intervienen en el funcionamiento de la misma, lo cual es de gran importancia para la implementación del algoritmo de control LQG debido al estimador que contiene, el cual requiere emular en su mayor aproximación el comportamiento del sistema, a diferencia del control PID y LQI, puesto que estos algoritmos cuentan con una acción integral que permite un seguimiento robusto del setpoint, sorteando problemas relacionados con la física interna de la planta.

Se diseñó e implementó una planta personalizada utilizando tecnologías de fabricación por adición con herramientas CAD-CAM, acorde a las características físicas de la misma y con la optimización de uso de espacio, minimizando errores en su estructura y ruidos a nivel electrónico y mecánico, lo que a su vez facilita la obtención de una función de transferencia con un alto porcentaje de aproximación al comportamiento real que describe la planta.

Para la implementación del sistema desarrollado, se empleó un sistema embebido como la tarjeta Raspberry Pi 4, que, por sus características, presta ventajas para la comunicación y desarrollo de plataformas IoT mediante Node-RED, y gracias a la capacidad de procesamiento y memoria permite ejecutar y alojar los algoritmos de control y a la vez cumplir con la función de servidor para la comunicación MQTT y de la App de Blynk.

Al desarrollar una App mediante el uso de herramientas IoT como Node-RED y Blynk, se provee una interfaz amigable para el usuario, facilitando el monitoreo y control de la planta, y con un alcance dado por la red WLAN, lo que a su vez genera mayor seguridad, y más aún, con las herramientas que brindan estas plataformas, como el uso

de un tocken de autenticación para el acceso al sistema y el hecho de que se permite la implementación de un servidor local.

Mediante las pruebas de funcionamiento de cada uno de los algoritmos se determinó:

i. Que el controlador óptimo LQG fue el que presentó el menor tiempo de establecimiento, al partir del reposo, debido a que la acción de control va desde su punto más alto hasta el más bajo, lo que corrige con mayor rapidez el error en estado transitorio recortando de gran manera el tiempo de establecimiento, a diferencia de los demás algoritmos de control cuya acción de control es de forma ascendente.

ii. El controlador óptimo LQI presenta una respuesta lenta en términos de tiempo de establecimiento, en comparación con un control no óptimo, como el PID, ya que la demora añade perturbación en la salida de la planta aumentando el tiempo que le toma estabilizarla, pero cumple con la consigna de manera satisfactoria.

iii. Cuando los controladores parten del reposo el control PID, requiere más tiempo que el control LQI, debido a que cuando se parte del reposo los errores de realimentación del controlador PID tienen que acumularse para romper con la inercia del sistema hasta llegar al setpoint, en donde alcanza su máximo rendimiento, siendo capaz de seguir rápidamente a cualquier setpoint, en contraste con las ganancias constantes de realimentación del controlador LQI, que le permiten romper con la inercia del sistema de forma rápida desde el primer instante.

iv. Con la presencia de perturbación el control LQI presenta la respuesta más rápida, pero con mayor sobreimpulso, frente al control PID, quien presenta un retardo de 4 segundos suavizando su respuesta, lo que convierte al control LQI en un controlador robusto frente a perturbación emergentes en el sistema.

5.2 RECOMENDACIONES

Se propone implementar un sensor de alta gama que brinde mayor precisión, repetitividad y confiabilidad en las medidas que entregue, sorteando de la mejor manera las perturbaciones tanto físicas, que presenta la planta, como electrónicas, limitadas por hardware.

Para la obtención de la función de transferencia se debería realizar un análisis a detalle de los todos los factores físicos que intervienen en el proceso que describe su funcionamiento para obtener un modelo matemático que se aproxime al máximo el comportamiento de la planta.

Con la finalidad de solventar los percances dados por la dinámica de la planta en el controlador LQG, se propone añadir una acción Integral que aporte a la corrección del error y permita al controlador seguir la consigna en diferentes setpoints, lo que convierte al controlador LQG en un controlador LQGI, del cual actualmente se tiene poca información, pero puede ser sometido a estudios con mayor profundidad en el futuro próximo.

Se propone migrar la red WLAN a una WAN cloud y segura, que permita un mayor rango de control y provee un sin número de herramientas extra, dado que las plataformas IoT implementadas facilitan este tipo infraestructura.

BIBLIOGRAFÍA

Raspberry Pi Trading Ltd. (Enero de 2021). *Raspberry Pi 4 Tech Specs.* Obtenido de Raspberry Pi: https://datasheets.raspberrypi.org/rpi4/raspberry-pi-4-product-brief.pdf

Al-Dabbagh, A. W., & Chen, T. (Septiembre de 2016). Design Considerations for Wireless Networked Control Systems. *IEEE Transactions on Industrial Electronics, 63*(9), 5547-5557.

Améstegui Moreno, M. (Enero de 2001). APUNTES DE CONTROL PID. La Paz, Bolivia. Obtenido de https://d1wqtxts1xzle7.cloudfront.net/39445583/Control_Pid.pdf?1445913670=&response-content-disposition=inline%3B+filename%3DControl_Pid.pdf&Expires=1623527544&Signature=CZFzCV3VBcd6CrbYC8QDqWaqUOUptriRr2Rvk2tU8bPhg2bAoCOyRtbsgit7MQSZjTNDEf4mwWx~RZ9dJpSpw

Banerjee, R., & Pal, A. (2018). Stabilization of Inverted Pendulum on Cart Based on LQG Optimal Control. *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*, (págs. 1-4). Kottayam, India.

Barbera Moral, E. (2013). Comparación entre diferentes procedimientos de ajuste de controladores PID. I. Valores máximos de la variable controlada y de la señal reguladora. *Afinidad, 70*(564), 242-245.

Beauchamp Báez, G., & Batista, R. (2016). Aplicación de Técnicas de Control Óptimo a una plataforma estacionaria cuatrimotor. *RIELAC, XXXVII*, 34 - 49. Obtenido de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59282016000300004

Blynk. (10 de Junio de 2021). *Introduction*. Obtenido de Blynk Documentation: https://docs.blynk.io/en/

Brunton, S. L., & Kutz, J. N. (Mayo de 2019). Linear Control Theory. *Data Driven Science & Engineering Machine Learning, Dynamical Systems, and Control*. Cambridge University Press.

