UNIVERSIDAD RICARDO PALMA

FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

Implementación de control en cascada en Raspberry, para monitorear por Internet de las Cosas el nivel y flujo en un prototipo de tanque de riego

TESIS

Para optar el título profesional de Ingeniero Electrónico

AUTOR

Chacon Ruiz, Fabrizio Alberto
ORCID: 0009-0005-7748-1505

ASESOR

Huamani Navarrete, Pedro Freddy
ORCID: 0000-0002-3753-9777

Lima, Perú

2024

METADATOS COMPLEMENTARIOS

Datos del autor
Chacon Ruiz, Fabrizio Alberto
DNI: 71046976

Datos de asesor
Huamani Navarrete, Pedro Freddy
DNI: 10032682

Datos del jurado
JURADO 1
Burneo Gonzalez, Katia Janet
DNI: 09391942
ORCID: 0000-0002-7046-8106

JURADO 2
Rivas Leon, Javier Hipolito
DNI: 10250991
ORCID: 0000-0002-8365-4346

JURADO 3
Sánchez Bravo, Miguel Angel
DNI: 08443357
ORCID: 0000-0001-9384-1391

JURADO 4
Chong Rodriguez, Humberto
DNI: 07811343
ORCID: 0000-0002-4643-0538

Datos de la investigación
Campo del conocimiento OCDE: 2.02.03
Código del Programa: 712026

DECLARACIÓN JURADA DE ORIGINALIDAD

Yo, Fabrizio Alberto Chacon Ruiz, con código de estudiante N° 201510366, con DNI N° 71046976, con domicilio en Av. Los Lirios 6 Block A7 Dpt. 202, distrito San Juan de Miraflores, provincia y departamento de Lima, en mi condición de bachiller en Ingeniería Electrónica de la Facultad de Ingeniería, declaro bajo juramento que: La presente tesis titulada: "IMPLEMENTACIÓN DE CONTROL EN CASCADA EN RASPBERRY, PARA MONITOREAR POR INTERNET DE LAS COSAS EL NIVEL Y FLUJO EN UN PROTOTIPO DE TANQUE DE RIEGO" es de mi única autoría, bajo el asesoramiento del docente Dr. Ing. Pedro Freddy Huamani Navarrete y no existe plagio y/o copia de ninguna naturaleza, en especial de otro documento de investigación presentado por cualquier persona natural o jurídica ante cualquier institución académica o de investigación, universidad, etc.; la cual ha sido sometida al antiplagio Turnitin y tiene el 21 % de similitud final.

Dejo constancia que las citas de otros autores han sido debidamente identificadas en la tesis, el contenido de estas corresponde a las opiniones de ellos, y por las cuales no asumo responsabilidad, ya sean de fuentes encontradas en medios escritos, digitales o de internet.

Asimismo, ratifico plenamente que el contenido íntegro de la tesis es de mi conocimiento y autoría. Por tal motivo, asumo toda la responsabilidad de cualquier error u omisión en la tesis y soy consciente de las connotaciones éticas y legales involucradas.

En caso de falsa declaración, me someto a lo dispuesto en las normas de la Universidad Ricardo Palma y a los dispositivos legales nacionales vigentes.

Surco, 5 de abril de 2024

Fabrizio Alberto Chacon Ruiz

DNI N° 71046976

INFORME DE ORIGINALIDAD-TURNITIN

IMPLEMENTACIÓN DE CONTROL EN CASCADA EN RASPBERRY, PARA MONITOREAR POR INTERNET DE LAS COSAS EL NIVEL Y FLUJO EN UN PROTOTIPO DE TANQUE DE RIEGO

INFORME DE ORIGINALIDAD

21%	20%	4%	9%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	repositorio.urp.edu.pe Fuente de Internet	3%
2	hdl.handle.net Fuente de Internet	3%
3	Submitted to Universidad Ricardo Palma Trabajo del estudiante	1%
4	naylampmechatronics.com Fuente de Internet	1%
5	laccei.org Fuente de Internet	1%
6	idoc.pub Fuente de Internet	1%
7	repositorio.lamolina.edu.pe Fuente de Internet	1%
8	apirepositorio.unh.edu.pe Fuente de Internet	<1%

DEDICATORIA

A Dios, por guiarme en cada paso de mi vida, por darme la fuerza y motivación para seguir siempre adelante.

A mis padres Rosa y Francisco por su amor y apoyo incondicional durante toda la vida.

A mi sobrina Sofia que viene en camino, te esperamos con muchas ansias.

AGRADECIMIENTO

A mis profesores en pregrado, por sus enseñanzas y sus valiosos consejos que aportaron mucho a mi formación profesional.

A mi asesor el Dr. Pedro Huamani, por su apoyo y compromiso al decidir apoyarme con mi tema de investigación.

A mis amigos de la carrera, por el apoyo mutuo y el haber compartido memorables experiencias en la universidad.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

Esta tesis se centra en la implementación de un prototipo de tanque de riego para usarse en áreas rurales. El prototipo se basó en un tanque de agua con capacidad de 25 litros de agua y buscó optimizar las tareas de almacenamiento de agua y riegos programados mediante la implementación de un control en cascada y un control secuencial de riegos programados. La primera fase del proyecto implicó la elección y posicionamiento de los dispositivos de campo a instalarse en el tanque de riego, realización de ajustes y modificaciones para adaptarlos a las necesidades específicas del riego. La parte central de la implementación fue el diseño del algoritmo de control en cascada. Donde se encontraron los parámetros de ajuste ideales para el sistema, los cuales fueron Kp = 1.6, Ki = 1.7 para el controlador de flujo y Kp = 1.3 para el controlador de nivel. Para la habilitación del monitoreo remoto y la recopilación de datos, se utilizó la plataforma ThingSpeak. Esto permitió la supervisión en tiempo real de variables de control, lo que permite la toma de decisiones y la detección temprana de posibles problemas en el sistema. La última fase del proyecto consistió en diseñar un control secuencial para la programación de riegos automatizados. Esto brinda a los agricultores la capacidad de establecer horarios y condiciones específicas para el riego, lo que mejora la eficiencia del uso del agua.

*Palabras claves***:** Control en cascada, Internet de las cosas, Automatización, Python, Raspberry PI.

ABSTRACT

This thesis focuses on the implementation of a prototype irrigation tank for use in rural areas. The prototype was based on a water tank with a capacity of 25 liters of water and sought to optimize water storage and scheduled irrigation tasks by implementing cascade control and sequential control of scheduled irrigation. The first phase of the project involved the selection and positioning of the field devices to be installed in the irrigation tank, making adjustments and modifications to adapt them to the specific irrigation needs. The central part of the implementation was the design of the cascade control algorithm. Where the ideal adjustment parameters for the system were found, which were Kp = 1.6, Ki = 1.7 for the flow controller and Kp = 1.3 for the level controller. To enable remote monitoring and data collection, the ThingSpeak platform was used. This enabled real-time monitoring of control variables, facilitating data-driven decision making and early detection of potential system issues. The last phase of the project consisted of designing a sequential control for programming automated irrigation. This gives farmers the ability to set specific schedules and conditions for irrigation, improving water use efficiency.

Keywords: Cascade Control, Internet of Things, Automation, Python, Raspberry PI.

INTRODUCCIÓN

Este proyecto de tesis tiene como fin implementar un control y monitoreo remoto de la variable nivel y flujo en un prototipo de tanque de reserva de agua, utilizado para el riego de cultivos en localidades rurales. En la actualidad la agricultura ha recibido enormes beneficios con el Internet de las Cosas (IoT), lo cual ha dado origen a la agricultura inteligente o agricultura de precisión, debido a que muchos avances y soluciones tecnológicas han llegado para este campo con el fin de alcanzar el máximo partido de una explotación. Asimismo, la aparición de nuevos sensores, actuadores y demás dispositivos de campo se complementan para desarrollar una solución viable para estas aplicaciones. En esta investigación se unirán estas tecnologías y herramientas para desarrollar una solución eficiente para la distribución de agua en los riegos de cultivos; además, permitiendo un seguimiento de las variables de control a través de internet para un posible estudio posterior. En el mundo existen trabajos referentes al tema en cuestión, aunque la forma de tratar el asunto es muy variada, esta investigación se enfoca en desarrollar una solución económica y con el objetivo de disminuir costos de producción, optimizarla y permitir prácticas de riego más inteligentes sin descuidar la precisión que se requiere en estas aplicaciones industriales.

De esta manera, en el capítulo 1 se aborda lo referente al planteamiento del problema; asimismo, en el capítulo 2 se presenta al marco teórico del tema de investigación. Finalmente, en los capítulos 2 y 3, se desarrolla el proyecto y se realizan las pruebas respectivas para dejar evidencia de los resultados de esta investigación.

CAPÍTULO I: PLANTEAMIENTO Y DELIMITACIÓN DEL PROBLEMA

1.1 Formulación del problema

Al día de hoy, algunas localidades rurales de nuestro país donde se realiza la actividad agrícola son afectadas por sequías, debido a fenómenos climatológicos afectando varios sectores económicos, principalmente el de agricultura según el Servicio Nacional de Meteorología e Hidrología del Perú (Senamhi, 2016) . Este déficit hídrico afecta los regadíos de cultivos y su producción. Según datos de la Autoridad Nacional del Agua (ANA), la agricultura usa 80% del total de agua. La buena gestión de este recurso es de vital importancia para permitir un consumo eficiente y sostenible. Igualmente, la brecha tecnológica existente en nuestro país contempla la falta de sistemas de control de almacenamiento, distribución y regulación de agua para riego, así como también la integración de infraestructura aplicada al riego tecnificado; por lo tanto, estas deficiencias dificultan el regadío de cultivos y su producción. De esta manera, esta investigación se enfoca a desarrollar una propuesta a bajo coste orientada a la dificultad de almacenamiento y distribución de agua de un tanque de riego para zonas rurales con la utilización de un control en cascada implementado en un Raspberry PI 4 para monitorear periódicamente el nivel y flujo. Esta propuesta fue posible con la implementación del prototipo de tanque de riego utilizando un depósito cilíndrico a escala y no mayor a 1 metro de altura, incluyendo los dispositivos de campo adecuados tales como los sensores y actuadores, para lograr la precisión que se necesita; tomando en cuenta las normativas del estándar ISA 5.1. Además, un módulo de control local constituido por una pantalla LCD de 4x20 para establecer el setpoint del nivel del tanque, establecer riegos programados y visualizar alarmas e indicadores del sistema; por otro lado, se tiene en cuenta el diseño del algoritmo del controlador proporcional integral para un sistema de control en cascada, el cual se encuentra programado en el lenguaje de programación Python dentro de un Raspberry PI 4. Además, se cuenta con el uso de la plataforma ThingSpeak que ofrece de manera libre, acceso a sus aplicaciones IoT a través de una interfaz web gráfica, y donde es posible obtener información en tiempo real de las variables monitoreadas en el prototipo de tanque de riego.

1.1.1. Problema general

¿Cómo implementar un control en cascada en un Raspberry PI 4, para monitorear por Internet de las Cosas el nivel y flujo en un prototipo de tanque de riego, a escala, para las zonas rurales?

1.1.2. Problemas específicos

a. ¿Cómo implementar el prototipo de tanque de riego a escala incluyendo el posicionamiento de los instrumentos y equipos que permitirán controlar las variables de nivel y flujo?

b. ¿Cómo diseñar el algoritmo del controlador PI para el sistema de control en cascada en un Raspberry PI 4, de tal forma que se permita controlar las variables de nivel y flujo en un prototipo de tanque de riego a escala para una zona rural?

c. ¿Cómo monitorear los datos de las variables obtenidas del sistema de control en cascada a través del internet de las cosas, mediante de la plataforma ThingSpeak?

d. ¿Cómo diseñar e implementar un control secuencial de modo que se puedan programar tareas automatizadas de riego?

1.2 Objetivos

1.2.1. Objetivo general

Implementar un control en cascada en un Raspberry PI 4, para monitorear por Internet de las Cosas el nivel y flujo en un prototipo de tanque de riego a escala para las zonas rurales.

1.2.2. Objetivos específicos

a. Implementar el prototipo de tanque de riego a escala incluyendo el posicionamiento de los instrumentos y equipos que permitirán controlar las variables de nivel y flujo.

b. Diseñar el algoritmo del controlador PI para el sistema de control en cascada en un Raspberry PI 4, de tal forma que se permita controlar las variables de nivel y flujo en un prototipo de tanque de riego a escala para una zona rural.

c. Monitorear los datos de las variables obtenidas del sistema de control en cascada a

través del internet de las cosas, mediante la plataforma ThingSpeak.

d. Diseñar e implementar un control secuencial de modo que se puedan programar tareas automatizadas de riego.

1.3 Importancia y justificación

1.3.1. Importancia

Abordar este tema es importante porque permite contar con un sistema de riego automatizado y de bajo costo para ser utilizado en el riego de cultivos en localidades rurales, principalmente en las regiones costeras como Lambayeque, Piura, Ica, Moquegua y Tacna, donde el suministro de agua es muy limitado debido a las sequías originadas por la escasez continua de precipitaciones pluviales en ciertas temporadas; además, otorga múltiples beneficios al usuario para mejorar la producción de sus cultivos.

1.3.2. Justificación

Esta investigación se justifica porque se cuenta con materiales, instrumentos y equipos a disposición para poder implementar el sistema de control en cascada para controlar las variables de nivel y flujo en un prototipo de tanque de riego a escala, para una zona rural. También se justifica porque en algunas localidades rurales de nuestro país se producen sequías o se tiene poco abastecimiento del agua, lo que termina afectando la producción de los cultivos. Por ello, con esta propuesta de control en cascada para un prototipo de tanque de riego a escala, se alcanza el control de las variables nivel y flujo en el proceso con el propósito de llevar a cabo prácticas de riego más inteligentes.

1.4 Limitaciones

● Espacial: Este prototipo se implementó en el domicilio del tesista, debido a que fue dificultoso implementarlo en una planta real. También, se realizó la etapa de pruebas sin dejar de pasar por alto las consideraciones necesarias que requiere el control de un tanque de riego en una zona de cultivos.

● Teórica: Dado que se dificulta desarrollar la investigación en una planta real con dispositivos de campo industriales, este proyecto se limitó a una implementación del tipo prototipo, utilizando un tanque cilíndrico a escala y no mayor a 1 metro de altura. Además, un hardware disponible en el mercado nacional como es la computadora programable Raspberry PI 4, dispositivos de campo de bajo costo tales como los sensores de nivel y flujo, así como la electroválvula y la bomba de agua. Adicionalmente, se empleó una plataforma IoT de libre acceso como lo es ThingSpeak, que permite el monitoreo continuo de las variables de control a través de internet.

● Temporal: Esta tesis se enfoca en desarrollar un prototipo para la mejora del control y distribución de agua en un tanque de riego en las zonas rurales; sin embargo, debido a restricciones de tiempo y recursos, el alcance del estudio se limita a la implementación y evaluación del sistema en un periodo de seis meses a partir del inicio de la investigación. Por lo tanto, la tesis se centra únicamente en analizar los resultados obtenidos durante este lapso, sin considerar la evaluación a largo plazo del sistema y su efectividad a lo largo del tiempo. Es transversal.

CAPÍTULO II: MARCO TEÓRICO

2.1. Marco histórico

Acerca de la evolución del control en procesos industriales, Battikha (2017) relata lo siguiente:

> Incluso hace unos años, el alcance de la medida y control de procesos era mucho más sencillo de definir que en la actualidad. Fue referido simplemente como "instrumentación". Con el advenimiento de la funcionalidad basada en software y los avances tecnológicos en los distintos campos, esta especialidad ha comenzado a ramificarse en subespecialidades individuales.
>
> La medición y control de procesos, también conocida comúnmente como "Instrumentación y Control", ha evolucionado de una tecnología manual y mecánica a, sucesivamente, una tecnología neumática, electrónica y digital. El crecimiento exponencial de este campo vino luego de la Segunda Guerra Mundial, y su progreso hacia los sistemas y dispositivos basados en la tecnología digital sigue avanzando rápidamente en la actualidad.
>
> No sabemos con certeza quién inventó el control. Alrededor del 2600 a. C., los ingenieros egipcios seguramente estaban usando dispositivos de medición simples pero precisos para nivelar los cimientos y construir la Gran Pirámide y para cortar sus piedras a las dimensiones precisas. También utilizaron presas para medir y distribuir el agua de riego en el fértil delta. Muchos siglos después, los romanos construyeron sus acueductos y distribuyeron el agua mediante elementales caudalímetros.
>
> El tubo de Pitot se inventó en el siglo XVII. Luego, el regulador centrifugo para motores de vapor se inventó en 1774 durante la Revolución Industrial (con versiones mejoradas y todavía en uso en la actualidad). El regulador centrifugo se considera la primera aplicación de un concepto de controlador de retroalimentación.
>
> A fines del siglo XIX, los termómetros de caja de hojalata y de madera y los barómetros de mercurio comenzaron a estar disponibles comercialmente. Y a principios de la década de 1900, llegaron al mercado los registradores de datos, controladores neumáticos y controladores de temperatura.

Con el advenimiento de la Primera Guerra Mundial, hubo necesidad de instrumentos más eficientes ayudó a mejorar y desarrollar aún más el campo de la instrumentación. Se desarrollaron salas de control y surgió el concepto de control proporcional, integral y derivado (PID). A mediados de la década de 1930, se desarrollaron analizadores, caudalímetros y potenciómetros electrónicos. En ese momento había más de 600 empresas que vendían instrumentos industriales.

A principios de la década de 1940, se desarrolló el método de afinación Ziegler-Nichols (todavía en uso en la actualidad). La Segunda Guerra Mundial fue una gran influencia para mover el control a un nuevo nivel. Se produjeron transmisores de presión, instrumentos totalmente electrónicos e instrumentos de equilibrio de fuerza. Durante la década de 1950, la industria del control de procesos se transformó con la introducción del transistor. Durante este período también se introdujeron en el mercado los siguientes: transmisores neumáticos de presión diferencial, controles electrónicos y el rango de señal de CC de 4-20 mA.

Ya en la década de 1960, se introdujeron las computadoras junto con la implementación del control digital directo (DDC) con interfaces de operador basadas en tubos de rayos catódicos (CRT), los controladores lógicos programables (PLC), el medidor de vórtice y válvulas de control mejoradas. La década de 1970 trajo el microprocesador, los sistemas de control distribuido (DCS), la transmisión por fibra óptica, los analizadores de oxígeno in situ y el chip de memoria ram.