Cargua Abril, W. A., & Gallegos Herrera, M. X. (Diciembre de 2017). ESTUDIO, ANÁLISIS Y SIMULACIÓN DE ESQUEMAS DE CONTROL TIPO PID, SMC, Y LQR PARA REACTORES QUÍMICOS TIPO CSTR. Quito, Ecuador: Escuela Politécnica Nacional Facultad de Ingeniería Eléctrica y

Electrónica . Obtenido de https://bibdigital.epn.edu.ec/bitstream/15000/19069/1/CD-8470.pdf

Chalán Padilla, V. A. (Octubre de 2020). DESARROLLO DE UN CONTROLADOR ÓPTIMO LQR UTILIZANDO HERRAMIENTAS IOT PARA UN SISTEMA DE PRESIÓN CONSTANTE CONTROLADO REMOTAMENTE. Quito, Ecuador. Obtenido de https://dspace.ups.edu.ec/handle/123456789/19395

Colín Rivas, L., García Mejía, J. F., & Flores Fuentes, A. A. (2020). DISEÑO E IMPLEMENTACIÓN DE UN REGULADOR PID DEL FLUJO DE UN LEVITADOR NEUMÁTICO. *Tecnología, Diseño E Innovación, 6*(1), 39-50. Obtenido de https://www.unae.edu.py/ojs/index.php/facat/article/view/260

Components 101. (22 de Abril de 2020). *Components101*. Obtenido de Components101: https://components101.com/development-boards/nodemcu-esp8266-pinout-features-and-datasheet

Cova, W. J. (Diciembre de 2005). CONTROL PID UN ENFOQUE DESCRIPTIVO. La Rioja, La Rioja, Argentina: Universidad Tecnológica Nacional . Obtenido de http://www.frlr.utn.edu.ar/archivos/alumnos/electronica/catedras/38-sistemas-de-control-aplicado/Publicaciones/Control_PID_Enfoque_Descriptivo.pdf

Cova, W. J. (2008). SISTEMAS DE CONTROL APLICADO APUNTES DE CATEDRA. La Rioja, La Rioja, Argentina: Universidad Tecnológica Nacional. Obtenido de http://www.frlr.utn.edu.ar/index.php/menu-dep-electronica-cat-es/55-art-alumnos-electronica/132-catedras-electronica-sist-control-aplicado

Dorado, J., & Ruíz, J. F. (s.f.). IMPLEMENTACIÓN DE FILTROS DE KALMAN COMO MÉTODO DE AJUSTE A LOS MODELOS DE PRONÓSTICO (GFS) DE TEMPERATURAS MÁXIMAS Y MÍNIMA PARA ALGUNAS CIUDADES DE COLOMBIA. Colombia. Obtenido de http://www.ideam.gov.co/documents/21021/21132/Filtro_de_Kalman.pdf/e98277fe-a85c-49b6-b11b-e415f262c066

Eclipse Foundation. (5 de Abril de 2019). *Eclipse Mosquitto*. Obtenido de Eclipse Mosquitto: https://mosquitto.org/

ELECFreaks. (2013). *Sparkfun.* Obtenido de Sparkfun: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

Espressif Inc. (2020). *ESPRESSIF*. Obtenido de ESPRESSIF: https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf

González Uribe, C. D. (2016). COMPARACIÓN DE UN CONTROLADOR LQR VS UN CONTROLADOR PID IMPLEMENTADOS EN UN HELICÓPTERO DE DOS GRADOS DE LIBERTAD PIVOTADO. Bogotá D.CC, Colombia: UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS INGENIERÍA EN CONTROL. Obtenido de https://repository.udistrital.edu.co/bitstream/handle/11349/4020/GonzalezUribeChristianDavid2016.pdf?sequence=1&isAllowed=y

Hita Albarracín, Á. (Noviembre de 2020). *MQTT vs HTTP: ¿qué protocolo es mejor para IoT?* Obtenido de Borrowbits: https://borrowbits.com/2020/04/mqtt-vs-http-que-protocolo-es-mejor-para-iot/

Kisszölgyémi, I., Beneda, K., & Faltin, Z. (2017). Linear Quadratic Integral (LQI) Control for a Small Scale Turbojet Engine with Variable Exhaust Nozzle István Kisszölgyémi*, Károly Beneda† a. Brno, República Checa. Obtenido de http://conference.unob.cz/icmt/2017/iso/papers/706.pdf

Madakam, S. L. (2015). Internet of Things (IoT): A literature review. *Journal of Computer and Communications, 3*(05), 164.

MDN contributors. (30 de Julio de 2021). *WebSockets*. Obtenido de MDN Web Docs: https://developer.mozilla.org/es/docs/Web/API/WebSockets_API

Mouser Electronics. (29 de 11 de 2019). *SparkFun*. Obtenido de SparkFun: https://www.mouser.es/images/marketingid/2019/img/177109536.png?v=052020.0949

MQTT. (2020). *FAQ*. Obtenido de MQTT: https://mqtt.org/faq/

Munuera Raga, M. C. (27 de Junio de 2018). Filtro de Kalman y sus aplicaciones. Barcelona, España. Obtenido de http://diposit.ub.edu/dspace/bitstream/2445/127417/2/memoria.pdf

Novales, A. (Diciembre de 2017). Filtro de Kalman: teoria y aplicaciones. Madrid, España: Departamento de EconomÌa Cuantitativa Universidad Complutense. Obtenido de https://www.ucm.es/data/cont/media/www/pag-41459/Filtro%20de%20Kalman.pdf

Ñeco Garcia, R. P., Reinoso García, Ó., García Aracil, N., & Aracil Santonja, R. (2003). *APUNTES DE SISTEMAS DE CONTROL.* Alicante, España: Editorial Club Universitario.

Ogata, K. (2010). *Ingeniería de control moderna.* Madrid: PEARSON EDUCACIÓN, S.A.

OpenJS Foundation. (2021). *About*. Obtenido de Node-RED: https://nodered.org/about/

Ornelas Tellez, F. (2016). Introducción al Control Óptimo. Morelia, Michoacan, México. Obtenido de http://dep.fie.umich.mx/~fornelas/data/uploads/intro_optimo.pdf

Park, P., Ergen, S. C., Fischione, C., & Lu, C. (2018). Wireless Network Design for Control Systems: A Survey. *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, 978-1013.