Las décadas de 1980 y 1990 vieron el advenimiento de la computadora personal y la era del software, que amplió la aplicación de DCS y PLC. También se introdujeron las redes neuronales, los sistemas expertos, la lógica difusa, los instrumentos inteligentes y los controladores autoajustables.

El futuro de la medición y el control es desconocido; sin embargo, basándose en las tendencias actuales, se espera que la línea de demarcación entre los DCS y los PLC siga desapareciendo, que aumenten el autodiagnóstico y la autorreparación, que la inteligencia artificial aumente su aceptación y su facilidad de uso, y que un sistema de bus de comunicación estándar para toda la planta se convierta en la norma. (p. 16)

2.2. Investigaciones relacionadas con el tema

La investigación de Bermeo et al. (2021) presentó una comparación controlador PID sobre el proceso de la variable nivel implementando dos tecnologías diferentes, un controlador lógico programable y un Raspberry PI. Ambos controladores utilizaron un controlador PID sintonizado con los mismos parámetros; por lo cual, en su fase de pruebas notaron que los controladores tuvieron el mismo rendimiento en sus parámetros de control, solo se diferenciaron en el esfuerzo de control donde el PLC tuvo 2,10% menos esfuerzo de control que el Raspberry PI y un sobreimpulso de señal de 2,05% en comparación con el 4,32% del Raspberry PI. Los resultados mostraron que la implementación de un controlador con un Raspberry PI es buena alternativa para procesos con parecidos resultados que con un PLC.

Comentario:
A través de la comparación de los resultados, el artículo proporciona una evaluación del desempeño relativo del controlador PID en un PLC y en un Raspberry, también proporciona información sobre la configuración, ajuste y evaluación de los controladores, brindando una comparación objetiva de su desempeño en diferentes plataformas de control. Esto ayuda a los investigadores a tener una comprensión más clara de las virtudes de cada plataforma de control.

En el trabajo de Fuentes et al. (2018) se propuso el cotejo de controladores digitales, controlador P, PI y PID en Raspberry y programados en Python para controlar temperatura en un horno a pequeña escala donde los parámetros P, I y D fueron calculados con PID Tuner; además, utilizaron un Arduino UNO como conversor analógico digital e inversamente, y un display para la interfaz gráfica. La eficiencia se evaluó con la comparación de los tres modelos de controladores y del rendimiento del procesador del Raspberry. Llegaron a la conclusión que lo mejor que se probó en Raspberry fue el controlador PI, ya que presentó un tiempo de establecimiento menor de la variable en función de los controladores, además se controló con precisión la temperatura. En cuanto a la rapidez del algoritmo se presentó mejor tiempo en el controlador PI.

Comentario:
La investigación describe detalladamente el diseño en Raspberry, así como la configuración y ajuste de los controladores PID. Además, se llevaron a cabo experimentos

y pruebas utilizando diferentes parámetros de control para evaluar el rendimiento del sistema en términos de precisión, estabilidad y respuesta a perturbaciones. Los resultados obtenidos se analizan y se comparan con otros métodos de control, brindando una evaluación del rendimiento de los controladores digitales clásicos en el proceso térmico estudiado. Proporciona información sobre la implementación, ajuste y evaluación de estos controladores, ofreciendo una perspectiva práctica y aplicada en el campo del control digital.

En el trabajo de Álvarez et al. (2017) se cuenta la implementación del sistema de control en una planta térmica, basado en la estrategia cascada temperatura-flujo que incluyó transmisores de flujo y temperatura, controladores lógico programables y válvulas de control; el modelamiento de lazos de control fue por funciones de transferencia, en el caso del diseño de los controladores de flujo y temperatura la fijación de los parámetros de sintonía fue por Matlab. Asimismo, decidieron realizar la simulación de 4 controladores PID por medio de Simulink del Matlab para evaluar el rendimiento de acuerdo a los parámetros de sintonía establecidos, y llegaron a la conclusión que su metodología aplicada conllevó a resultados satisfactorios, observando que la velocidad de respuesta en lazo cerrado fue mejor para los controladores. De igual manera en los lazos de control se mostró error de estado estacionario pequeño al igual que el porcentaje de sobreimpulso; además, sus resultados presentaron características iguales a la simulación.

Comentario:
El estudio se centra en el desarrollo de un sistema de control avanzado que utiliza una estructura de control en cascada para optimizar la respuesta y la precisión del proceso de intercambio térmico. Se detalla el diseño del controlador en cascada, que consiste en un controlador primario y un controlador secundario. Además, se presenta la implementación práctica del sistema de control en la planta térmica. Se describen los componentes y equipos utilizados, así como los detalles de la interconexión y la configuración.
Destaca el uso de controladores primarios y secundarios para mejorar la respuesta y la precisión del proceso; así como también, el estudio proporciona una solución efectiva para el control avanzado de sistemas de intercambio térmico, lo que puede resultar en un mejor rendimiento y eficiencia en aplicaciones industriales.

En el trabajo de Dimas y Huamani (2019), se implementó un control en cascada, en el laboratorio de control de la Universidad Ricardo Palma, se utilizaron transmisores de nivel y flujo, un PLC con sus módulos de entrada y salida, una pantalla, además el software Step 7 Basic de TIA Portal, donde se configuraron los equipos. Así como también, se configuró el PLC con programación Ladder; por otro lado, se utilizó el método de ensayo y error como procedimiento de sintonización. En el diseño se estableció la variable flujo como lazo secundario debido a su velocidad y el nivel como primario, por ello la salida del controlador primario fue la entrada de Setpoint para el controlador secundario. Además, se llegó a la conclusión de que controlador secundario sea implementando como controlador P, mientras que el primario con controlador PI para mejor respuesta.

Comentario:
El estudio se enfoca en mejorar el rendimiento y la precisión del control de nivel en un sistema de laboratorio utilizando un control en cascada. Esta estrategia implica el uso de dos controladores en serie: un controlador primario y un controlador secundario. El controlador primario se encarga de regular la variable de nivel y el controlador secundario ajusta el flujo de entrada al proceso para mantener el nivel deseado. El estudio proporciona información sobre el diseño, configuración y evaluación del sistema de control en cascada, destacando los beneficios en términos de rendimiento y precisión en comparación con un controlador de lazo único convencional. Se detallan los pasos para la configuración y ajuste de los controladores, así como los experimentos realizados para evaluar el desempeño del sistema. Finalmente se analizan los resultados obtenidos, incluyendo métricas de desempeño como el error de seguimiento, la respuesta transitoria y la estabilidad del sistema. Los resultados muestran una mejora significativa en comparación con un sistema de control convencional de un solo lazo.

En Mogrovejo (2017) se presentó un trabajo que sirvió de modelo para al diseño de controladores de sistemas de múltiples entradas y múltiples salidas, para ello se diseñó e implementó un controlador PID multivariable robusto que permitió controlar cuatro tanques interconectados. El algoritmo empleado para el diseño del controlador se basó en desigualdades matriciales lineales iterativas (ILMI); y para la implementación de los dispositivos de campo se emplearon sensores de flujo, presión y nivel, así como un módulo de adquisición de datos; posteriormente, se programó el sistema en la plataforma

LabVIEW. Finalmente, llegaron a la conclusión que el sistema de tanques interconectados presentó mejoras en comparación con sistemas comerciales de 2 tanques, y afirmaron que el diseño del controlador PID robusto multivariable aseguró estabilidad en lazo cerrado para los puntos de operación del sistema.

Comentario:
En el estudio se muestra cómo se aborda el desafío de controlar de manera eficiente y precisa los niveles de líquido en los cuatro tanques, considerando las interacciones entre ellos. Se emplea una estrategia de control PID robusta, que tiene en cuenta la incertidumbre en el modelo del sistema y busca mantener los niveles deseados de manera estable y rápida. Se describe detalladamente el diseño del controlador PID robusto multivariable, incluyendo la metodología de sintonización utilizada y los parámetros empleados. También se presenta la implementación del sistema en un entorno experimental, mostrando los resultados obtenidos y comparándolos con otros métodos de control. El autor considera la importancia de abordar las interacciones entre los tanques y se muestra cómo el enfoque de control PID robusto puede lograr un control eficiente y preciso en este tipo de sistemas.

El trabajo de García (2016) dio a conocer la programación de un controlador PID, utilizando un Raspberry PI y tarjetas de adquisición de datos. También se diseñó una interfaz donde el usuario puede configurar los procesos, el lenguaje utilizado para programar el controlador PID fue Python y los lenguajes utilizados para la programación de la interfaz web fue HTML, CSS y JavaScript. Además, se llegó a la conclusión que los resultados fueron buenos ya que se control las variables, su supervisión y la configuración los parámetros a través de su interfaz.

Comentario:
El estudio se enfoca en la creación de un controlador PID, utilizando hardware y software accesibles, Raspberry es ampliamente utilizado. Se explica detalladamente el proceso de diseño y configuración del controlador, así como la implementación del algoritmo de control PID, también destaca la versatilidad de uso de Raspberry PI para la implementación de sistemas de control en aplicaciones industriales, y se muestra cómo se puede aprovechar su capacidad de procesamiento y su conectividad para lograr un control preciso y confiable de la temperatura. Se resaltan las ventajas de utilizar esta

plataforma en aplicaciones de control de temperatura y se presenta una solución accesible y efectiva para el control de procesos.

En la tesis de Valderrama (2020) se nota la integración de tecnología IoT en proyectos de automatización, es así que en dicho trabajo se utilizó la plataforma ThingSpeak para el control remoto de un invernadero, así como también se recogieron principalmente datos de temperatura, humedad ambiental y humedad de suelo. Estos datos fueron transferidos a un software de monitoreo de datos y a una base de datos inalámbrica en una plataforma IoT, empleando una terminal de comunicación inalámbrica GPRS. Finalmente, los clientes pudieron revisar la información sobre la temperatura y la humedad de sus cultivos en la página web durante las 24 horas de manera semanal o mensual. ThingSpeak no solamente permitió recibir y enviar datos y realizar monitoreo de pantalla, sino que también posibilitó la comunicación con los entornos de programación Matlab & Simulink, y así utilizar los datos medidos para la construcción de modelos matemáticos y simulaciones que permitieron estudiar dinámicas internas del microclima, fenómenos de transferencia de calor dentro del invernadero. Adicionalmente, se utilizaron tarjetas Arduino UNO, módulo de comunicación WiFi ESP8266, y módulos relé como arquitectura del sistema de control. La programación de código en Arduino permitió realizar el control de los dispositivos a través de reglas de decisión con el fin de mantener los parámetros del micro clima alrededor de un rango deseado o de un umbral establecido, permitiendo el envío de datos hacia ThingSpeak.

Comentario:
El artículo describe la configuración y la interconexión de los dispositivos y sensores utilizados en el micro invernadero automatizado. Se detalla cómo se recopilan los datos ambientales a través de los sensores y se envían a ThingSpeak para su visualización y análisis. Además, se discute la implementación del control remoto, que permite a los usuarios ajustar las condiciones ambientales dentro del invernadero a través de la plataforma ThingSpeak. Esto incluye el control de dispositivos como sistemas de riego y sistemas de iluminación. El uso de esta plataforma proporciona una solución eficiente y práctica para el monitoreo y control de invernaderos, lo que puede contribuir a optimizar el crecimiento de las plantas y mejorar la eficiencia en la agricultura.

2.3. Estructura teórica y científica que sustenta el estudio

2.3.1. Control en cascada

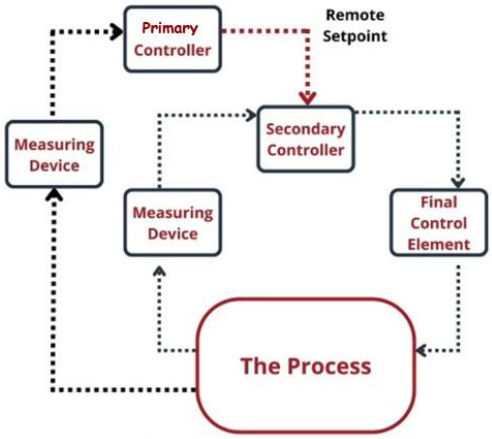
En el sitio web de Control Automation (2020) se explica el funcionamiento del control en cascada desde el siguiente punto de vista:

> Conectar controladores en cascada significa unir la salida de un controlador a la entrada del otro controlador. El primer controlador (llamado primario) esencialmente "da órdenes" al segundo controlador (llamado secundario) a través de una señal de punto de ajuste. Por lo tanto, un sistema de control en cascada consta de dos lazos de control de retroalimentación. (p. 1)

El contenido de la cita anterior se aclara mejor con la figura 1.

Figura 1

Sistema de control en cascada



Nota. https://control.com/textbook/basic-process-control-strategies/cascade-control/

El autor Woolf (2022) describe las condiciones necesarias para efectuar el control en cascada de la siguiente manera.

> Se debe mantener la jerarquía a nivel de los lazos para poseer un flujo de información en el sistema de control. En el sistema en cascada, la acción del lazo secundario debe ser mucho más rápida que la acción del lazo primario. Esto garantiza que las variaciones en la salida principal se reflejen rápido en el proceso y se observen cuando se calcule la siguiente variable. Esta jerarquía se puede mantener aplicando algunas condiciones a tener en cuenta:

1. Tiene que haber una relación notoria entre las variables de los lazos primario y secundario.
2. El lazo secundario debe producir efectos sobre el lazo primario.
3. El tiempo de respuesta del lazo primario debe ser al menos 3 veces mayor que el tiempo de respuesta del lazo secundario.
4. La principal perturbación en el sistema debe estar en el lazo primario.
5. El lazo primario debería poder tener una gran ganancia. (p. 1)

Asimismo, el autor Woolf (2022) también afirma que:

> En otros términos, el control en cascada es más conveniente cuando el lazo interno controla un proceso de alta frecuencia. Este está diseñado para que el controlador primario responda a cambios lentos, mientras que el secundario controla las perturbaciones que suceden con velocidad. Si se establece al revés, causará muchos errores. Por lo tanto, es importante mantener la jerarquía de la información. También se requiere que el control interno se configure bien para que no suceda la acumulación de errores. (p. 2)

En la tabla 1 se observan las ventajas y desventajas del esquema de control en cascada.

Tabla 1

Ventajas y desventajas del control en cascada

Ventajas	Desventajas
Los lazos que emplean correctamente la arquitectura en cascada responden más eficazmente a las perturbaciones.	La arquitectura requiere la instalación y el uso de un segundo sensor para medir la variable interna del proceso.
El bucle interno ayuda a corregir las no linealidades como la fricción estática que están asociadas con el elemento de control final	El control en cascada requiere más equipos e instrumentos que aumentarán el costo del proceso

Un bucle interno más rápido reduce la variabilidad general experimentada por el proceso.	Existe la posibilidad de un mayor desgaste del elemento final de control del proceso
El bucle exterior se puede sintonizar de forma más conservadora.	Los costos de configuración también son casi el doble.
Se puede combinar con control anticipativo u otras formas de control	El control en cascada hace que el sistema sea más complejo

Nota. https://controlstation.com/blog/pros-cons-cascadecontrol/

2.3.2. Raspberry PI 4

En el sitio web de Raspberry (2019) se describe la Raspberry PI como:

> Raspberry PI es un hardware asequible y con un tamaño reducido, puede ser usada como una computadora común. Tiene un sistema operativo basado en Linux capaz de permitir a los usuarios conocer más acerca de computación y aprender lenguajes de programación. Como computadora de escritorio es capaz de navegar en internet, reproducir videos, crear o editar documentos, hasta usarlo como consola de juegos.
>
> Además, Raspberry puede ser utilizado en gran cantidad de proyectos, desde robótica hasta decoders de video por internet. (p. 1)

En los siguientes subtítulos a, b y c se describen las entradas y salidas configurables de Raspberry PI, las especificaciones técnicas de la placa y la programación Python dentro de Raspberry PI.

El sitio web de Raspberry (2019) define los GPIO como:

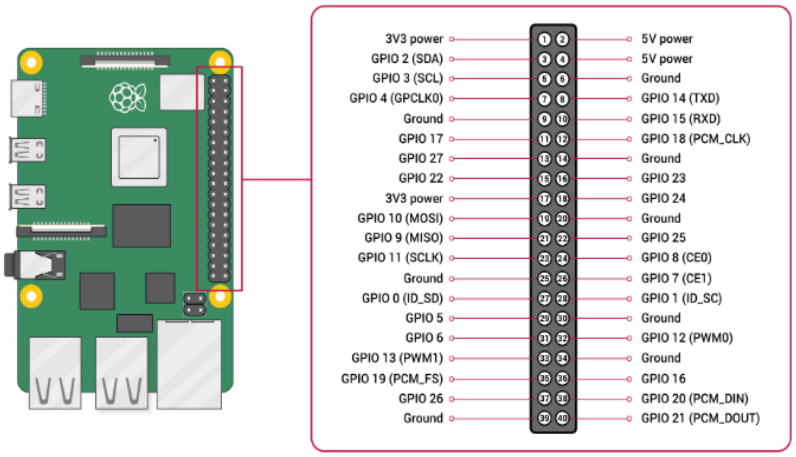
> a) GPIO y el encabezado de 40 pines
>
> La característica esencial de Raspberry PI son los pines GPIO (entradas y salidas) dentro de la tarjeta. El cual posee 40 pines en todas las placas Raspberry PI. Hay pines de fuente y tierra que no se pueden configurar. Los demás pines son pines son de 3,3V, lo que significa que las salidas tienen una señal de 3,3V y las entradas

solo admiten hasta 3,3V. El encabezado GPIO en sus placas tienen la clavija de 0,1" (2,34 mm). (p. 2)

En la figura 2 se detalla la utilidad de los GPIO en Raspberry PI 4.

Figura 2

Pines de entrada y salida del Raspberry PI 4 Model B



Nota. https://www.raspberrypi.com/docu mentation/computers/raspberry-pi.html#raspberry-pi-4-model-b

b) Especificaciones técnicas

Seguidamente, en la tabla 2 se observan las principales especificaciones técnicas del Raspberry PI4.