PCJAM20. (2021). *MercadoLibre*. Obtenido de MercadoLibre: https://articulo.mercadolibre.cl/MLC-434965998-ventilador-8x8x25-cm-de-12v-170ma-2600rpm-vn-351-_JM

Perez, M. A., Perez Hidalgo, A., & Perez Berenguer, E. (2007). INTRODUCCION A LOS SISTEMAS DE CONTROL Y MODELO MATEMÁTICO PARA SISTEMAS LINEALES INVARIANTES EN EL TIEMPO. San Juan, Argentina. Obtenido de http://dea.unsj.edu.ar/control1/apuntes/unidad1y2.pdf

Pillajo, C., & Hincapié, R. (2018). *Wireless Network Control Systems.* Cuenca: Editorial Universitaria Abya-Yala.

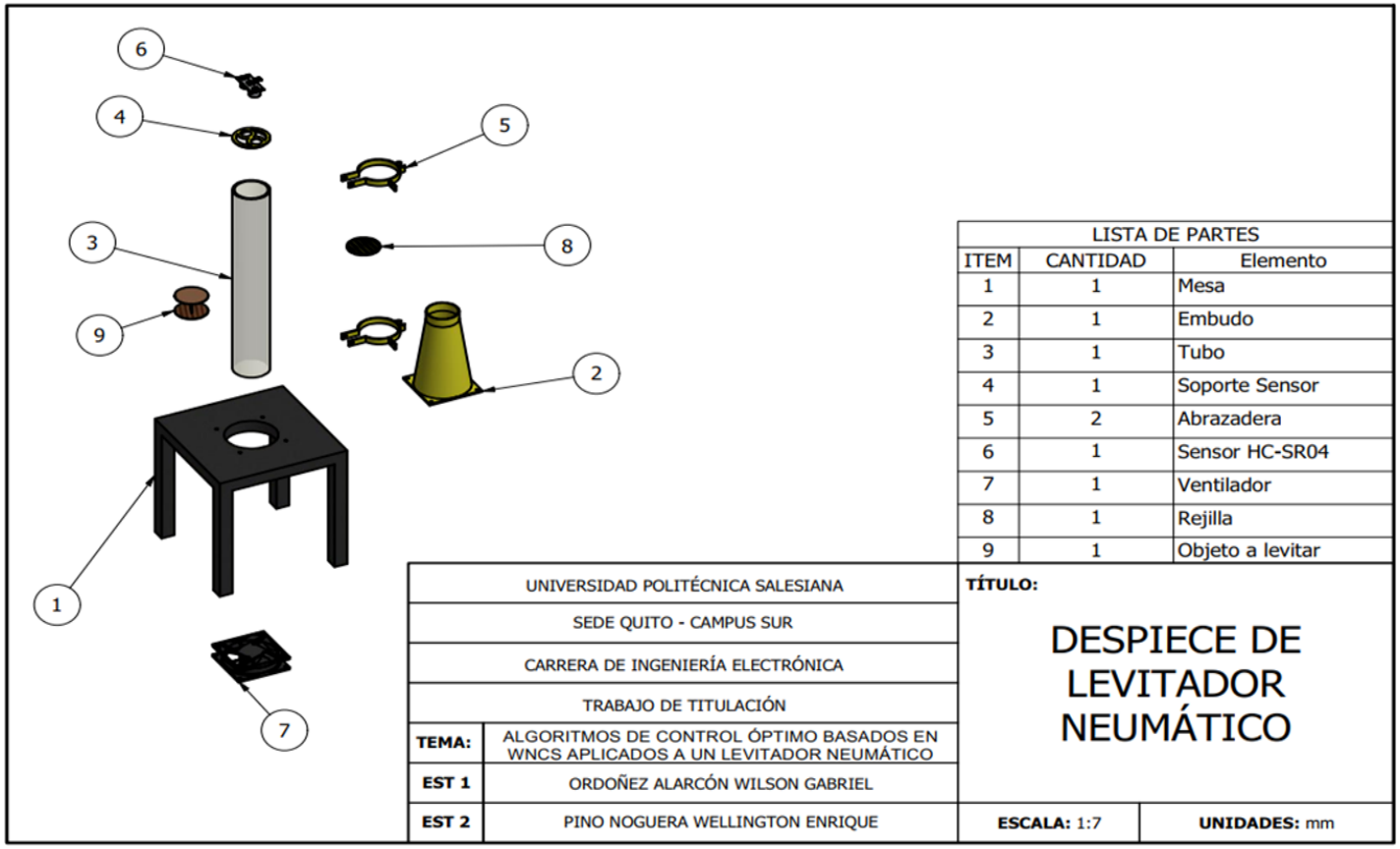
Solórzano Peñafiel, D. K. (Julio de 2018). DISEÑO E IMPLEMENTACIÓN DE UN CONTROLADOR ÓPTIMO LQG. Quito, Pichincha, Ecuador. Obtenido de https://dspace.ups.edu.ec/bitstream/123456789/15814/1/UPS-ST003638.pdf

Sreenivasulu, R., & Chalamalasetti, S. R. (2019). Applicability of Industrial Internet of Things (IIoT) in Lean Manufacturing: A brief Study.

Toro, A., Sánchez, G., Strefezza, M., & Granado, E. (2017). IIoT y sistemas de control: oportunidades, desafíos y arquitecturas. *Ciencia e Ingeniería, 38*(3).

ANEXO I

Despiece de la planta



LISTA DE PARTES		
ITEM	CANTIDAD	Elemento
1	1	Mesa
2	1	Embudo
3	1	Tubo
4	1	Soporte Sensor
5	2	Abrazadera
6	1	Sensor HC-SR04
7	1	Ventilador
8	1	Rejilla
9	1	Objeto a levitar

UNIVERSIDAD POLITÉCNICA SALESIANA	
SEDE QUITO - CAMPUS SUR	
CARRERA DE INGENIERÍA ELECTRÓNICA	
TRABAJO DE TITULACIÓN	
TEMA:	ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO
EST 1	ORDOÑEZ ALARCÓN WILSON GABRIEL
EST 2	PINO NOGUERA WELLINGTON ENRIQUE