Tabla 2

Especificaciones técnicas de Raspberry PI 4

SoC Broadcom BCM2710, Cortex-A72 de cuatro núcleos (ARM v8) de 64 bits a 1,5 GHz
SDRAM LPDDR4-3200 de 8 GB
2,4 GHz IEEE 802.11ac, Bluetooth 5.0, Gigabit Ethernet
4 puertos USB 3.0
GPIO de 40 pines de fuente, tierra, entrada y salida
2 puertos micro-HDMI (4kp60)
H.265 (descodificación 4kp60), H264 (descodificación 1080p60, codificación 1080p30)

GPU OpenGL ES 3.1, Vulkan 2.0
Ranura de tarjeta micro-SD

Nota. https://www.raspberrypi.com/raspberry-pi-4-model-b/specifications/

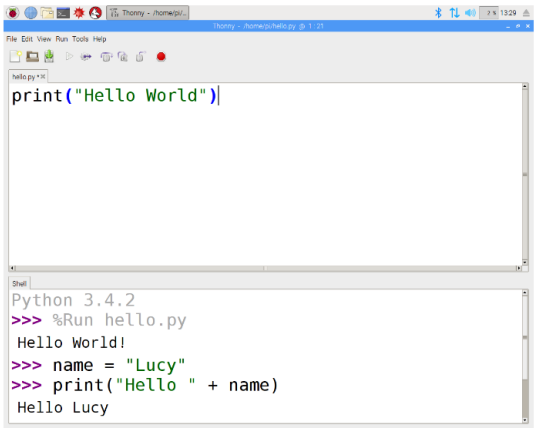
El sitio web de Raspberry (2019) define el uso de Python de la siguiente manera:

> c) Python en Raspberry PI
>
> Python es un lenguaje de programación fácil de usar en Raspberry PI, permite introducir un proyecto en la realidad. La sintaxis del lenguaje Python es legible y usa sentencias clave estándar. La manera más intuitiva de interactuar con Python es usando el entorno Thonny, Puede acceder a Thonny desde el escritorio. En la aplicación, esto se llama la ventana Shell. Thonny también tiene resaltado de sintaxis incorporado y cierto soporte para el autocompletado. Puede mirar hacia atrás en el historial de los comandos que ha ingresado. (p. 3)

En la figura 3 se puede observar el entorno de desarrollo de Thonny en Raspberry PI.

Figura 3

Entorno de desarrollo integrado de Thonny



Nota. https://magpiraspberrypi.com/articles/thonny

2.3.3. Sensor de ultrasonido JSN-SR04T

En el sitio web de Naylamp (2021) se describe el sensor de ultrasonido JSN-SR04T de la siguiente manera:

> El sensor de ultrasonido JSN-SR04T es un sensor de nivel que usa el ultrasonido para calcular la distancia de objetos. Una característica especial es su resistencia al agua, tamaño reducido, poco consumo eléctrico y precisión. Además, es útil en

> aplicaciones donde el sensor está sometido a salpicones de agua, ambientes con humedad, también es usado en autos para calcular distancia de parqueo. El sensor contiene toda la electrónica que permite la medición. El trabajo del sensor sucede cuando primero emite un pulso de sonido, además se mide el ancho del pulso de retorno, la medición se calcula a partir de las diferencias de tiempos entre el pulso de ida y el pulso de retorno. El sensor no se ve afectado por el solar o los objetos de color negro. (p. 1)

El sensor de ultrasonido descrito en la anterior cita se muestra en la figura 4.

Figura 4

Sensor de ultrasonido JSN-SR04T

Nota.https://naylampmechatronics.com/sensores-proximidad/326-sensor-ultrasonido-jsn-sr04t.html

En cuanto a las especificaciones técnicas del sensor de ultrasonido JSN-SR04T, según Naylamp (2021), se observan en la tabla 3:

Tabla 3

Especificaciones técnicas del sensor de ultrasonido JSN-SR04T

Modelo	JSN-SR04T
Voltaje de trabajo	5VDC
Corriente nominal	30mA
Rango	25cm - 450cm
Precisión	+-0.3mm
Frecuencia de emisión	40KHz
Duración mínima del pulso de disparo	10μS
Periodo mínimo de espera entre una medición y el inicio de otra	20mS
Ángulo de detección	menor a 50º
Resistencia al agua	Si
Dimensiones PCB	41*28.5 mm
Dimensiones sensor	D25*L19 mm
Temperatura de operación	-10ºC hasta 70ºC

Nota. https://naylampmechatroni cs.com/sensores-proximida d/326-sensor-ultrasonido-jsn-sr04t.html

En el Anexo A se muestran las características y especificaciones técnicas del sensor de ultrasonido JSN-SR04T en su totalidad.

2.3.4. Sensor de flujo de agua YF-S201

En el sitio web de Naylamp (2021) se describe el sensor de flujo de agua YF-S201 de la siguiente manera:

> El sensor de flujo de agua YF-S201 es un instrumento para la medición de caudal de un fluido. El caudal es la cantidad de fluido que ingresa a través de una tubería, se expresa en: litros por minutos (l/m). Los caudalímetros se instalan en las tuberías que circulan los fluidos. También se llaman flujómetros. El sensor de

> flujo de agua de YF-S201 es de tipo turbina además está aislado del agua, por lo cual se mantiene seguro., se usa para calcular el caudal de agua en tuberías de 1/2" de diámetro. También puede ser usado para realizar mediciones de otros líquidos como el combustible. El sensor tiene tres pines: fuente, tierra y la salida de pulsos. El trabajo del sensor es el siguiente: el caudal ingresa al sensor y gira la turbina, dicha turbina está unida a un imán que activa un sensor de efecto Hall, a su vez emite un pulso eléctrico que es admitido en la entrada de Raspberry PI. Se determina el caudal a partir de la cantidad de pulsos por segundo y la cantidad de volumen por pulso. (p. 2)

El sensor de flujo descrito en la anterior cita se muestra en la figura 5.

Figura 5

Sensor de flujo de agua YF-S201

Nota. https://naylampmecha t ro nics.com/sensores-liquido/108-sensor-de-flujo-de-agua-12-yf-s201.html

En cuanto a las especificaciones técnicas del sensor de flujo de agua YF-S201, según Naylamp (2021), se observan en la tabla 4:

Tabla 4

Especificaciones técnicas del sensor de flujo de agua YF-S201

Modelo	YF-S201
Voltaje de fuente	5V - 18V DC
Consumo nominal	15mA (5V)
Capacidad de carga	10mA (5 VDC)
Salida	Onda cuadrada pulsante
Rango	1-30L/min
Volumen promedio por pulso	2.25mL
Pulsos por litro	450
Factor de conversión	7.5
Presión de trabajo máx	1.75MPa (17 bar)
Conector tubería	Rosca externa 1/2" NPS Macho
Temperatura de funcionamiento	-25ºC a 80ºC

Nota. https://naylampmechatronics.com/sensores-liquido/108-sensor-de-flujo-de-agua-12-yf-s201.html

En el Anexo B se muestran las características y especificaciones técnicas del Sensor de flujo de agua YF-S201 en su totalidad.

2.3.5. Válvula solenoide FDP-270A

En el sitio web de Naylamp (2021) se describe la válvula solenoide de la siguiente manera:

> La válvula solenoide FDP-270A es un tipo de válvula de apertura y cierre. Posee dos componentes: el solenoide y el cuerpo de plástico. El solenoide es un electroimán que al energizarse se apertura el diafragma y permite el paso del fluido. Cuando no está energizado la válvula se cierra. El material de la válvula es plástico con conexión rosca macho de 1/2". La mayoría de tuberías en las viviendas del Perú poseen un diámetro de 1/2", esta válvula es buena opción para controlar el flujo de agua en un hogar. (p. 3)

La válvula solenoide descrita en la anterior cita se muestra en la figura 6.

Figura 6

Válvula Solenoide FPD-270A

Nota. En la figura 6 se muestra la válvula solenoide. Fuente: https://naylampmechatronics.com/valvulas/314-valvula-solenoide-1p2-pulg-12vdc-nc.html

En cuanto a las especificaciones técnicas de la válvula solenoide FPD-270A, según Naylamp (2021), se observan en la tabla 5:

Tabla 5

Especificaciones técnicas de la válvula solenoide FPD-270A

Modelo	FPD-270A
Voltaje de fuente	12VDC
Corriente nominal	0.6A
Consumo	8W
Presión de funcionamiento mínima	0.02 MPa (0.2 Bar = 2.04 mca)
Presión de funcionamiento máxima	0.8 MPa (8 Bar = 81.6 mca)
Apertura	≤ 0.15 s
Cierre	≤ 0.3 s
Conexión	Rosca externa 1/2" NPS Macho
Tipo	Normalmente cerrado
Dimensiones	85*60*26 mm

Nota. https://nmpmechatronics.com/valvula-solenoide.html

En el Anexo C se muestran las características y especificaciones técnicas de la válvula solenoide FDP-270A en su totalidad.

2.3.6. Bomba de agua 5M 800L/H

En el sitio web de Naylamp (2021) se describe la bomba de 5M 800L/H agua de la siguiente manera:

> La bomba de agua 5M 800L/H es centrífuga y tiene conexión rosca de 1/2" de diámetro. La bomba es de tamaño pequeño, bajo consumo, poco ruido, trabajo continuo y cuenta con un motor interno del tipo sin escobillas lo que asegura una larga vida útil sin necesidad de mantenimiento. Su potencia permite hasta un flujo 800 litros por hora. Además, se utiliza en diferentes trabajos como riego programado, llenado de tanques, etc. (p. 4)

La bomba de agua descrita en la anterior cita se muestra en la figura 7.

Figura 7

Bomba de agua 5M 800L/H

Nota. https://naylampmechatronics. com/bombas-de-agua/446-bomba-de-agua-d12-12vdc-5m-800lh.html

En cuanto a las especificaciones técnicas de la bomba de agua 5M 800L/H, según Naylamp (2021), se observan en la tabla 6:

Tabla 6

Especificaciones técnicas de la bomba de agua 5M 800L/H

Voltaje de fuente	5-12V DC
Corriente nominal	350mA
Potencia nominal	19W
Máximo caudal	800L/h (13L/min)
Altura máxima	5m
Conexión	Rosca 1/2" Macho
Protección	IP68
Ruido	<40dB a 0.5m
Líquidos de trabajo	agua, aceite, gasolina
Temperatura	100°C máx.
Dimensiones	80*80*65 mm aprox.

Nota. https://naylampmechatronics.com/bombas-deagua/446-bomba-d e-agua-d12-12vdc-5m-800lh.html

En el Anexo D se muestran las características y especificaciones técnicas de la Bomba de agua en su totalidad.

2.3.7. Servomotor MG996R

En el sitio web de Naylamp (2021) se describe el servomotor de la siguiente manera:

> El servo MG996R Tower Pro tiene un buen torque ya que posee engranajes metálicos que le proveen fiabilidad. Se utiliza mayormente en proyectos de robótica. Rota a 180 grados, Se adapta a diferentes entornos de desarrollo como Raspberry PI y otros microcontroladores. Posee 3 pines tierra, fuente y señal de control PWM. Es recomendable suministrar energía al servomotor de una fuente externa para evitar el ruido eléctrico y posibles daños en la placa de Raspberry PI. (p .5)

El servomotor descrito en la anterior cita se muestra en la figura 8.

Figura 8

Servomotor MG996R Tower Pro

Nota. https: //naylampmechatronics.co m/servomotores/343-servo-mg996r-11kg.html

En cuanto a las especificaciones técnicas del servomotor MG996R Tower Pro, según Naylamp (2021), se observan en la tabla 7:

Tabla 7

Especificaciones técnicas del servomotor MG996R Tower Pro

Voltaje de Alimentación	6-7.2V DC
Torque	9.4kg/cm (4.8V), 11kg/cm (6V)
Velocidad del servomotor	(4.8V sin carga): 0.2 seg / 60° (6V sin carga): 0.16 seg / 60°
Ángulo máximo	0-180°
Periodo del pulso	20ms
Ancho del pulso	entre 500us y 2400us
Ancho de banda muerto	20useg
Tipo de engranaje	de metal
Temperatura	-30 a +60 °C
Dimensión	40.6*19.8*42.9 mm
Peso	55 gramos

Nota. https://naylampmechatronics.com /servomotores/343-servo-mg996r-11kg.html

En el Anexo E se detallan las características y especificaciones técnicas del servomotor en su totalidad.

2.3.8. ThingSpeak

Fue utilizado para solucionar el monitoreo de datos de las variables de control a través de internet, y según ThingSpeak (2022) se define como:

> ThingSpeak es un servicio de análisis de datos en la nube que permite agregar, visualizar y analizar flujos de datos en tiempo real desde internet. ThingSpeak proporciona gráficos de los datos ingresados por los elementos de medición en ThingSpeak. Tiene la capacidad de integrar código MATLAB® dentro, para realizar un análisis y procesamiento de los datos. ThingSpeak se utiliza usualmente para la implementar prototipos y probar el concepto de sistemas IoT. (p. 1)

Y como también ThingSpeak (2022) afirma que su plataforma ofrece lo siguiente:

> ThingSpeak permite ingresar, observar y procesar flujos de datos en tiempo real. Algunas funcionalidades de ThingSpeak son:
>
> - Configuración fácil de sensores que envían datos a ThingSpeak mediante protocolos IoT.
> - Visualización de datos de sensores en vivo.
> - Adición de datos a fuentes externas.
> - Procesamiento de MATLAB
> - Ejecución de análisis de automático en función de horarios o eventos.
> - Prototipado y desarrollo de sistemas IoT, sin la necesidad de configurar hardware y software.
> - Acción automática sobre los datos y comunicación utilizando servicios de terceros como Twilio o Twitter. (p. 2)

2.4. Definición de términos básicos

2.4.1. Control de lazo cerrado

Según Kavianpour (2017), "El control de lazo cerrado usa retroalimentación para determinar si su salida ha alcanzado el objetivo deseado. Además, el sistema de lazo cerrado es difícil de construir, pero es muy confiable y funciona con precisión debido a la retroalimentación". (p. 4)

2.4.2. Control proporcional

Según Kavianpour (2017), "La corrección es proporcional a la cantidad de error, la ganancia proporcional consiste en el producto entre la señal de error y la constante para disminuir el error en estado estacionario". (p. 4)

2.4.3. Control derivativo

Según Kavianpour (2017), "El control derivativo tiene en cuenta la velocidad con la que cambia el error. Cuanto más veloz aumenta el error, más corrección se aplica. Cuanto más lento es el aumento del error, menos corrección se aplica". (p. 4)

2.4.4. Control integral

Según Kavianpour (2017), "El control integral tiene en cuenta el tiempo del error antes de aplicar una corrección". (p. 5)

2.4.5. Algoritmo de control

Según Visaya Solutions (2021), "Es un tipo de lógica programada en un controlador para analizar el error entre un valor medido (MV) y un punto de ajuste (SP). Esta lógica ayuda al controlador a procesar los datos de campo y decidir qué hacer". (p. 1)

2.4.6. Internet de las Cosas (IoT)

Según Oracle (2019), describe la red de dispositivos que cuentan con sensores, software y tecnologías para conectarse e intercambiar datos con otros dispositivos o sistemas a través de Internet. Estos dispositivos pueden ser de casa o a nivel industrial. (p. 1)

2.4.7. Relé

Según Oxford (s.f.), "Es un componente electrónico que, impulsado por una corriente

eléctrica muy débil, abre o cierra un circuito en el cual fluye una mayor corriente". (p. 1)

2.5. Diseño de la investigación

2.5.1. Variables de investigación

a) Variable independiente: Control en cascada en Raspberry PI4 e IoT.

Dimensiones: Algoritmo de control en cascada, Panel de control local.

b) Variable dependiente: Monitoreo del nivel y flujo en un prototipo de tanque de riego.

Dimensiones: Monitoreo remoto, Implementación de prototipo de tanque de riego.

2.5.2. Tipo y método de investigación

En este proyecto, se desarrolló una investigación del tipo aplicada porque se buscó generar una nueva solución orientada al control de tanques de riego, específicamente en el campo de la agricultura inteligente; además, según Lozada (2014) "la investigación aplicada pretende encontrar la generación de nuevo conocimiento con aplicación eficaz a los problemas del sector" (p. 1).

El método de investigación aplicado en este trabajo fue del tipo experimental porque se manipuló más de una variable de estudio y se realizaron pruebas con el valor de las variables para poder observar su comportamiento entre sí, y desarrollar una solución eficiente; además, según Arias (2012) "la investigación experimental consiste en someter al objeto o grupo de objetos en varias condiciones o estímulos, para observar los resultados o reacciones que se originan (p. 34).

2.5.3. Técnicas e instrumentos de recolección de datos

Las técnicas de recolección de datos son diversas maneras o métodos de conseguir información. En esta investigación se utilizaron técnicas de recolección de datos para monitorear y gestionar el sistema. La técnica utilizada fue a través de la adquisición de datos y mediciones directas desde el sensor de nivel JSN-SR04T y el sensor de flujo YF-S201, dado que el sistema de control en cascada controló tales variables. Para realizar las mediciones de datos de los sensores se utilizó una programación en código Python para adquirir los datos de nivel y flujo en tiempo real. La adquisición de datos se realizó en la

plataforma ThingSpeak, ya que, mediante programación en el código, se integraron unas líneas de código de solicitudes http, para escribir los datos en la plataforma. A partir de estas técnicas de recolección de datos se logró monitorear y gestionar el sistema de control en cascada.

Por otro lado, los instrumentos de recolección de datos son los sensores usados para sensar el nivel y flujo del proceso, para ello se eligió el sensor de ultrasonido JSN-SR04T para obtener la variable nivel, el cual tiene un rango de medición de 25 – 450 cm, este sensor determina las distancias a través de pulsos de ultrasonido. Para medir la variable flujo se eligió el sensor de flujo YF-S201, el cual basa su funcionamiento en el efecto Hall, gracias a este principio se puede sensar el flujo o caudal dentro de las tuberías de agua. Estos instrumentos de recolección de datos fueron los más convenientes para realizar esta investigación. Además, fueron ubicados estratégicamente en el prototipo de tanque de riego, para poder controlar las variables de nivel y flujo con en el sistema de control en cascada.

2.5.4. Procedimiento para la recolección de datos

Para la recolección de datos, se procedió a trabajar de la siguiente manera:

a) Se evaluaron los distintos tipos de sensores para las variables de nivel y flujo y se optó por elegir, el sensor JSN-SR04T para la variable nivel y el sensor YF-S201 para la variable flujo. Estos sensores fueron convenientes a usar en el prototipo debido a que generan una señal continua y proporcionan datos continuos.

b) Se posicionó el sensor JSN-SR04T en la parte superior del tanque de riego, debido a que este genera una señal ultrasónica para determinar el nivel.

c) Se posicionó el sensor YF-S201 en la tubería horizontal de ingreso de agua en el tanque, para poder sensar el caudal de agua que ingresa para llenar el tanque.

d) Se programó el código necesario para recibir los datos de los sensores en Raspberry PI 4, con el lenguaje de programación Python.

e) Se integraron líneas de código de solicitudes http, para comunicar el prototipo de tanque de riego con la plataforma ThingSpeak.

f) Se realizó la visualización de los datos en la plataforma ThingSpeak, a través de cualquier dispositivo con conexión a internet.