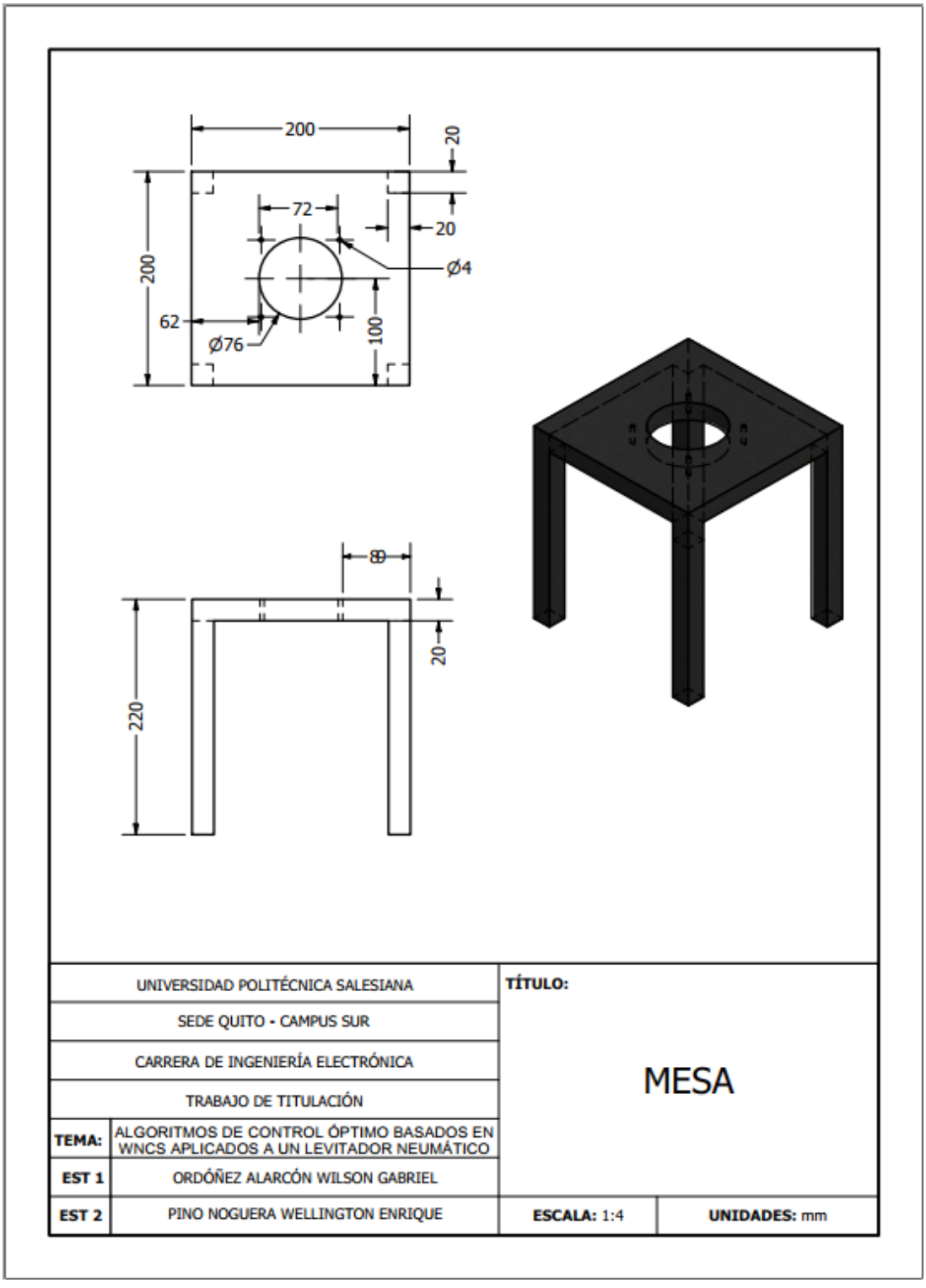
TÍTULO:

DESPIECE DE LEVITADOR NEUMÁTICO

ESCALA: 1:7 | **UNIDADES:** mm

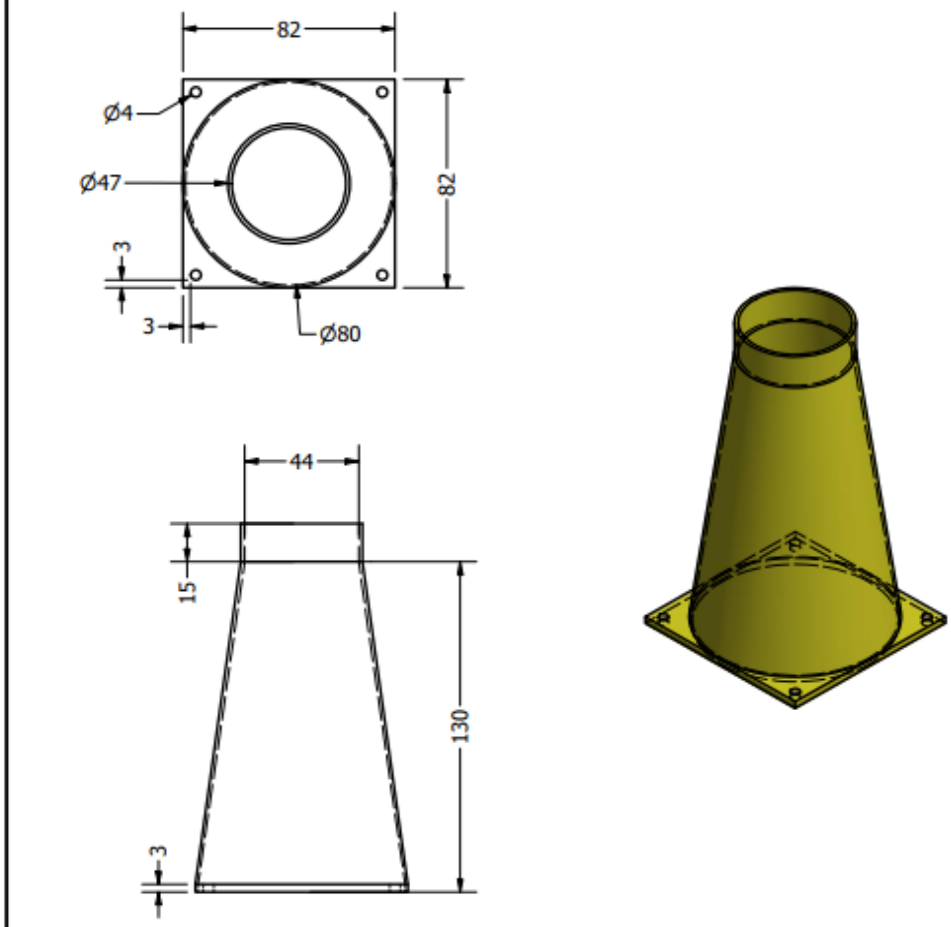
ANEXO II

PLANO DE MESA DISEÑADA



ANEXO III

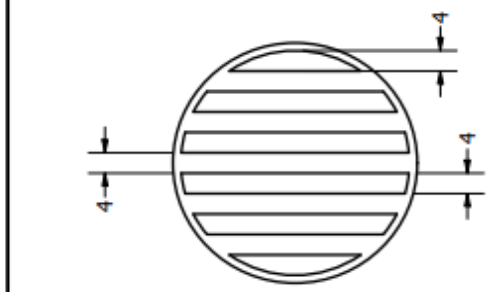
PLANO DE EMBUDO DISEÑADO

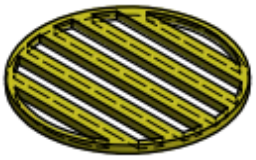


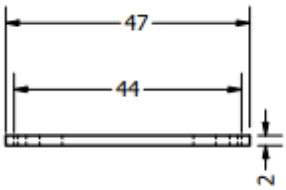
UNIVERSIDAD POLITÉCNICA SALESIANA		**TÍTULO:**	
SEDE QUITO - CAMPUS SUR		EMBUDO	
CARRERA DE INGENIERÍA ELECTRÓNICA			
TRABAJO DE TITULACIÓN			
TEMA:	ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO		
EST 1	ORDÓÑEZ ALARCÓN WILSON GABRIEL		
EST 2	PINO NOGUERA WELLINGTON ENRIQUE	**ESCALA:** 1:2	**UNIDADES:** mm

ANEXO IV

PLANO DE REJILLA DISEÑADA



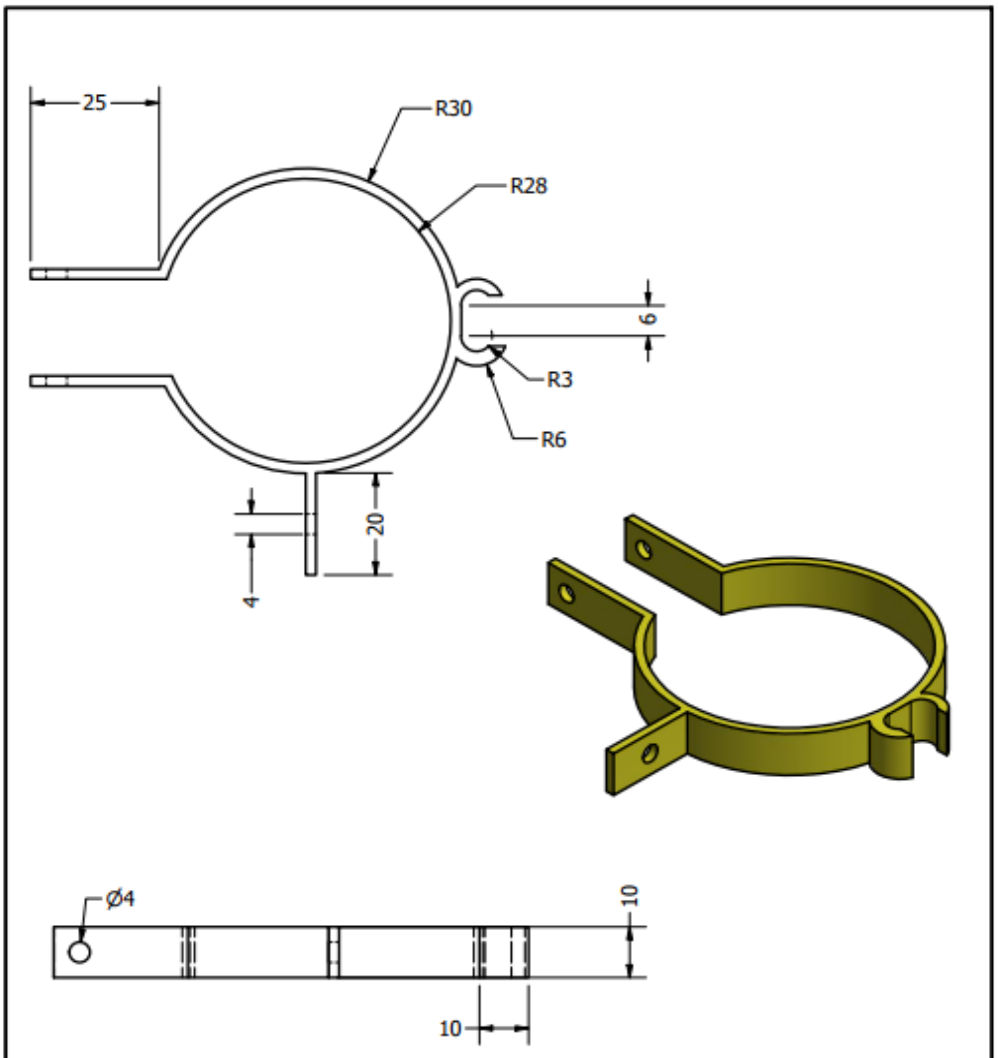




UNIVERSIDAD POLITÉCNICA SALESIANA		**TÍTULO:**	
SEDE QUITO - CAMPUS SUR		REJILLA	
CARRERA DE INGENIERÍA ELECTRÓNICA			
TRABAJO DE TITULACIÓN			
TEMA:	ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO		
EST 1	ORDÓÑEZ ALARCÓN WILSON GABRIEL		
EST 2	PINO NOGUERA WELLINGTON ENRIQUE	**ESCALA:** 1:1	**UNIDADES:** mm