CAPÍTULO III: DESARROLLO DEL PROYECTO

Este capítulo describe las etapas del desarrollo de la implementación de control en cascada en el Raspberry PI 4 para monitorear por internet de las cosas el nivel y flujo en un prototipo de tanque de riego.

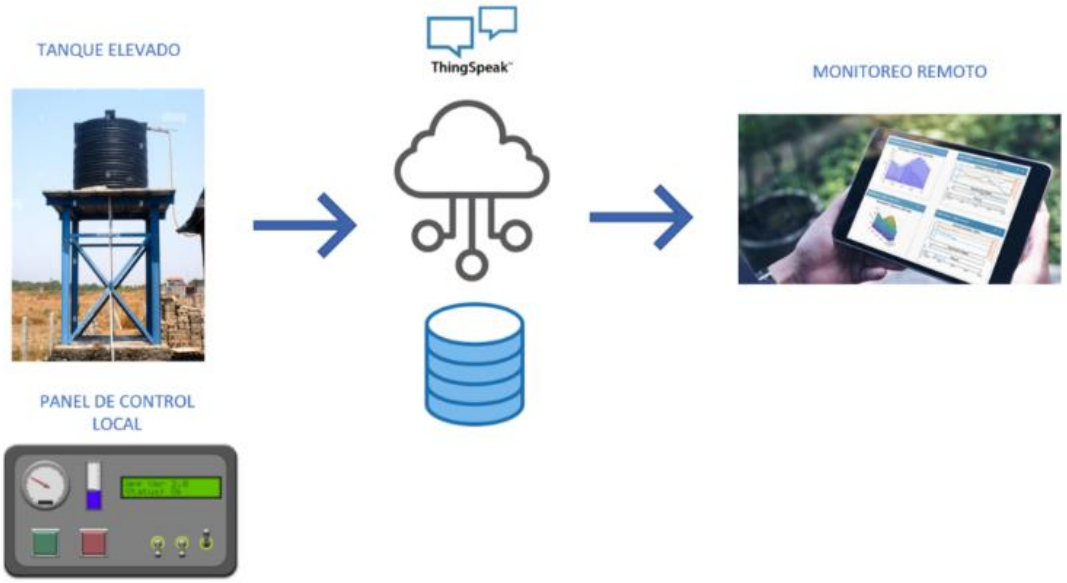
3.1. Diagrama general de la propuesta del proyecto

La propuesta del proyecto fue desarrollada en etapas, desde la implementación de tanque de riego hasta su comunicación con el software ThingSpeak y la programación de riegos. Es así que, en la primera etapa, sección 3.1, se muestran los diagramas necesarios que explican el funcionamiento del proyecto, en la sección 3.2, se detalla con la estructura del prototipo de tanque de riego, que comprende, la elección de los dispositivos de campo adecuados para el prototipo, el posicionamiento estratégico de los mismos, el diseño de algunas piezas para la funcionalidad del proyecto y la implementación del panel de control local. Luego, en la tercera etapa, sección 3.3, se detalla la estructura del algoritmo de control en cascada implementado en el Raspberry PI, considerando el algoritmo de control de los controladores P y PI para las variables nivel y flujo respectivamente; asimismo, en la cuarta etapa, sección 3.4, se describe la comunicación entre el prototipo de tanque de riego y ThingSpeak. Finalmente, en la quinta etapa del desarrollo del proyecto, sección 3.5, se detalla la programación de riegos programados dentro del algoritmo de control en el Raspberry PI 4.

Asimismo, en la figura 9, se muestra el diagrama pictográfico de la propuesta del proyecto, donde se observan los procesos involucrados tales como el control del prototipo de tanque de riego, donde se observa un tanque de agua elevado con sus tuberías de ingreso y salida de agua, además del panel de control local donde se encuentra la interfaz de control del sistema. Por último, se muestra la comunicación del prototipo con la plataforma IoT ThingSpeak.

Figura 9

Diagrama pictográfico del proyecto



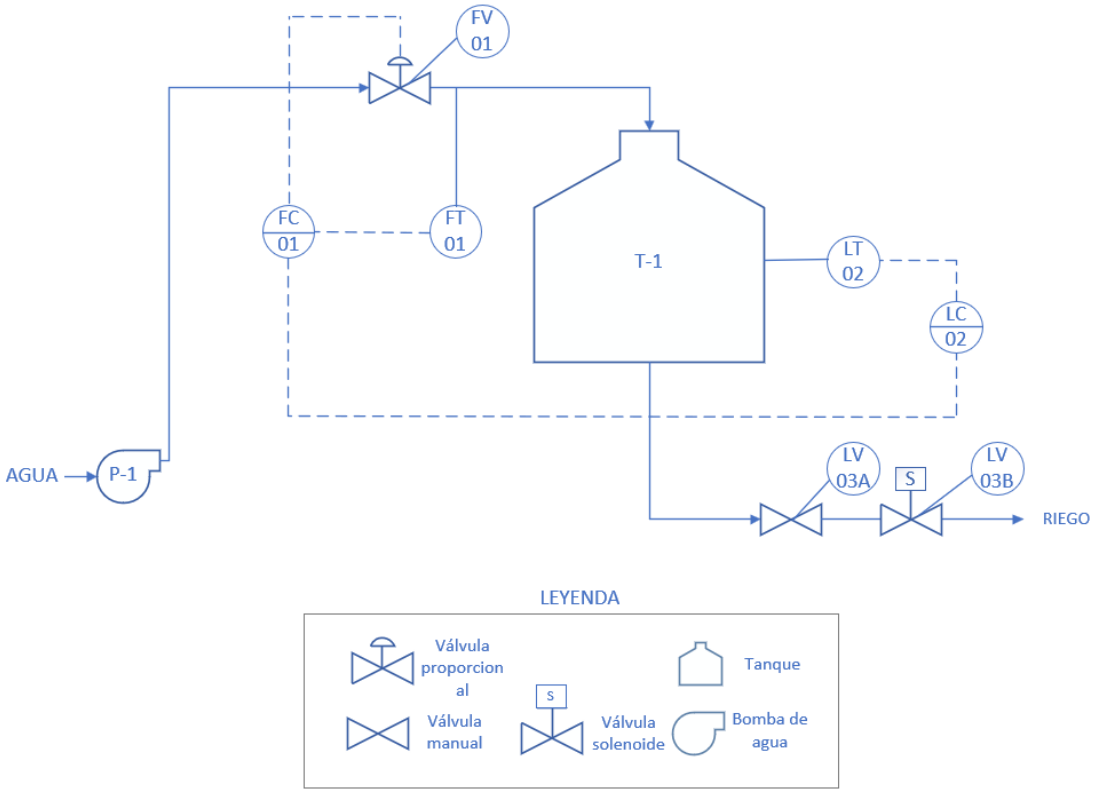
Nota. Elaboración Propia

A continuación, en el anexo F, se presenta el diagrama de bloques del control en cascada utilizado para controlar las variables de proceso nivel y flujo en el prototipo de tanque de riego a escala. Tal es así que, en dicho diagrama se observan los lazos de control primario y secundario, los cuales identifican a los procesos primario (control de nivel en el tanque de riego), y secundario (control de la apertura de la válvula). Además, se muestran los respectivos controladores y el disturbio que corresponde a los riegos programados.

Para un mejor panorama de la solución, en la figura 10 se muestra el diagrama de tuberías e instrumentación P&ID del proyecto propuesto, como especifica la norma ISA 5.1. Donde se observan los lazos de control en cascada, siendo el controlador primario el de nivel y el secundario el de flujo, así como también el sensor de flujo YF-S201 y el sensor de ultrasonido JSN-SR04T, una válvula proporcional como actuador del control en cascada, diseñada a partir de una válvula manual tipo mariposa, un servomotor MG996R y un molde en 3D, también como elemento final de control, una válvula solenoide FPD-270A, utilizada para realizar los riegos programados y por último una bomba de agua para suministrar agua al tanque.

Figura 10

Diagrama de Tuberías e Instrumentación P&ID del proyecto

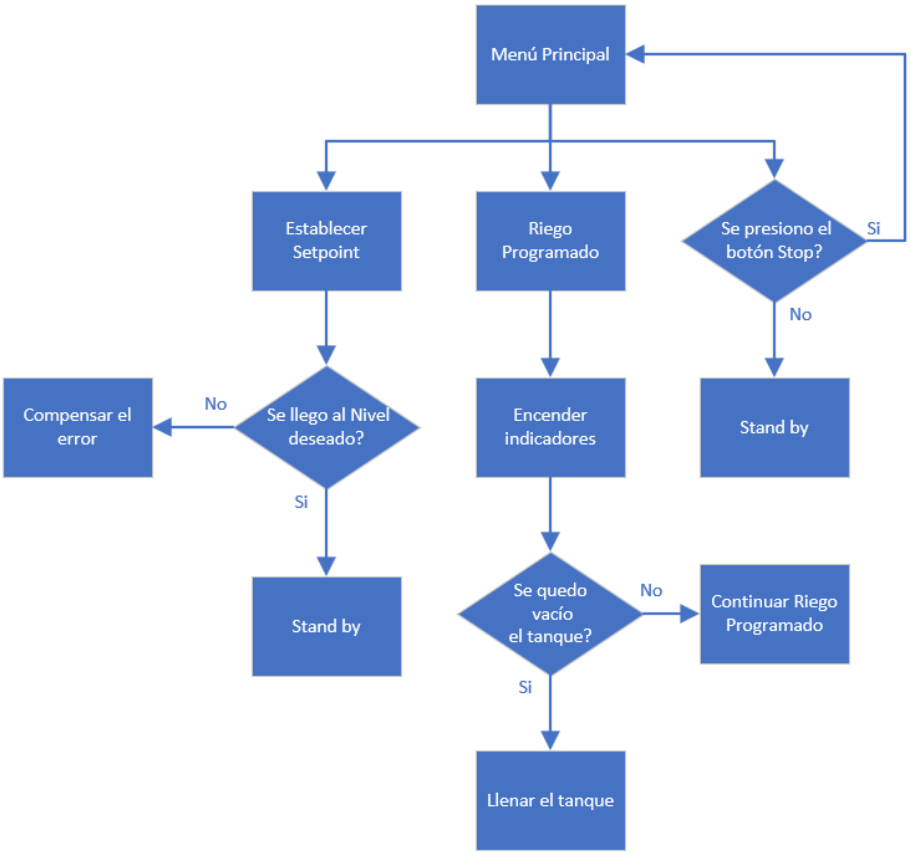


Nota. Diagrama P&ID representativo para el control del prototipo de tanque de riego. Elaboración Propia

A continuación, se muestra un flujograma en la figura 11, donde se puede observar la lógica de funcionamiento del prototipo de tanque de riego. El flujograma se explica de la siguiente manera. La interfaz de usuario del panel de control inicia con un menú principal que contiene con 2 opciones, las cuales son funciones. Establecer Setpoint y Riego Programado, cada una con sus respectivos flujos de trabajo. Adicional a los 2 procesos anteriores, se ejecuta un tercer proceso en paralelo, que tiene por funcionalidad, detener los procesos de establecer setpoint o de riegos programados. Para así volver al inicio, en el menú principal y ejecutar otra función. Según lo requiera el operador del prototipo de tanque de riego. Por lo cual, la interacción con el prototipo de tanque de riego fue posible gracias a la interfaz de usuario del panel de control local que se implementó, la cual se encuentra detallada en la sección 3.2.2.

Figura 11

Flujograma del proyecto

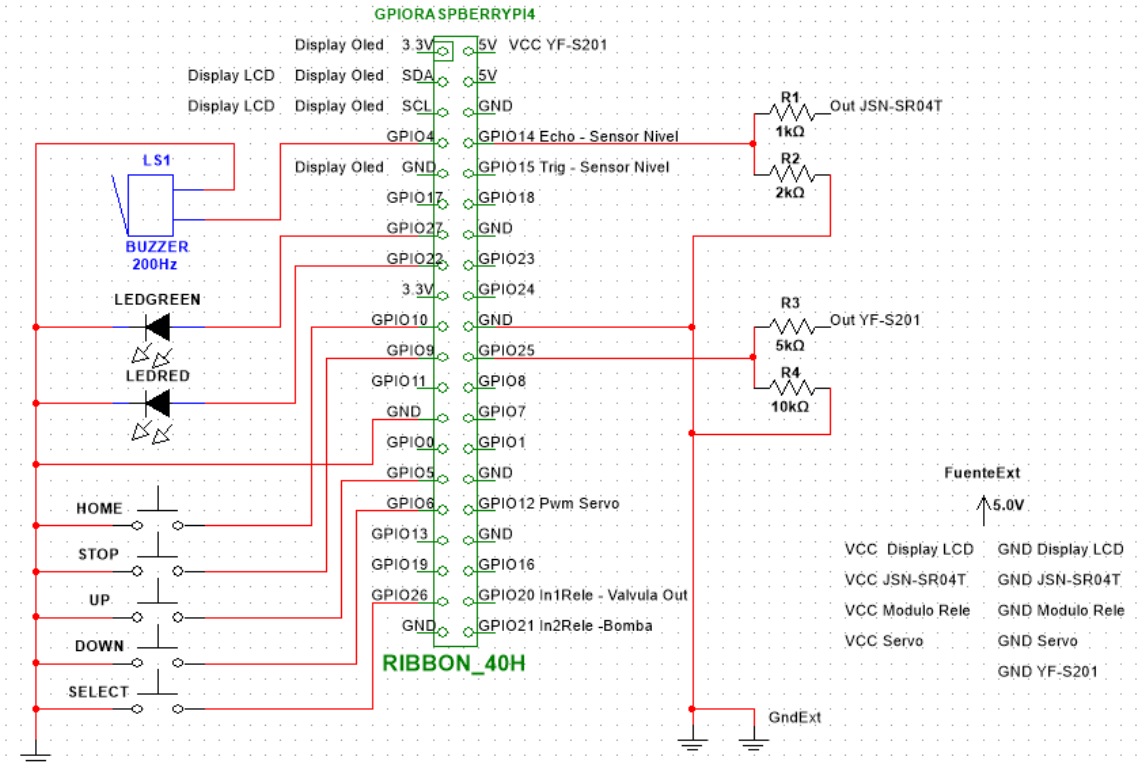


Nota. Elaboración Propia

Seguidamente, se muestra en la figura 12, el diagrama electrónico esquemático, donde se interconecto los GPIO de Raspberry con los diferentes módulos, sensores y actuadores, etc. del panel de control local, el diagrama fue sumamente necesario para implementar el control en cascada, además otorgar de interfaz de control para el usuario.

Figura 12

Diseño Electrónico Esquemático



Nota. Elaboración Propia

3.2. Implementación de la estructura del prototipo de tanque de riego

La implementación de la estructura del prototipo de tanque de riego comprendió los dispositivos de campo, tales como los sensores y actuadores, y el módulo de panel de control local, donde fue posible interactuar con el tanque y programar funciones. Este prototipo se implementó en dimensiones reducidas como se explicó en las limitaciones en la sección 1.4, utilizando un depósito de tanque de plástico de 40 centímetros de altura, con capacidad de 25 litros de agua, y tuberías de diámetro nominal de ½", tal como se muestra en la figura 13. Además, se consideró utilizar un segundo depósito de agua, con el fin de emular a un pozo o canal por donde se abastecen de agua las zonas de cultivos.

Figura 13

Prototipo de tanque de riego

Nota. Elaboración Propia

3.2.1. Elección y posicionamiento de los dispositivos de campo

El prototipo de tanque de riego es controlado por dispositivos de campo de bajo costo, que se pueden conseguir fácilmente en el mercado. Se seleccionaron como sensores el sensor de ultrasonido JSN-SR04T y el sensor de flujo de agua YF-S201, ambos con conexión de ½". Por otro lado, como actuadores, la válvula solenoide FDP-270A con conexión de ½", la válvula proporcional con conexión de ½" y por último la bomba de agua con conexión de ½". Estos dispositivos de campo tienen compatibilidad con muchas placas electrónicas de desarrollo, tal como la que cuenta la Raspberry PI 4. Además, son de gran ayuda por tener capacidad de control, precisión en las mediciones y un bajo consumo eléctrico.

Según las características físicas del tanque de agua se procedió a ubicar los dispositivos de campo de la siguiente manera:

El sensor de flujo de agua YF-S201, el cual posee un rango de flujo de 1-30L/min, se posicionó en la tubería horizontal de PVC de ½" a continuación de la válvula proporcional, tal como se observa en la figura 14, con la finalidad de sensar el flujo de agua que ingresa para llenar el tanque.

Figura 14

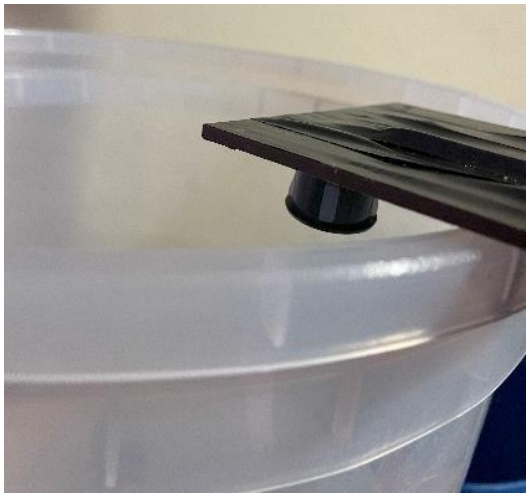
Ubicación del Sensor YF-S201

Nota. Elaboración Propia

El sensor de ultrasonido JSN-SR04T, el cual posee un rango de medición de 25-450 cm, rango suficiente para la medición del nivel de agua en el prototipo de tanque de riego. Este se posicionó en la parte superior del tanque de agua, tal como se muestra en la figura 15.

Figura 15

Ubicación del Sensor JSN-SR04T



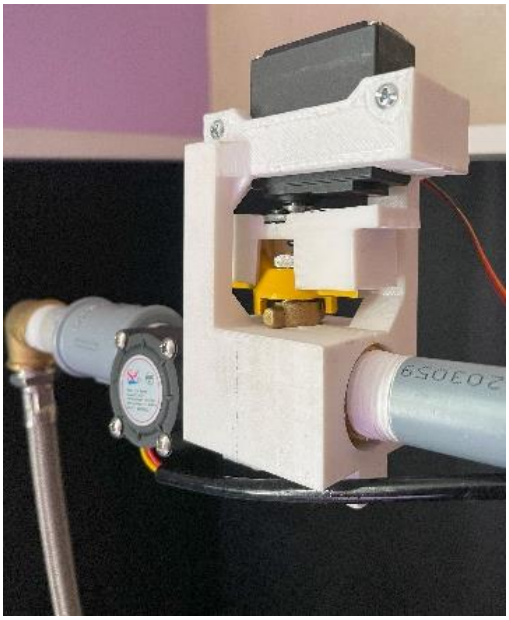
Nota. Elaboración Propia

En cuanto a los dispositivos actuadores, se utilizó una válvula proporcional, su elaboración es detallada en la sección 3.2.3. Esta se ubicó en la tubería horizontal de PVC de ½" y antes del sensor de flujo de agua YF-S201 donde por ella fluye el agua para llenar

el tanque, tal como se muestra en la figura 16.

Figura 16

Ubicación de la válvula proporcional

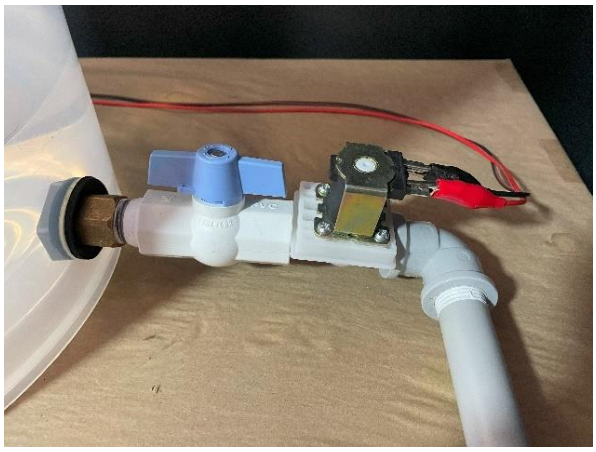


Nota. Elaboración Propia

También como dispositivo actuador, se eligió una válvula solenoide FDP-270A con conexión ½", la cual es de comportamiento ON/OFF, necesaria para el desfogue del agua del prototipo de tanque de riego, para cuando se inicien los riegos programados. Esta válvula se ubicó en la parte más baja del tanque de 25L, a continuación de una válvula manual, como se muestra en la figura 17.

Figura 17

Ubicación de la válvula solenoide FDP-270A



Nota. Elaboración Propia

Por último, se optó por contar con una bomba de agua de conexión ½” con un caudal de 800L/H, utilizada para suministrar agua al tanque de 25L. La bomba se ubicó en la parte más baja del segundo tanque de agua, que emula un pozo o canal de abastecimiento de agua de las zonas de cultivos. Se conectó a la bomba una tubería vertical de PVC de ½” donde por la cual fluye el agua para llenar el tanque de 25L, tal como se muestra en la figura 18.

Figura 18

Ubicación de la bomba de agua

Nota. Elaboración Propia

3.2.2. Implementación del panel de control local

La implementación del panel de control local conllevó a elegir los componentes electrónicos adecuados para poder conseguir una interfaz amigable con el usuario, en este proyecto se optó por utilizar componentes de bajo costo, pero a la vez fiables.

Los componentes que comprenden el panel de control local, son:

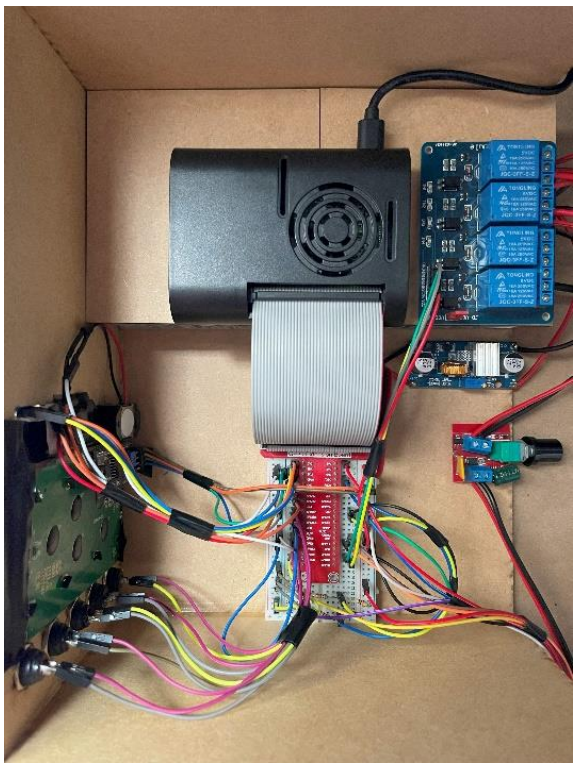
- (01) Raspberry PI 4
- (01) Display LCD 4x20
- (01) Display OLED 0.96”
- (01) Buzzer
- (02) Diodos LED

- (05) Push buttons
- (01) Fuente switching de 12V 10A
- (01) Módulo Step Down DC-DC 12 a 5VDC
- (01) Módulo relé de 4 canales
- (01) Controlador de velocidad PWM de 5A

Los componentes anteriormente mencionados fueron instalados dentro de un case, el cual está detallado en la sección 3.2.4. Además, se puede visualizar como quedan instalados dentro del case los componentes del panel de control local en la figura 19.

Figura 19

Instalación del panel de control local



Nota. Elaboración Propia

El suministro de energía al panel de control local y a todos los dispositivos de campo se otorga desde la fuente switching de 12VDC 10A que fue adquirida para este proyecto. Ver la figura 20.

Figura 20

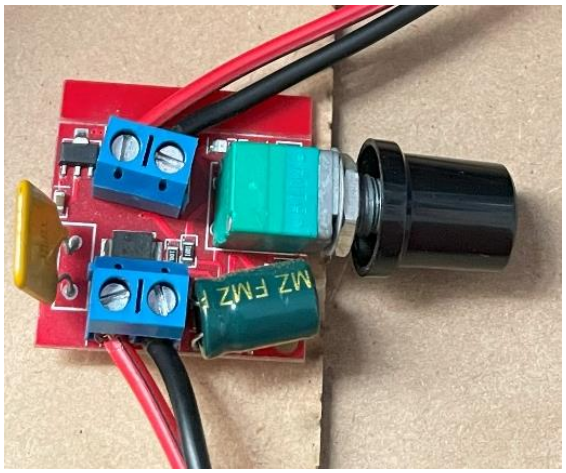
Fuente switching de 12VDC

Nota. Elaboración Propia

Dicha fuente posee 2 canales de 12VDC. El primer canal de la fuente fue utilizado para alimentar la válvula solenoide FDP-270A y su vez para conectar la bomba de agua, a través de un controlador de velocidad PWM de 5A, como se muestra en la figura 21, con el fin regular del flujo de agua de ingreso al tanque.

Figura 21

Controlador de velocidad PWM

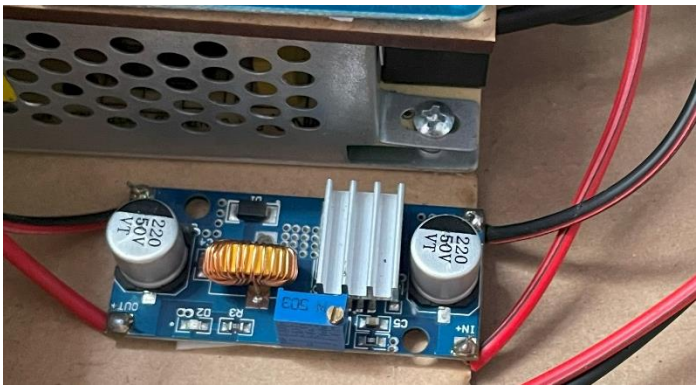


Nota. Elaboración Propia

En el otro canal libre de la fuente switching de 12V 10A se conectó en paralelo un módulo Step Down DC-DC 12 a 5VDC tal como se muestra en la figura 22, con el fin de reducir el voltaje para alimentar los demás dispositivos que operan a 5VDC.

Figura 22

Módulo Step Down DC-DC 12 a 5VDC

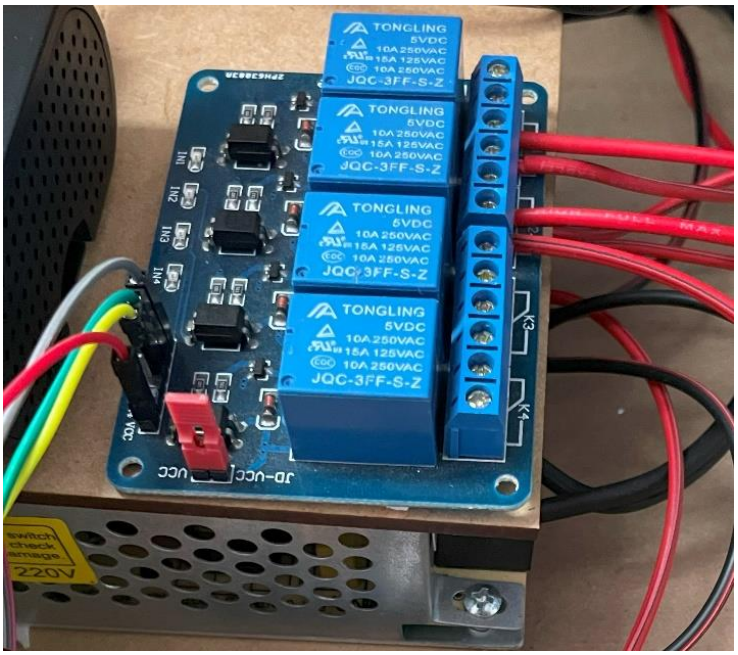


Nota. Elaboración Propia

Estos actuadores fueron conectados a un driver de potencia como se muestra en la figura 23, módulo relé de 4 canales, con el fin de poder controlarlos con el Raspberry PI 4 mediante el algoritmo de control.

Figura 23

Módulo relé de 4 canales



Nota. Elaboración Propia

La interacción con el panel de control local fue posible gracias a los push buttons y el display LCD 4x20, tal como se muestra en la figura 24, donde el usuario pudo interactuar y visualizar el menú inicial, el cual contiene las opciones de establecimiento del setpoint e inicialización de los riegos programados. El display OLED fue ubicado en la parte frontal del panel de control local, donde el usuario logró visualizar el monitoreo local de las variables de control. Por último, las alarmas visuales y sonoras, diodos led y buzzer, fueron activados según el estado en que se encontraba el proceso.

Figura 24

Panel de control local – Vista frontal

Nota. Elaboración Propia

Adicionalmente, la implementación de un menú de opciones fue posible con la programación de un código en lenguaje Python, utilizando las librerías adecuadas para el control de la pantalla LCD 4x20 y configurando los pulsadores como entradas en los GPIO de Raspberry PI 4. En la figura 25 se muestran las librerías utilizadas, la definición y configuración de los pines de los pulsadores.

Figura 25

Inicialización del menú interactivo

```
from RPLCD.i2c import CharLCD
import RPi.GPIO as GPIO
import time
# Definición de los pines de los pulsadores
UP_PIN = 5
DOWN_PIN = 6
SELECT_PIN = 26
HOME_PIN = 10
STOP_PIN = 9
PUMP_PIN  = 21
VALVE_PIN =20
# Configuración de los pines de los pulsadores
GPIO.setmode(GPIO.BCM)
GPIO.setup(UP_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(DOWN_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(SELECT_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(HOME_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(STOP_PIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

Nota. Elaboración Propia

En la figura 26 se muestra la declaración de la variable “menu_items”, el cual contiene una lista el menú y submenús necesarios para la interacción del panel de control local, esta sentencia es conocida como diccionario en Python.

Figura 26

Lista de opciones del menú interactivo

```
menu_items = [
    {
        "label": "Establecer SetPoint",
        "sub_menu": [
            {
                "label": "Setpoint 50%",
                "action": lambda: control_cascada_50()
            },
            {
                "label": "Setpoint 80%",
                "action": lambda: control_cascada_80()
            },
            {
                "label": "Setpoint 100%",
                "action": lambda: control_cascada_100()
            }
        ]
    },
    {
        "label": "Riegos Programados",
        "sub_menu": [
```

Nota. Elaboración Propia

Por otro lado, en la figura 27 se muestra la función “update_lcd”, la cual actualiza la pantalla LCD. Esta función borra la pantalla LCD y muestra los elementos del menú actualmente seleccionados, permitiendo la interacción con el menú de opciones.

Figura 27

Función Update LCD

```
def update_lcd():
    lcd.clear()
    # Escribir el título del menú
    lcd.write_string("   Tanque de riego")
    lcd.crlf()
    if current_menu is menu_items:
        # Menú principal
        for i, item in enumerate(current_menu):
            if i == selected_item:
                lcd.write_string(">" + item["label"])
            else:
                lcd.write_string(" " + item["label"])
            lcd.crlf()
    else:
        # Submenú
        for i, item in enumerate(current_menu):
            if i == selected_item:
                lcd.write_string(">" + item["label"])
            else:
                lcd.write_string(" " + item["label"])
            lcd.crlf()
```

Nota. Elaboración Propia

Luego, en la figura 28, se muestra la función “handle_buttons”, la cual define el funcionamiento de los botones para la interacción con el menú.

Figura 28

Función Handle Buttons

```
def handle_buttons(channel):
    global selected_item
    global current_menu
    if channel == SELECT_PIN:
        item = current_menu[selected_item]
        if "sub_menu" in item:
            current_menu = item["sub_menu"]
            selected_item = 0
        elif "action" in item:
            item["action"]()
    elif channel == HOME_PIN:
        current_menu = menu_items
        selected_item = 0
    elif channel == STOP_PIN:
        print("Reebot")
    elif channel == UP_PIN:
        selected_item = (selected_item - 1) % len(current_menu)
    elif channel == DOWN_PIN:
        selected_item = (selected_item + 1) % len(current_menu)
```

Nota. Elaboración Propia

Finalmente, en la figura 29, se muestra la configuración de interrupción en los GPIO, donde se agrega un tiempo de 200ms para mejorar la lectura de entrada cuando se presionan los push buttons, con la finalidad de evitar falsos pulsos.

Figura 29

Configuración de interrupción en los GPIO

```
# Configurar las interrupciones para los pulsadores
GPIO.add_event_detect(SELECT_PIN, GPIO.FALLING, callback=handle_buttons, bouncetime=200)
GPIO.add_event_detect(HOME_PIN, GPIO.FALLING, callback=handle_buttons, bouncetime=200)
GPIO.add_event_detect(STOP_PIN, GPIO.FALLING, callback=handle_buttons, bouncetime=200)
GPIO.add_event_detect(UP_PIN, GPIO.FALLING, callback=handle_buttons, bouncetime=200)
GPIO.add_event_detect(DOWN_PIN, GPIO.FALLING, callback=handle_buttons, bouncetime=200)
```

Nota. Elaboración Propia

3.2.3. Diseño de la válvula proporcional

Para realizar el control en cascada en el presente proyecto se necesitó de una válvula proporcional de conexión ½". La solución fue diseñar una válvula proporcional a partir de una válvula manual convencional, un servomotor MG996R y un molde en 3D el cual se utilizó para adaptar el servomotor MG996R a la válvula manual y así poder conseguir una válvula proporcional, ya que en el mercado no se encuentra una válvula proporcional que se pueda controlar con los GPIO del Raspberry PI.

Como se mencionó antes se eligió el servomotor MG996R, ya que posee engranajes de metal, estos lo dotan de durabilidad a comparación de los servomotores con engranajes de plástico; además, de ofrecer un mejor torque, necesario para poder girar la válvula manual convencional. Se utilizó además una válvula con llave tipo mariposa tal como se muestra en la figura 30.

Figura 30

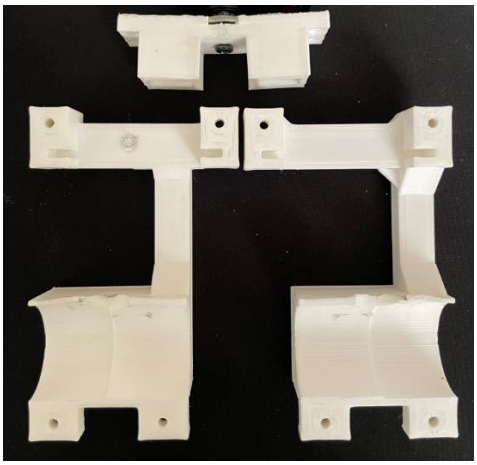
Válvula manual tipo mariposa

Nota. Elaboración Propia

Seguidamente se diseñó e imprimió en 3D el molde para poder adaptar la válvula manual al servomotor, el diseño de las piezas en 3D se muestran en la figura 31.

Figura 31

Molde en 3D

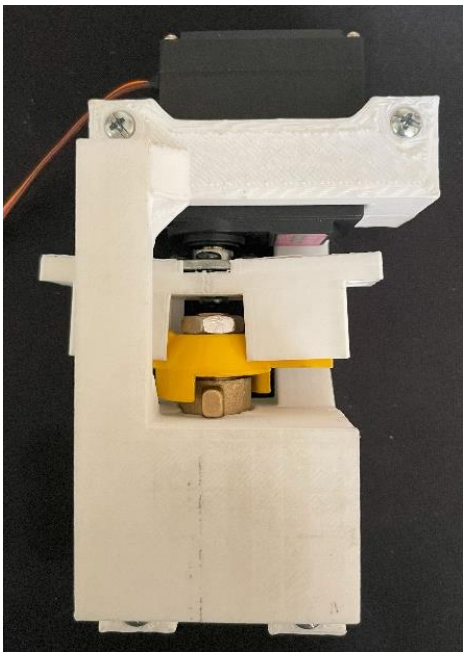


Nota. Elaboración Propia

Por último, se unieron los 3 componentes: válvula manual, servomotor y el molde en 3D. El resultado final de la válvula proporcional se observa en la figura 32.

Figura 32

Válvula proporcional



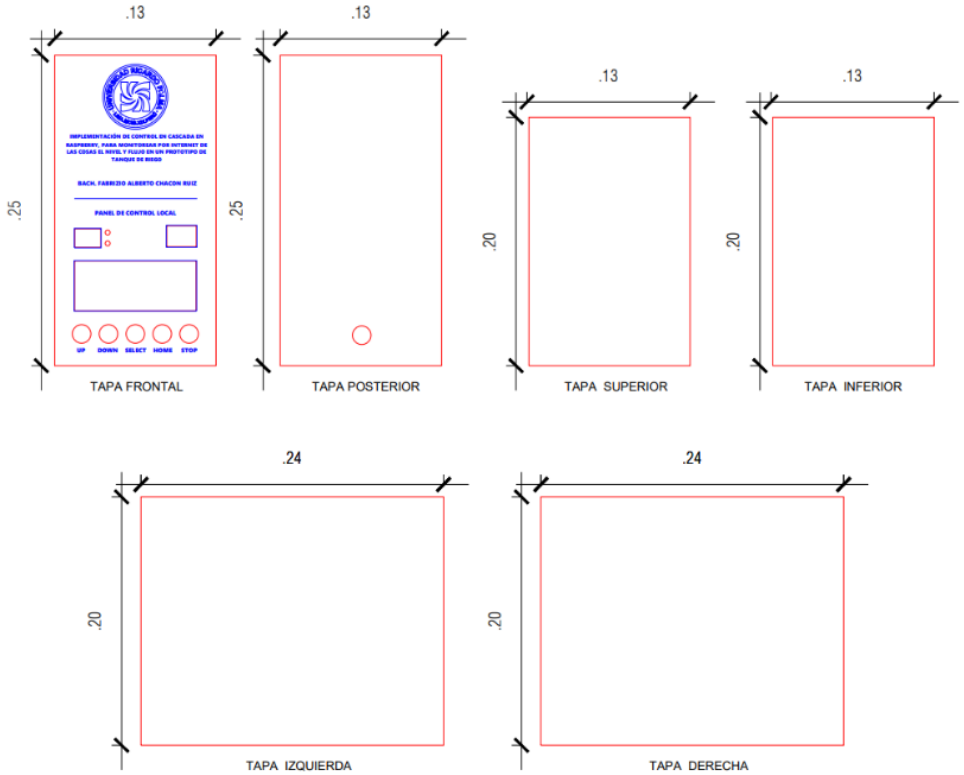
Nota. Elaboración Propia

3.2.4. Diseño del case del panel de control local

Se diseñó la estructura del panel de control local en el software AutoCAD, definiendo las medidas de los componentes instalados en la parte frontal del case, el cual mide (0.25mx0.13m). Asimismo, se dibujaron los otros componentes del case como la tapa posterior, superior, inferior, izquierda y derecha como se puede apreciar en la figura 33. Todas las piezas fueron cortadas y grabadas a láser en material MDF de 3mm.

Figura 33

Diseño del case en AutoCAD



Nota. Elaboración Propia

3.3. Desarrollo del algoritmo de control en cascada en Raspberry PI 4

Para desarrollar la programación del proyecto de una manera más sencilla, escalable y optimizada, se estructuró el código fuente del proyecto modularizando sus funcionalidades. Esto implicó dividir el código fuente en 3 scripts, con la finalidad de dividir las tareas o procesos del código total y así evitar posibles bugs o errores en el procesamiento de tareas.

Las tareas de los scripts son las siguientes:

Script 1: Ejecutar el menú interactivo y utilizar funciones de los Script 2 y 3.
Script 2: Ejecutar el algoritmo de control en cascada para establecer un setpoint.
Script 3: Ejecutar el control secuencial para programar riegos.

3.3.1. Estructura del código fuente

El Script 1, el cual es el script principal tiene la funcionalidad de ejecutar el código de interacción con el menú principal, el cual se detalló en la sección 3.2.2. Así como también, su finalidad fue importar funciones a utilizar de los Script 2 y 3, como se muestra en la figura 34. De esta manera se consiguió ejecutar las funciones que requerían establecer el operador del prototipo de tanque de riego en determinado momento, sin que haya errores en el procesamiento o problemas con los bucles de las funciones.

Figura 34

Importación de funciones de los script