ANEXO V

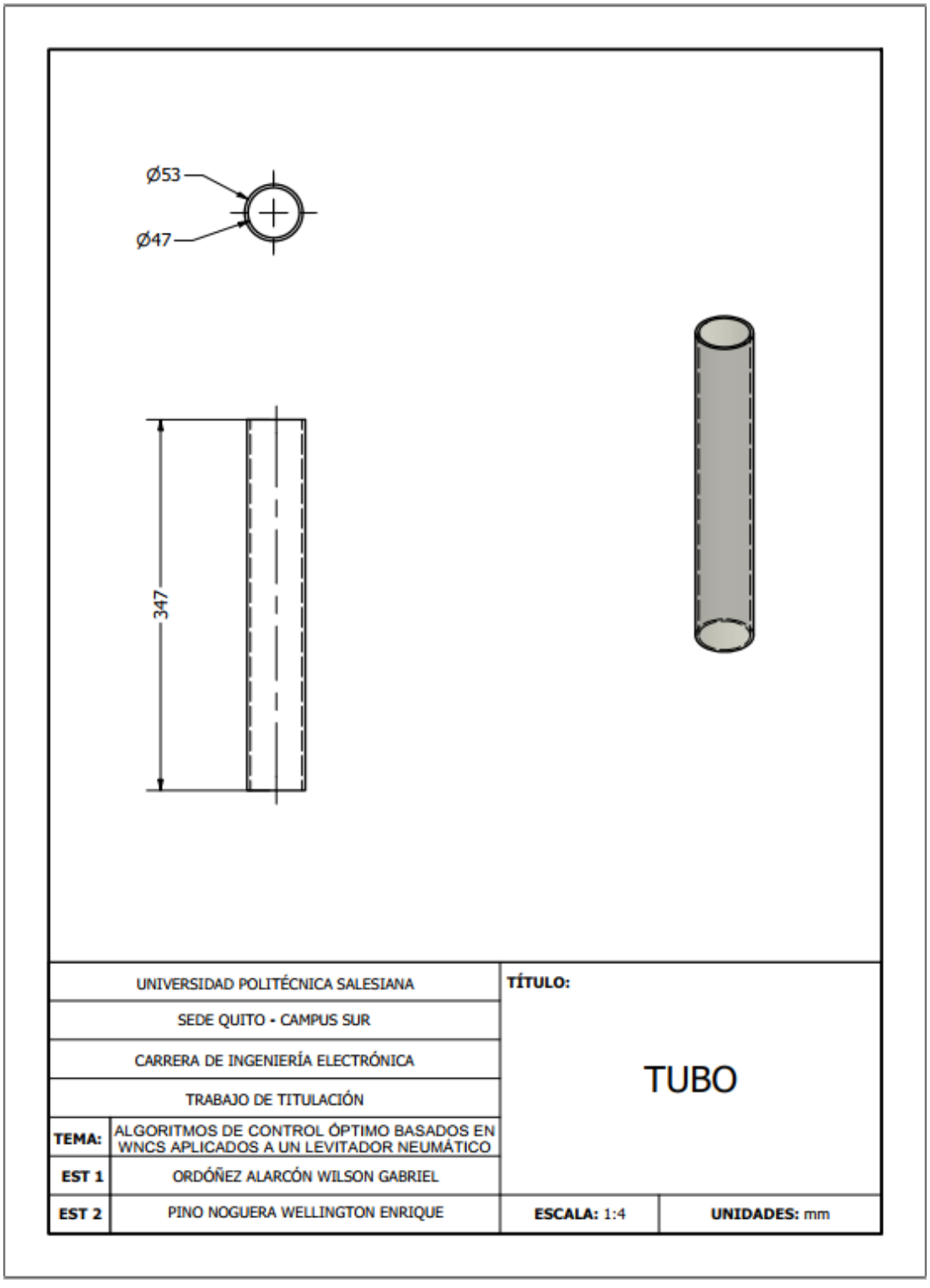
PLANO DE ABRAZADERA DISEÑADA



UNIVERSIDAD POLITÉCNICA SALESIANA		**TÍTULO:** ABRAZADERA	
SEDE QUITO - CAMPUS SUR			
CARRERA DE INGENIERÍA ELECTRÓNICA			
TRABAJO DE TITULACIÓN			
TEMA:	ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO		
EST 1	ORDÓÑEZ ALARCÓN WILSON GABRIEL		
EST 2	PINO NOGUERA WELLINGTON ENRIQUE	**ESCALA:** 1:2	**UNIDADES:** mm

ANEXO VI

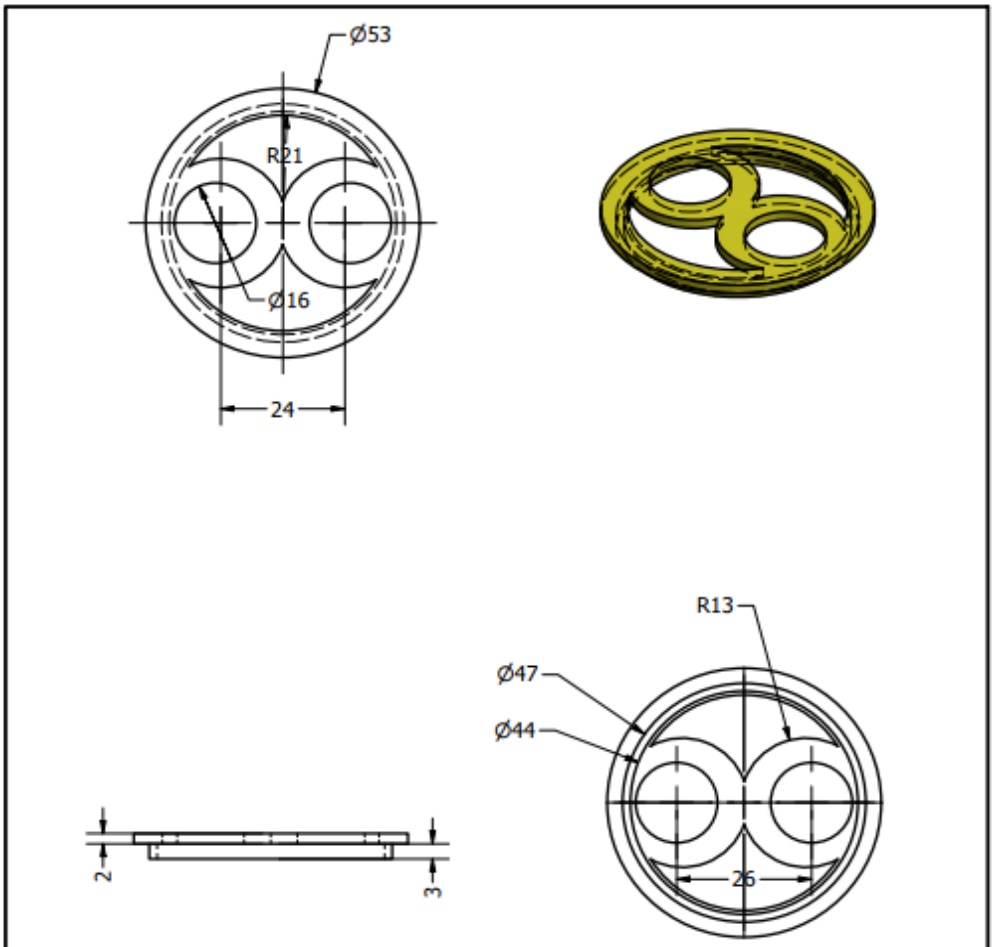
PLANO DE TUBO ADQUIRIDO PARA EL PROYECTO