```
# Importar las funciones de los script 2 y 3
from script2 import control_cascada_50
from script2 import control_cascada_80
from script2 import control_cascada_100
from script3 import riego_programado_1
from script3 import riego_programado_2
from script3 import riego_programado_3
```

Nota. Elaboración Propia

Además, se importaron librerías de procesamiento en paralelo, esto con el fin de ejecutar dos o más procesos a la vez, como, por ejemplo, detener una de las opciones del menú y ejecutar nuevamente el código principal, el cual esta implementado; también es usado como medida de seguridad, si en algún caso algún dispositivo de campo deje de funcionar.

El segundo script tiene la funcionalidad de ejecutar el algoritmo de control en cascada, el cual es detallado en la sección 3.3.2. Este script contiene 3 funciones a la vez, las cuales se pueden llamar dentro del submenú de la opción “Establecer Setpoint” del menú principal. Estas son:

Control_Cascada_50: Establece el Setpoint de nivel del tanque en 50% de capacidad.
Control_Cascada_80: Establece el Setpoint de nivel del tanque en 80% de capacidad.
Control_Cascada_100: Establece el Setpoint de nivel del tanque en 100% de capacidad.

El tercer script tiene la funcionalidad de ejecutar el control secuencial para programar tareas automatizadas de riego, el cual es detallado en la sección 3.5. Este script contiene 3 funciones, las cuales se pueden llamar dentro del submenú de la opción “Riegos Programado” del menú principal. Por lo cual, estas 3 funciones son:

Riego_Programado_1: Establece el Riego Programado 1 dentro de un horario especifico.
Riego_Programado_2: Establece el Riego Programado 2 dentro de un horario especifico.
Riego_Programado_3: Establece el Riego Programado 3 dentro de un horario especifico.

3.3.2 Desarrollo del algoritmo de control en cascada

El desarrollo del algoritmo de control en cascada se implementó en el Script 2 y se estructuró tal como se observa en el código de programación de la figura 35: librerías utilizadas, definición y configuración de las entradas y salidas.

Figura 35

Inicialización del control en cascada

```
import RPi.GPIO as GPIO
import time
import requests
#Definición de los pines de los sensores actuadores e indicadores
TRIG_PIN = 15
ECHO_PIN = 14
FLOW_PIN = 25
SERVO_PIN = 12
PUMP_PIN  = 21
VALVE_PIN = 20
GREEN_LED_PIN = 27
RED_LED_PIN   = 22
BUZZER_PIN    = 4
# Configuración de los pines de los pulsadores
GPIO.setmode(GPIO.BCM)
#Configuración de los pines de los sensores y actuadores
GPIO.setup(TRIG_PIN, GPIO.OUT)
GPIO.setup(ECHO_PIN, GPIO.IN)
GPIO.setup(FLOW_PIN, GPIO.IN)
GPIO.setup(SERVO_PIN, GPIO.OUT)
GPIO.setup(PUMP_PIN, GPIO.OUT)
GPIO.setup(VALVE_PIN, GPIO.OUT)
GPIO.setup(GREEN_LED_PIN, GPIO.OUT)
GPIO.setup(BUZZER_PIN, GPIO.OUT)
```

Nota. Elaboración Propia

El controlador en cascada comprende, un controlador principal, el cual es el controlador de nivel y un controlador secundario, el cual es el controlador de flujo. Para el controlador de nivel se utilizó un controlador tipo P y para el controlador de flujo se utilizó un controlador tipo PI ambos controladores se representaron en ecuaciones que determinaron el funcionamiento del control en cascada, tal como se muestra en la figura 36.

Figura 36

Función Control en Cascada

```
# Función para el control en cascada
def control_cascada():
    global nivel_actual, nivel_anterior, flujo_actual, flujo_anterior
    nivel_actual = obtener_nivel()
    flujo_actual = obtener_flujo()
    print (nivel_actual)
    # Controlador principal (nivel)
    error_nivel = nivel_deseado - nivel_actual
    control_nivel = Kp_nivel * error_nivel
    # Controlador secundario (flujo)
    error_flujo = flujo_deseado - flujo_actual
    control_flujo = Kp_flujo * error_flujo + Ki_flujo * (error_flujo - (flujo_actual - flujo_anterior))
    # Control final
    control_final = control_nivel + control_flujo
    # Aplicar control al servomotor
    controlar_servomotor(control_final)
```

Nota. Elaboración Propia

Así como también, se definió la función de acción de control del controlador en cascada, como se muestra en la figura 37, la cual es ejecutada por el servomotor MG996R. La acción de control del servomotor es el cambio del ciclo de trabajo del mismo, el cual fue de 0 a 10. Por lo cual, la fórmula matemática tuvo que ajustarse de tal manera que siempre otorgue un valor de 0 a 10. Previamente se tuvo también que incluir la fórmula matemática: nivel = 100 – ((distancia-10) / (36-10)) *100, en la función obtener nivel, para obtener un valor de nivel del 0 al 100% de capacidad.

Figura 37

Función de acción de control del servomotor

```
# Función para controlar el servomotor
def controlar_servomotor(nivel):
    duty_cycle = nivel * 0.05 + 5
    GPIO.output(SERVO_PIN, True)
    pwm = GPIO.PWM(SERVO_PIN, 50)
    pwm.start(duty_cycle)
    time.sleep(1)
    GPIO.output(SERVO_PIN, False)
    pwm.stop()
```

Nota. Elaboración Propia

3.4. Comunicación entre el prototipo de tanque de riego y ThingSpeak

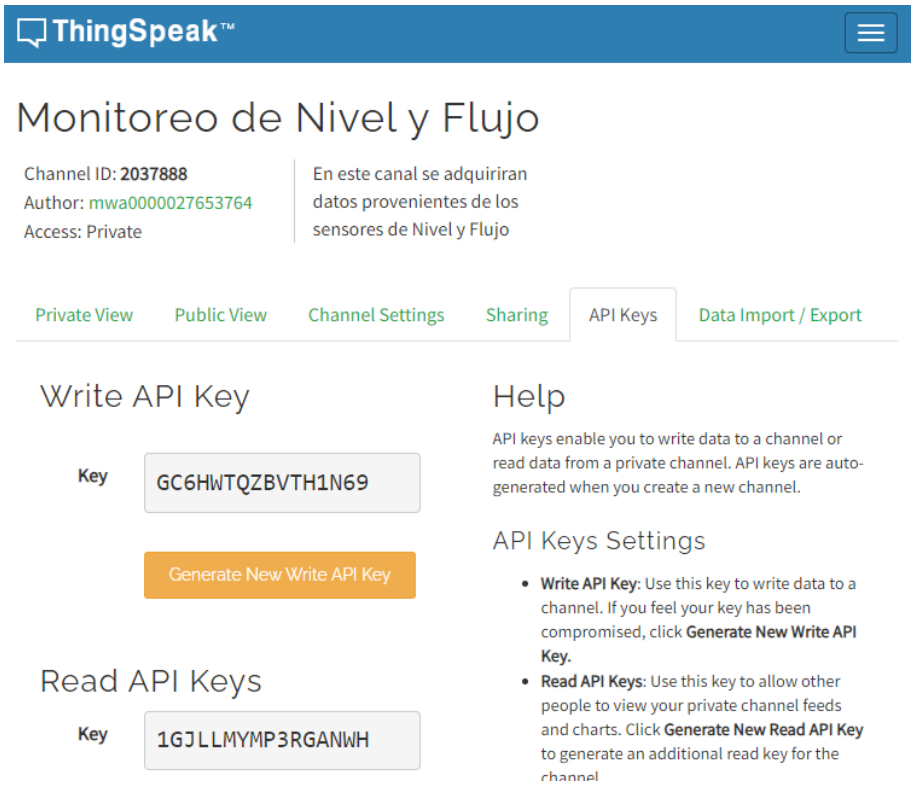
En este punto se describe la obtención y monitoreo de las variables de nivel y flujo del prototipo de tanque de riego por medio de programación y principalmente la utilización de la librería “requests”, para poder escribir en un canal de la plataforma ThingSpeak los datos de tales variables en tiempo real.

3.4.1. Envío de datos de los sensores hacia la plataforma ThingSpeak

La plataforma ThingSpeak ofrece la posibilidad para escribir y leer datos provenientes de cualquier dispositivo que esté conectado a internet. En este caso se usó para escribir datos provenientes de los sensores de nivel y flujo para así monitorear remotamente sus valores. En la figura 38, se muestra los API Key del canal usado en este proyecto.

Figura 38

API Key del canal en ThingSpeak



Nota. Elaboración Propia

Luego de conocer el código API Key para escribir datos, se procedió a definir la función Obtener Nivel la cual se muestra en la figura 39, donde se añade la sentencia "request.get", la cual genera una solicitud http que a su vez contiene la API Key del canal usado en el presente proyecto.

Figura 39

Función Obtener Nivel

```
def obtener_nivel():

    GPIO.output(TRIG_PIN, True)
    time.sleep(0.00001)
    GPIO.output(TRIG_PIN, False)

    while GPIO.input(ECHO_PIN) == 0:
        pulso_inicio = time.time()

    while GPIO.input(ECHO_PIN) == 1:
        pulso_fin = time.time()

    duracion = pulso_fin - pulso_inicio
    distancia = int(duracion * 17150)
    nivel = int(100 - ((distancia - 10) / (36 - 10)) * 100)
    enviar = requests.get("https://api.thingspeak.com/update?api_key=GC6HWTQZBVTH1N69&field1="+str(nivel))
    time.sleep(2)
    return nivel
```

Nota. Elaboración Propia

Asimismo, se definió la función Obtener Flujo, la cual se muestra en la figura 40, donde también se añade la sentencia “request.get”, la cual genera una solicitud http que a su vez contiene la misma API Key del canal usado en el presente proyecto.

Figura 40

Función Obtener Flujo

```
def obtener_flujo():

    def countPulse(channel):
        global count
        if start_counter == 1:
            count = count + 1
        GPIO.add_event_detect(FLOW_PIN, GPIO.FALLING, callback=countPulse)
    try:
        while True:
            start_counter = 1
            time.sleep(1)
            start_counter = 0
            flujo = (count / 7.5)
            count = 0
            enviar = requests.get("https://api.thingspeak.com/update?api_key=GC6HWTQZBVTH1N69&field2="+str(flujo))
            time.sleep(1)
    except KeyboardInterrupt:
        print('\nKeyboard interrupt!')

    return flujo
```

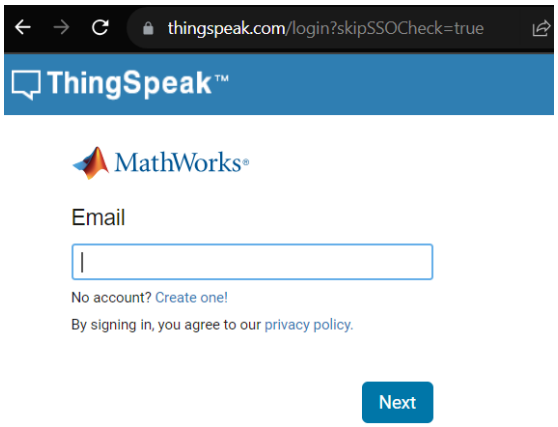
Nota. Elaboración Propia

3.4.2. Visualización de las variables de control a través de ThingSpeak

Para visualizar las variables de nivel y flujo del prototipo de tanque de riego, el usuario debe ingresar a la plataforma ThingSpeak a través de la url: https://thingspeak.com, como se muestra en la figura 41.

Figura 41

Acceso a ThingSpeak

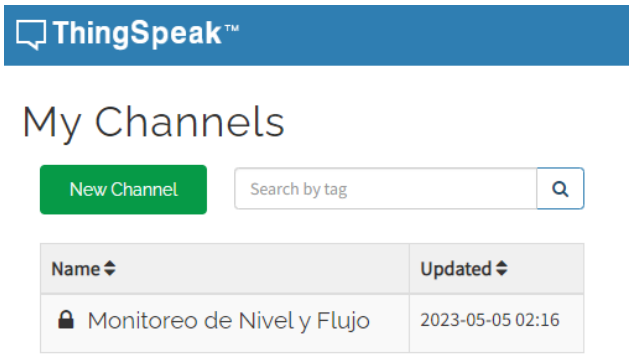


Nota. Elaboración Propia

Luego de ingresar a la plataforma se ubican los canales creados, en este caso se creó un canal denominado “Monitoreo de Nivel y Flujo”, como se muestra en la figura 42, con la finalidad de ahí visualizar el monitoreo remoto.

Figura 42

Canal de monitoreo remoto de nivel y flujo



Nota. Elaboración Propia

3.5. Control secuencial para los riegos programados

El control secuencial se implementó con un algoritmo de control basado en el control de tiempo, se utilizaron las librerías adecuadas para su correcto funcionamiento. Es así que, la válvula solenoide FPD-270A, la cual es la válvula de desfogue del prototipo de tanque de riego, fue el elemento principal a controlar de este proceso. Por esto, el operador tiene la posibilidad de establecer un riego programado, el cual se ejecutará siempre y tiene la función de aperturar la válvula solenoide a determinadas horas del día, según el requerimiento de riego establecido.

3.5.1. Desarrollo del algoritmo de control secuencial

El algoritmo de control secuencial se implementó en el Script 3 y se estructuró de la siguiente manera. Se definieron los GPIO a utilizarse, y se creó la función Riego Programado, tal como se muestra en la figura 43, donde se da la opción de programar la hora y minuto de apertura y cierre de la válvula solenoide.

Figura 43

Función Riego Programado

```
def riego_programado_1():
        print("Riego Programado 1")
        # Horario de apertura y cierre de la válvula
        hora_apertura = 16
        minuto_apertura = 44
        hora_cierre = 16
        minuto_cierre = 45
        # Función para los indicadores
        def encender_componentes():
            GPIO.output(GREEN_LED_PIN, GPIO.HIGH)
            GPIO.output(VALVE_PIN, GPIO.LOW)
            GPIO.output(BUZZER_PIN, GPIO.HIGH)
            time.sleep(2)
            GPIO.output(BUZZER_PIN, GPIO.LOW)
        def apagar_componentes():
            GPIO.output(GREEN_LED_PIN, GPIO.LOW)
            GPIO.output(VALVE_PIN, GPIO.HIGH)
            GPIO.output(BUZZER_PIN, GPIO.HIGH)
            time.sleep(2)
            GPIO.output(BUZZER_PIN, GPIO.LOW)
        # Ciclo principal para verificar el horario y controlar la válvula
        while True:
            hora_actual = time.localtime().tm_hour
            minuto_actual = time.localtime().tm_min
```

Nota. Elaboración Propia

Por lo tanto, cada vez que se apertura o cierra la válvula en un riego programado, se pone en ejecución la función de encender o apagar componentes, para obtener una alarma visual o auditiva. También se pone como condición de que cuando el nivel del agua en el prototipo de tanque se encuentre por debajo de 10% de capacidad, el tanque vuelva a llenarse hasta llegar al 100%, con la finalidad que durante el riego nunca se quede vacío.

CAPÍTULO IV: PRUEBAS Y RESULTADOS

Finalmente en este último capítulo, después de haber documentado la implementación total del prototipo de tanque de riego, se pone en evidencia, el funcionamiento de cada componente del proyecto, partiendo de la fácil interacción de la interfaz de usuario del panel de control local, para asignar las funciones del prototipo de tanque de riego, así como también, las pruebas y resultados obtenidos del funcionamiento individual de los sensores, actuadores y el funcionamiento en conjunto de los dispositivos utilizados para realizar el control en cascada y los riegos programados en el prototipo de tanque de riego. Además, se muestra evidencia del monitoreo remoto de las variables de control del tanque de riego, nivel y flujo, a través de la plataforma ThingSpeak.

4.1. Funcionamiento del prototipo de tanque de riego

En esta sección se pone en evidencia el correcto funcionamiento de los dispositivos utilizados en el prototipo de tanque de riego.

El panel de control local, el cual fue detallado en la sección 3.2.2, posee una interfaz de usuario amigable para el operador. Por ello, en la figura 44, se puede observar en la pantalla LCD 4x20 el menú principal con las opciones principales.

Figura 44

Menú principal de interfaz de usuario

Nota. Elaboración Propia

Desplazándose por el menú principal con los botones “UP” y “DOWN”, en caso se seleccione alguna opción del menú principal con el botón “SELECT” se dirigirá al submenú de cada opción, como se muestra en la figura 45. En dicho submenú se encuentran las funciones a ejecutar de los setpoint y riegos programados preestablecidos por el Script 1. De igual manera el botón “HOME” del panel de control local, funciona para volver al menú principal. El último botón llamado “STOP”, tiene la funcionalidad de detener el proceso que se esté llevando a cabo.

Figura 45

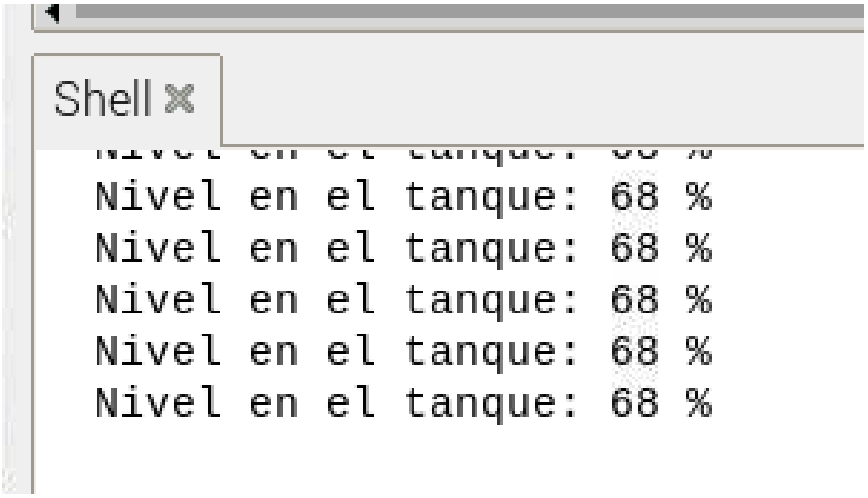
Submenú – Funciones preestablecidas

Nota. Elaboración Propia

El sensor de nivel JSN-SR04T, proporcionó los siguientes resultados como se muestra en la figura 46, sensando la distancia entre el sensor y la superficie del agua, estos datos no los mostró en centímetros, como normalmente se usa, sino que tuvo que ajustarse la fórmula matemática con respecto a las características físicas del tanque de riego de 25L, como se indicó en la sección 3.3.2, logrando así mostrar los datos en un rango de 0 a 100% de capacidad del tanque, esto con el fin de visualizar el estado del llenado del mismo.

Figura 46

Lecturas del sensor de nivel

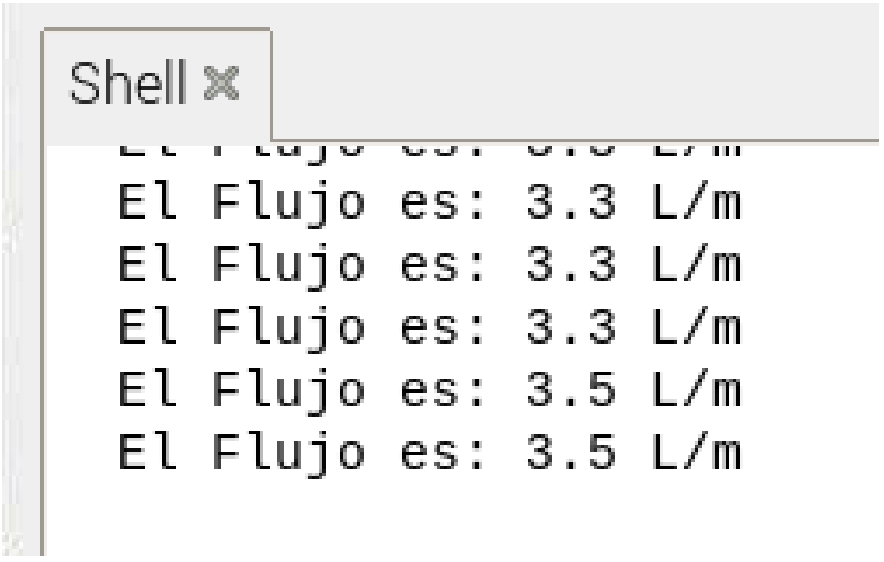


Nota. Elaboración Propia

Por otro lado, el sensor de flujo YF-S201, proporciono los siguientes resultados como se muestra en la figura 47, sensando el flujo en litros por minuto (L/m) que pasa a través de la tubería de ingreso de agua al tanque de riego de 25L.

Figura 47

Lecturas del sensor de flujo



Nota. Elaboración Propia

4.2. Pruebas del algoritmo de control en cascada

Dentro del algoritmo de control en cascada, se contienen dos controladores, uno de nivel y el otro de flujo. Se eligió llevar a cabo la sintonización de controladores por ajuste manual, también conocida como tanteo, el cual es un método para ajustar los parámetros de un controlador. Es así que, se sintonizaron los parámetros Kp y Ki para el controlador de flujo de tipo PI y el parámetro Kp para el controlador de nivel de tipo P. Este procedimiento tuvo como objetivo, mejorar la respuesta del sistema controlado en términos de velocidad y estabilidad. Seguidamente se procedió a realizar pruebas de estabilización del setpoint en 100% en el tanque, considerando una perturbación continua de un riego programado. Los controladores se sintonizaron con diferentes valores en sus parámetros, con la finalidad de encontrar los parámetros ideales de sintonización.

En la tabla 8, se pueden evidenciar los resultados obtenidos a partir de la modificación del parámetro Kp del controlador de nivel tipo P, para cumplir con los requisitos de velocidad y estabilidad en el prototipo de tanque de riego.

Tabla 8

Ajuste de parámetros para el controlador de nivel

Iteración	***Kp***	***Comportamiento del Sistema***	***Observaciones***
1	*4*	*El sistema es inestable y no puede mantener el nivel*	*Inestable, ajustar Kp*
2	*3*	*El sistema es más estable, pero el nivel disminuye debido a la pérdida de agua*	*Continuar reduciendo Kp*
3	*1.5*	*El sistema responde más rápido, pero no puede compensar completamente la perdida de agua*	*Ajuste fino de Kp*
4	*1.3*	*Respuesta más rápida y capacidad de recuperación después de la perdida de agua*	*Ajuste final del controlador*

Nota. Elaboración Propia

En la tabla 9, se pueden evidenciar los resultados obtenidos a partir de la modificación de los parámetros Kp y Ki del controlador de flujo tipo PI, para cumplir con los requisitos de velocidad y estabilidad en el prototipo de tanque de riego.

Tabla 9

Ajuste de parámetros para el controlador de flujo

Iteración	***Kp***	***Ki***	***Comportamiento del Sistema***	***Observaciones***
1	*4*	*0*	*Sistema inestable, incapaz de mantener el nivel*	*Inestable, ajustar Kp*
2	*3*	*0*	*Mas estable, pero el nivel disminuye debido a la pérdida de agua*	*Continuar reduciendo Kp*
3	*1.5*	*0*	*Respuesta más rápida, pero es incapaz de compensar completamente la pérdida de agua*	*Ajuste fino de Kp*
4	*1.4*	*1.2*	*Introducción de término Ki para compensar la pérdida de agua*	*Reducción del error en estado estacionario*
5	*1.6*	*1.4*	*Respuesta más rápida y capacidad de recuperación después de la perdida de agua*	*Eliminación parcial del error en estado estacionario*
6	*1.6*	*1.5*	*Ajuste adicional para mejorar la compensación de la pérdida de agua*	*Equilibrio entre velocidad y estabilidad*
7	*1.6*	*1.7*	*Mayor introducción de término Ki para compensar perturbaciones mayores*	*Ajuste final del controlador*

Nota. Elaboración Propia

Finalmente se logró encontrar los parámetros ideales de sintonización manual de los controladores. En el caso del controlador de flujo se escogió Kp = 1.6 y Ki = 1.7. Y para el controlador de nivel tipo P se escogió Kp = 1.3. Estos valores permitieron mejorar la respuesta del sistema en términos de velocidad y estabilidad.

En la tabla 10, se presenta la tabla de datos referencial de las pruebas de llenado del tanque, que muestra la relación entre caudal entregado por la bomba, volumen y tiempo de establecimiento.

Tabla 10

Parámetros de llenado del tanque

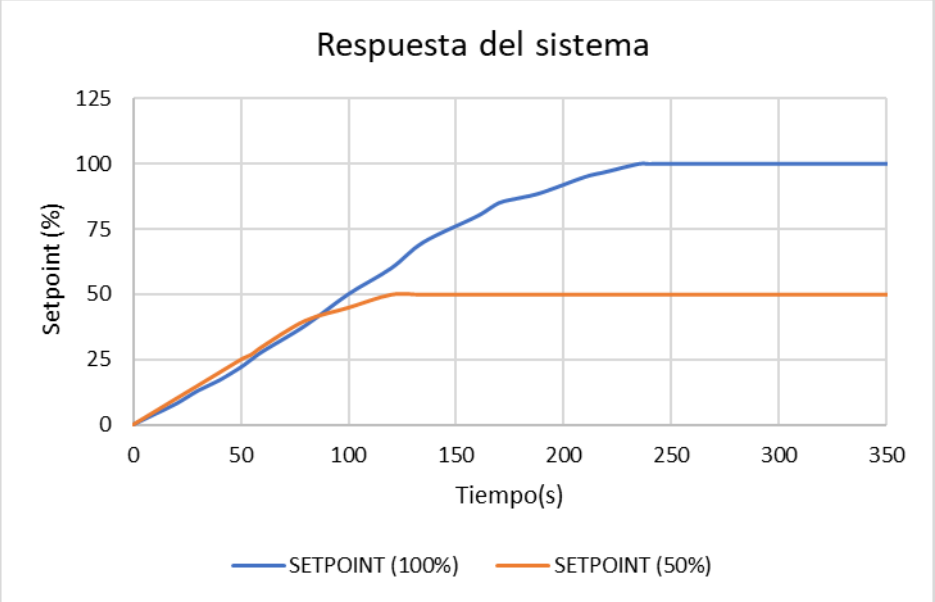
Tiempo (s)	*Setpoint (100%)*	*Setpoint (50%)*	*Caudal (L/m)*	*Volumen (L)*
0	*0*	*0*	*5.2*	*0*
10	*4*	*5*	*5.3*	*2*
20	*8*	*10*	*5.6*	*3.5*
30	*13*	*15*	*5.3*	*4.1*
40	*17*	*20*	*5.4*	*5.3*
50	*22*	*25*	*5.7*	*6.9*
55	*25*	*27*	*5.1*	*7.6*
60	*28*	*30*	*4.9*	*8.2*
80	*38*	*40*	*4.6*	*8.6*
100	*50*	*45*	*3.5*	*9.5*
120	*60*	*50*	*3.5*	*10*
135	*70*	-	*4.2*	*11.4*
160	*80*	-	*4.8*	*12.6*
170	*85*	-	*5*	*13.7*
180	*87*	-	*5.5*	*14.8*
190	*89*	-	*5.5*	*16.8*
210	*95*	-	*5.1*	*18.1*
220	*97*	-	*3.8*	*19.2*
235	*100*	-	*0*	*20*

Nota. Elaboración Propia

En la figura 48, se presenta el gráfico de la respuesta del sistema, cuando el tanque se encontraba vacío, donde se observó un tiempo de establecimiento de 235s para el setpoint 100% y un tiempo de establecimiento de 120s para el setpoint 50%. Esta prueba se hizo estableciendo los parámetros de sintonización ideales, además se consideró una perturbación de riego programado mientras el proceso de llenado de agua en el tanque estaba en curso. Este grafico refleja lo que ha sido el control de la variable principal.

Figura 48

Respuesta del sistema de control



Nota. Elaboración Propia

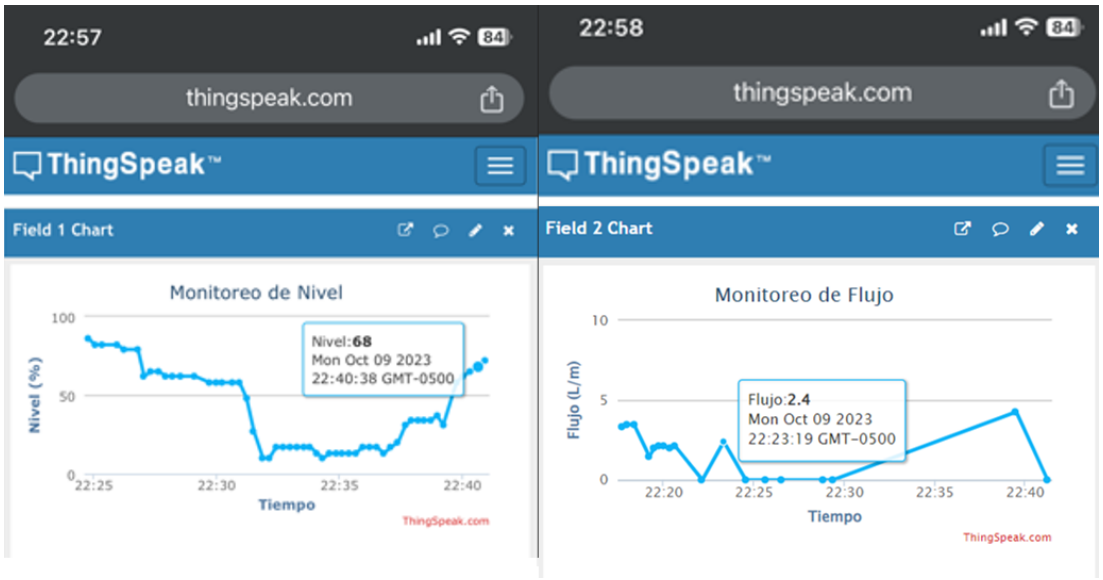
4.3. Monitoreo remoto de variables en ThingSpeak

Como se mencionó previamente en la sección 3.3, el monitoreo remoto se realizó accediendo a la plataforma ThingSpeak, la cual proporciona visualización de datos en tiempo real en la nube.

En la figura 49, se observa el monitoreo de los datos de las variables de nivel y flujo del prototipo de tanque de riego, a través de un dispositivo móvil.

Figura 49

Monitoreo de variables en dispositivo móvil

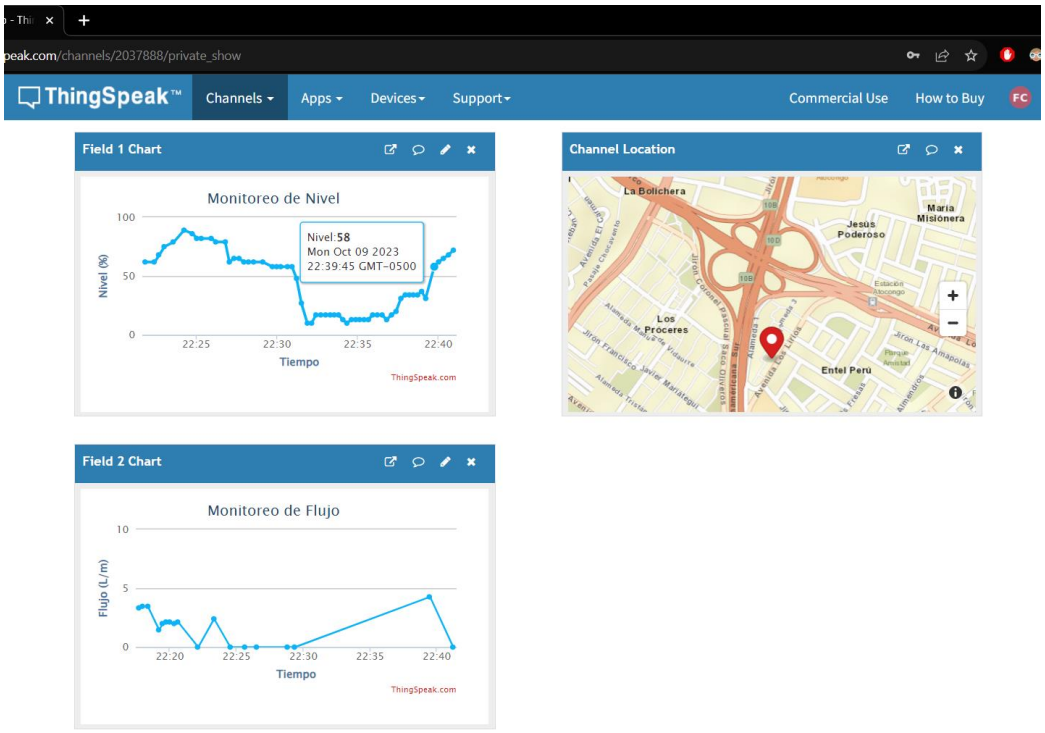


Nota. Elaboración Propia

Por otro lado, en la figura 50 se observa el monitoreo de los datos de las variables de nivel y flujo del prototipo de tanque de riego a través de una computadora portátil.

Figura 50

Monitoreo de variables en computadora portátil



Nota. Elaboración Propia

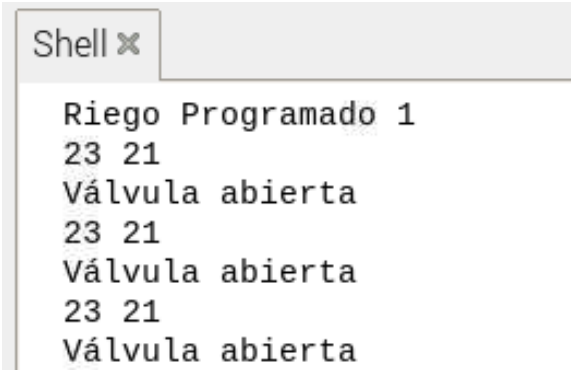
4.4. Pruebas de riegos programados

Por último, en el panel de control local, se ejecutó el Riego Programado 1 el cual fue preestablecido en el Script 3, donde se especificó que se aperture la válvula a las 23:21 horas y se cierre a las 23:27. Para ello, se puso en evidencia los mensajes que demuestran la acción de la válvula solenoide, según la hora en el tanque de riego.

En la figura 51, puede observar la hora actual y el mensaje de válvula abierta en el Shell del IDE de Thonny, según los horarios preestablecidos del Script 3.

Figura 51

Apertura de válvula – Inicio de riego programado

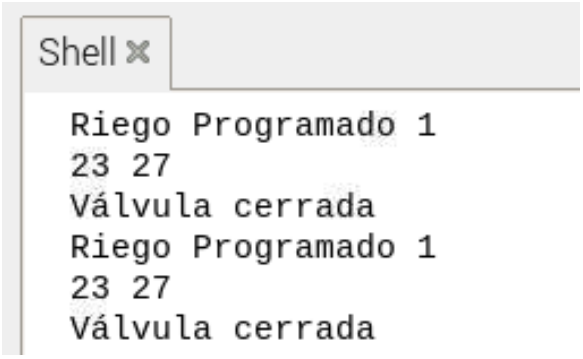


Nota. Elaboración Propia

En la figura 52 se puede observar la hora actual y el mensaje de válvula cerrada en el Shell del IDE de Thonny, según los horarios preestablecidos del Script 3.

Figura 52

Cierre de válvula – Fin de riego programado



Nota. Elaboración Propia

4.5. Presupuesto

El presupuesto utilizado para la implementación del prototipo de tanque de riego se detalla a continuación en la tabla 11.

Tabla 11

Presupuesto para la implementación del prototipo de tanque de riego

	Dispositivos/Bienes/Materiales		Precio (S/.)
	Características	**Cantidad**	
1	Raspberry PI 4 8GB	1	510
2	Sensor de Ultrasonido	1	45
3	Sensor de Flujo	1	25
4	Sensor HCSR	1	7.5
5	Bomba de agua	1	90
6	Tanque 25L	1	33
7	Válvula solenoide	1	25
8	Válvula de paso	2	40
9	Servomotor MG996R	1	24
10	Acople Servomotor	1	3
11	Impresión 3D	1	65
12	Modulo Relé	1	30
13	Display Oled	1	20
14	Display LCD 4x20	1	35
15	Adaptador T para GPIO	1	6
16	Cable plano 40 pines	1	5.5
17	Paquete de cables de conexión	4	20
18	Cable de fuente 12m	1	10

19	Pulsadores	4	10
20	Fuente de alimentación	1	80
21	Módulo Stepdown 12 a 5VDC 5A	1	10
22	Controlador de Velocidad 5A	1	7.5
23	Tubos	5	30
24	Adaptadores de Tubos	5	20
	TOTAL	**39**	**1101.5**

Nota. Elaboración Propia

CONCLUSIONES

1. La implementación del prototipo de tanque de riego a escala ha representado un paso significativo hacia la eficiencia en el control de las variables de nivel y flujo del proceso. La disposición y ubicación de los instrumentos y equipos para el control de las variables de nivel y flujo ha sido fundamental para la operación efectiva del sistema tal como se puede comprobar con mayor claridad en la sección 4.1.