UNIVERSIDAD POLITÉCNICA SALESIANA		**TÍTULO:** TUBO	
SEDE QUITO - CAMPUS SUR			
CARRERA DE INGENIERÍA ELECTRÓNICA			
TRABAJO DE TITULACIÓN			
TEMA:	ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO		
EST 1	ORDÓÑEZ ALARCÓN WILSON GABRIEL		
EST 2	PINO NOGUERA WELLINGTON ENRIQUE	**ESCALA:** 1:4	**UNIDADES:** mm

ANEXO VII

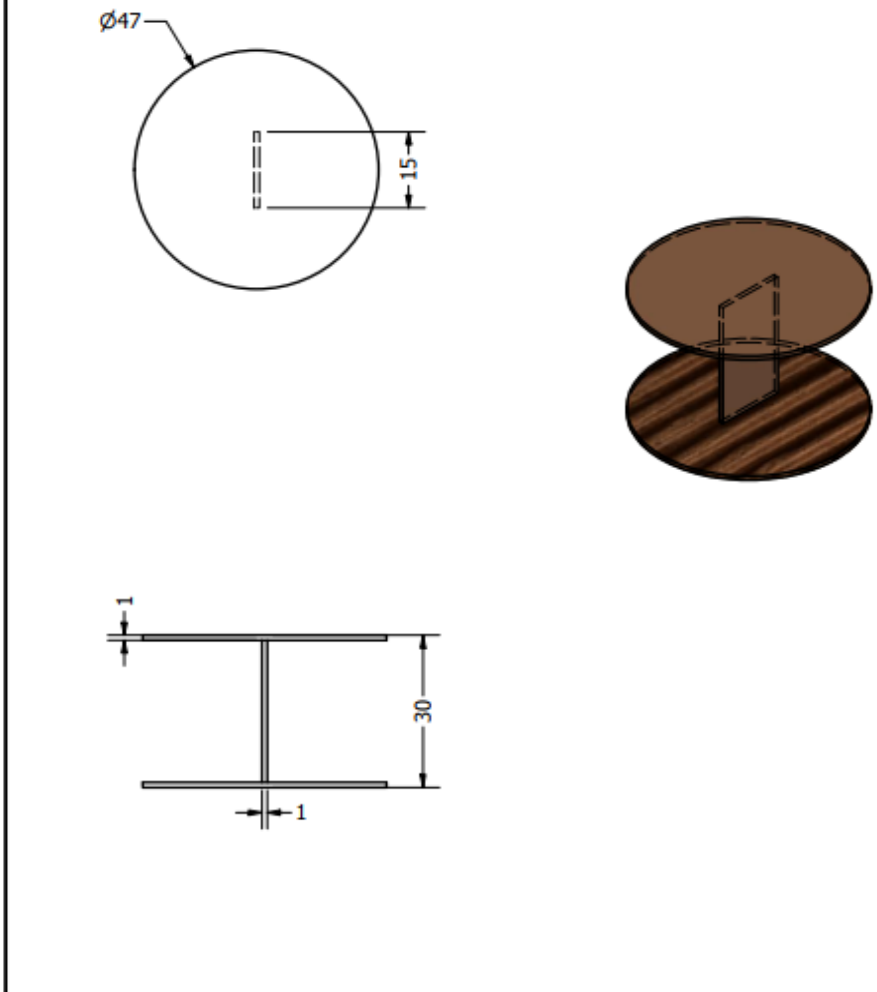
PLANO DE SOPORTE PARA SENSOR ULTRASÓNICO DISEÑADO



UNIVERSIDAD POLITÉCNICA SALESIANA		**TÍTULO:**	
SEDE QUITO - CAMPUS SUR		SOPORTE SENSOR	
CARRERA DE INGENIERÍA ELECTRÓNICA			
TRABAJO DE TITULACIÓN			
TEMA:	ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO		
EST 1	ORDÓÑEZ ALARCÓN WILSON GABRIEL		
EST 2	PINO NOGUERA WELLINGTON ENRIQUE	**ESCALA:** 1:1	**UNIDADES:** mm

ANEXO VIII

PLANO DE OBJETO A LEVITAR DISEÑADO



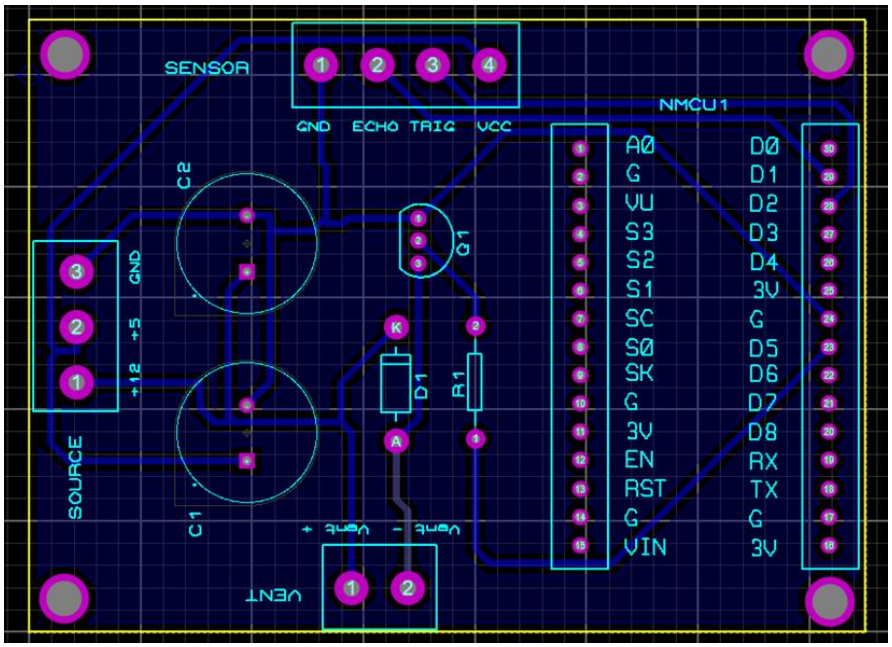
UNIVERSIDAD POLITÉCNICA SALESIANA		**TÍTULO:**	
SEDE QUITO - CAMPUS SUR		OBJETO A LEVITAR	
CARRERA DE INGENIERÍA ELECTRÓNICA			
TRABAJO DE TITULACIÓN			
TEMA:	ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO		
EST 1	ORDÓÑEZ ALARCÓN WILSON GABRIEL		
EST 2	PINO NOGUERA WELLINGTON ENRIQUE	**ESCALA:** 1:1	**UNIDADES:** mm

ANEXO IX

Diseño de tarjeta PCB

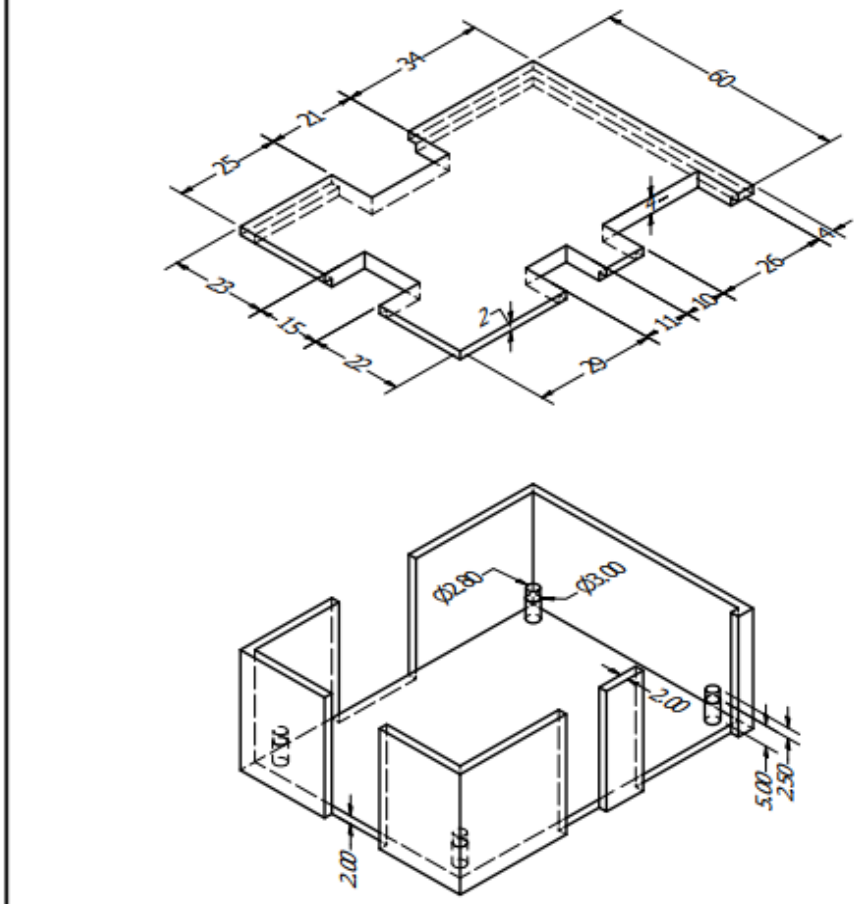
Vista 3D de la distribución de los elementos del PCB diseñado en software Proteus

Pistas del PCB diseñado en software Proteus



ANEXO X

Diseño de carcasa para placa PCB en software Inventor



UNIVERSIDAD POLITÉCNICA SALESIANA		**TÍTULO:**	
SEDE QUITO - CAMPUS SUR		CARCASA PCB	
CARRERA DE INGENIERÍA ELECTRÓNICA			
TRABAJO DE TITULACIÓN			
TEMA:	ALGORITMOS DE CONTROL ÓPTIMO BASADOS EN WNCS APLICADOS A UN LEVITADOR NEUMÁTICO		
EST 1	ORDÓÑEZ ALARCÓN WILSON GABRIEL		
EST 2	PINO NOGUERA WELLINGTON ENRIQUE	**ESCALA:** 1:1	**UNIDADES:** mm

ANEXO XI

Menús de configuración de proyecto y dispositivo a vincular en la App de Blynk

ANEXO XII

Descarga e Instalación de paquetes de software en Raspberry Pi

Comandos para descarga e instalación de node.js

a) Se accede al repositorio de node.js, según la versión de "setup" actual:

curl -sL https://deb.nodesource.com/setup_12.x | sudo bash –

b) Se instala node.js y su administrador de paquetes "npm":

sudo apt install nodejs

Comandos para descarga e instalación de Node-RED

a) Para acceder al repositorio e instalar Node-RED:

bash <(curl -sL https://raw.githubusercontent.com/Node-RED/linux-installers/master/deb/update-nodejs-and-nodered)

b) Para que se ejecute automáticamente en cada reinicio:

sudo systemctl enable nodered.service

Comandos para descarga e instalación de Mosquitto Broker

a) Se accede al repositorio de mosquitto:

wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key

sudo apt-key add mosquitto-repo.gpg.key

cd /etc/apt/sources.list.d/

b) Se selecciona la versión de Raspbian instalada:

sudo wget http://repo.mosquitto.org/debian/mosquitto-stretch.list

c) Se actualiza la base de datos:

sudo apt-get update

d) Se instala el Mosquitto Broker:

sudo apt install mosquitto mosquitto-clients

e) Para que se ejecute automáticamente en cada reinicio:

sudo systemctl enable mosquitto

ANEXO XIII

Comandos para descarga e instalación de servidor local de Blynk

a) Se instala java 8 en su versión actual:

sudo apt install openjdk-8-jdk openjdk-8-jre

b) Se descargan los paquetes de software de la plataforma del repositorio en GitHub propio de Blynk:

wget "https://github.com/blynkkk/blynk-server/releases/download/v0.41.13/server-0.41.13-java8.jar"

c) Se arranca el servidor por el puerto 9443 (SSL) que utiliza la App por defecto:

java -jar server-0.41.13-java8.jar -dataFolder /home/pi/Blynk

d) Para arrancar el servidor automáticamente en cada reinicio se abre el archivo de configuración de inicio de Raspberry:

crontab -e

e) Y se copia la siguiente línea al final del archivo "crontab -e" que ejecuta el servidor de Blynk instalado en cada inicio:

@reboot java -jar /home/pi/server-0.41.13-java8.jar -dataFolder /home/pi/Blynk &