2. Si bien es cierto que, para un algoritmo de control en cascada, la teoría recomienda que para el lazo primario se opte por un controlador tipo PI y para el lazo secundario un controlador tipo P, debido a que el proceso principal es más lento, en este proyecto de tesis que no es del tipo industrial se optó por lo contrario; es decir, después de realizar las correspondientes pruebas se eligió el tipo de controlador P para el lazo primario (variable nivel) y PI para el lazo secundario (variable flujo), porque la dinámica del proceso permitió una aceptable respuesta en términos de velocidad y estabilidad según los datos visualizados en la tabla 10, y también debido principalmente a la dimensión del prototipo de proceso implementado a escala. Por lo cual, los parámetros elegidos para el controlador de flujo fueron Kp=1.6 y Ki=1.7, mientras que para el controlador de nivel fue Kp=1.3, tal como fue descrito en la sección 4.2.

3. ThingSpeak ha demostrado ser una solución eficiente y efectiva para la recopilación y visualización de datos en tiempo real, como se comprobó en la sección 4.3, ya que permite la creación de hasta 4 canales, donde cada uno admite hasta 8 campos de datos. La integración exitosa del Internet de las Cosas en el prototipo de tanque de riego ha permitido un acceso remoto y continuo a los datos de las variables, lo que facilita la operación del sistema.

4. La capacidad de programar tareas de riego proporciona a los agricultores una buena herramienta para maximizar la productividad y minimizar el desperdicio de recursos. En la sección 4.4 se observó la eficacia del uso del riego programado, donde se asignó un horario específico de riego. Esta función del prototipo permite una operación precisa y programable para asignar un calendario de riegos, que se adapta a las necesidades específicas del cultivo, lo que conlleva a una mejor eficiencia del uso del agua.

RECOMENDACIONES

1. Antes de implementar el prototipo en el entorno real de la zona rural, es esencial llevar a cabo pruebas exhaustivas en un entorno controlado, esto permitirá identificar posibles problemas, ajustes necesarios en el diseño y la programación, además de asegurarse que todos los componentes funcionen correctamente. Igualmente realizar pruebas en condiciones controladas ayuda a minimizar el riesgo de errores costosos o ineficiencias una vez que el sistema esté en funcionamiento en un entorno real.

2. Se recomienda de que el algoritmo sea lo suficientemente robusto para funcionar en condiciones reales, donde pueden surgir variaciones y desafíos imprevistos, esto implica que se deben realizar numerosas pruebas para lograr la sintonización adecuada y la eliminación de perturbaciones. Su flexibilidad y la capacidad de respuesta es clave para garantizar que el prototipo de tanque de riego sea eficiente y sostenible.

3. Considerar la habilitación de un mayor número de funciones que ofrece la plataforma ThingSpeak, como, por ejemplo, retransmitir los datos hacia servicios de terceros como Twilio para notificar al usuario del prototipo de tanque de riego vía SMS, sobre eventos del sistema o Twitter para difusión de datos en caso sea necesario.

4. Añadir la posibilidad de que el operador del tanque de riego pueda establecer manualmente mediante un teclado numérico, la hora y minuto especifico de un riego programado, así como también poder asignar ciertos días de riego del mes, según la necesidad de los cultivos.

REFERENCIAS

Álvarez, J., Moreno, J., & Ramírez, E. (2017). Diseño e Implementación de un sistema de control en cascada en la planta de intercambio térmico - PIT000. *Informador Técnico*, 32-43.

Arias, F. (2012). *El Proyecto de Investigación 6° Edición.* Caracas: Editorial Episteme. Obtenido de https://www.researchgate.net/publication/301894369_EL_PROYECTO_DE_INVESTIGACION_6a_EDICION

Battikha, N. (2007). *The Condensed Handbook of Measurement and Control.* New York: ISA- The Instrumentation, Systems and Automation Society.

Bermeo, L., Álvarez, J., & Mantilla, W. (2021). Comparación del desempeño de un controlador PID sobre el proceso de nivel usando un controlador lógico programable y un sistema embebido. *Ingeniare. Revista chilena de ingeniería, vol. 29 N° 4 2021, pp. 622-632*, 622-632.

Control Automation. (20 de July de 2020). *Cascade Control - Basic Process Control Strategies*. Obtenido de Control Automation: https://control.com/textbook/basic-process-control-strategies/cascade-control/

Dimas, B., & Huamaní, P. (2019). Implementation of a cascade control strategy in the level module of a laboratory at the Ricardo Palma University. *LACCEI International Multi-Conference for Engineering, Education and Technology: "Industry, Innovation, and Infrastructure for Sustainable Cities and Communities"*. Jamaica.

Fuentes, J., Castro, S., Medina, B., Moreno, F., & Sepúlveda, S. (2018). Experimentación de controladores digitales clásicos en un sistema embebido aplicado en un proceso térmico. *Revista UIS Ingenierías Vol 17, no. 1, pp. 81-92, enero-junio 2018*, 81-92.

García, E. (2016). *Desarrollo de un controlador PID industrial de bajo coste mediante Raspberry PI para control de temperatura.* Valencia: Universidad Politécnica de Valencia.

Kavianpour, A. (2017). The Application of PID Control in Student Projects. *2017 ASEE Annual Conference & Exposition*, (pág. 15). Columbus, Ohio.

Lozada, J. (Diciembre de 2014). *Investigación Aplicada: Definición, Propiedad*

Intelectual e Industria. Obtenido de CienciAmérica: Revista de divulgación científica de la Universidad Tecnológica Indoamericana: https://dialnet.unirioja.es/servlet/articulo?codigo=6163749

Mogrovejo, D. (2017). *Diseño e implementación de un sistema de cuatro tanques interconectados con control PID robusto multivariable* . Cuenca: Universidad Politécnica Salesiana.

Naylamp. (2021). *Naylamp Mechatronics*. Obtenido de Naylamp Mechatronics: https://naylampmechatronics.com/

Oracle. (29 de Noviembre de 2019). *What is IOT?* Obtenido de Oracle: https://www.oracle.com/internet-of-things/what-is-iot/

Oxford. (s.f.). *Relé*. Obtenido de Oxford Languages: https://languages.oup.com/google-dictionary-es/

Raspberry. (17 de Abril de 2019). *¿Que es Raspberry PI?* Obtenido de Raspberry PI: https://raspberrypi.cl/que-es-raspberry/

Raspberry. (2019). *Raspberry Documentation*. Obtenido de Raspberry: https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#raspberry-pi-4-model-b

Sapiensman. (s.f.). *Industrial Controllers, Basic Theory*. Obtenido de Sapiensman: http://www.sapiensman.com/control/index.htm

Senamhi. (2016). *Análisis del Riesgo de Sequías en el Sur del Perú.* Lima: Senamhi.

Solutions, Visaya. (2021). *Control Algorithm*. Obtenido de Visaya Solutions: https://visaya.solutions/en/article/control-theory-types-of-control-algorithm

ThingSpeak. (2022). *ThingSpeak for IoT Projects*. Obtenido de ThingSpeak: https://thingspeak.com/

Valderrama, J. (2020). *Thingspeak para el monitoreo y control remoto de un micro invernadero automatizado.* Bogotá: Universidad de los Andes.

Woolf, P. (22 de May de 2022). *Cascade Control*. Obtenido de LibreTexts Engineering: https://eng.libretexts.org/Bookshelves/Industrial_and_Systems_Engineering/Book%3A_Chemical_Process_Dynamics_and_Controls_(Woolf)/11%3A_Control_Architectures/11.03%3A_Cascade_control-_What_is_it%3F_When_useful%3F_When_not%3F_Common_usage.

ANEXOS

Anexo A: Especificaciones técnicas del sensor de ultrasonido JSN-SR04T

JSN-SR04T - All-in-one ultrasonic distance measurement instruction manual

1 - product features:

The ultrasonic distance measurement module JSN-SR04T provides contactless distance sensing from 25 cm to 450 cm, with an accuracy of up to 3 mm. It includes an ultrasonic transmitter, receiver and a control unit. You can use it in the same way as the module HC-SR04 from JSN.

basic working principle:

(1) The IO port TRIG is used to trigger the ranging, giving a high-level signal for at least 10 µs.
(2) The module automatically sends eight square waves of 40 kHz and detects if a signal is returned.
(3) When a signal is returned, a high-level is output through the IO port ECHO, the duration of the high level is the time between the ultrasonic wave transmission an reception.
Test distance = (high level time * speed of sound (340m/s))/2

2 - pin assignment

Wiring as shown on the right
VCC for 5V supply
GND is the ground wire
TRIG is the trigger control signal input
ECHO is the echo signal output

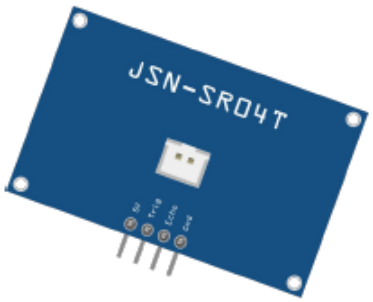


figure 1 - schematic board

3 - electrical parameters

electrical parameter	JSN-SR04T ultrasonic module
working voltage	DC 5 V
working current	40 mA
accoustic emission frequency	40 kHz
longest distance	4.5 m
shortest distance	25 cm
measuring angle	30 degree
input trigger signal	10 µs TTL pulse
output echo signal	output TTL level signal, proportional to range
size	41*29 mm
probe lead length	2.5 m

4 - ultrasonic timing diagram:

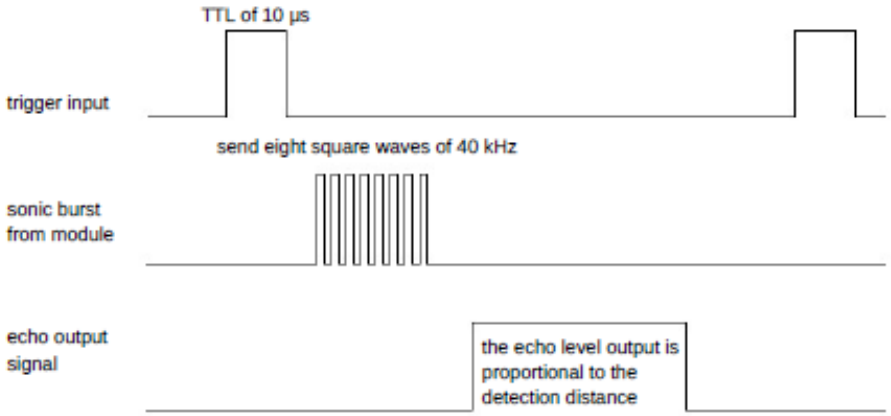


figure 2 - ultrasonic timing diagram

The above sequence diagram shows that you only need to provide a pulse trigger signal of more than 10 µs, and the module will send out eight cycles of 40 kHz and detect the echo. Once an echo signal is detected, an echo output signal is sent. The pulse width of the echo signal is proportional to the measured distance. The distance can be calculated from the duration between the transmitted signal and the received echo signal.
Formula: µs / 58 = cm or µs / 148 = inch; or: distance = high level time * speed of sound (340 m/s) / 2; the recommended measurement period is 60 ms or more to prevent the influence of the transmitted signal on the echo signal.

note: 1. If you connect the module to power, let the GND pin of the module be connected first, otherwise the normal operation of the module will be affected.

2. When measuring distance, the area of the object to be measured must be at least 0.5 square metres and the surface should be as flat as possible, otherwise the results of the measurement will be affected.

Anexo B: Especificaciones técnicas del sensor de flujo YF-S201

MODEL: YF-S201

Description:

Water flow sensor consists of a plastic valve body, a water rotor, and a hall-effect sensor. When water flows through the rotor, rotor rolls. Its speed changes with different rate of flow. The hall-effect sensor outputs the corresponding pulse signal. This one is suitable to detect flow in water dispenser or coffee machine. We have a comprehensive line of water flow sensors in different diameters. Check them out to find the one that meets your need most.

Features:

Compact, Easy to Install
High Sealing Performance
High Quality Hall Effect Sensor
RoHS Compliant

Specifications:

Working Voltage: DC 4.5V~24V
Normal Voltage: DC 5V~18V
Max. Working Current: 15mA (DC 5V)
Load capacity: ≤ 10 mA (DC 5V)
Flow Rate Range: 1~30L/min
Load Capacity: ≤10mA (DC 5V)
Operating Temperature: ≤80℃
Liquid Temperature: ≤120℃
Operating Humidity: 35%～90%RH
Allowing Pressure: ≤1.75MPa
Storage Temperature: -25～+ 80℃
Storage Humidity: 25%～95%RH
Electric strength 1250V/min
Insulation resistance ≥ 100MΩ
External threads: 1/2"
Outer diameter: 20mm
Intake diameter: 9mm
Outlet diameter: 12mm

Application:

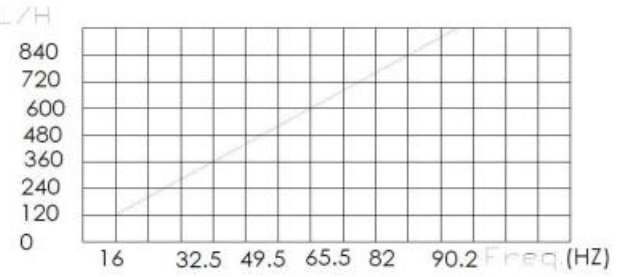
Water heaters, credit card machines, water vending machine, flow measurement device!

Cercuit:

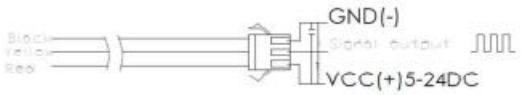
Red: Positive
Black: GND
Yellow: Output signal

Flow Range: 100L/H-/1800H-L/H

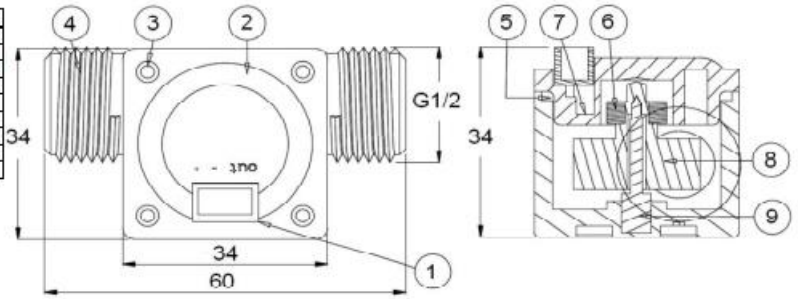
Flow (L/H)	Frezq (Hz)	Erro range
120	16	±10 5%
240	32.5	
360	49.3	
480	65.5	
600	82	
720	90.2	



Connection method:



N°	Item	Material
1	Wire	PVC
2	Bonnet	PA
3	Screw	Zinc Plated
4	Valve Body	PA
5	Press Valve	
6	Magnet	
7	Hall	
8	Impeller	POM
9	Steel Sharft	SUS304



Closed

Anexo C: Especificaciones técnicas de la válvula solenoide FPD-270A

1688 SUNLEPHANT官方旗舰店 品牌：SUNLEPHANT

Shenzhen Global Technology Co., Ltd

1" Electric Solenoid Valve

Model: FPD-270A-102- XXX

12VDC	5W
24VDC	5W
110VAC	5W
220VAC	5W

Primary technical indexes and notices of the product:

1. The working environmental conditions of the low-pressure electromagnetic inlet valve. Normally closed (N/C)

1.1 The elevation about sea level should be less than 2,500 meter.

1.2 The surrounding environmental temperature should be between 0°C and 40°C.

1.3 Relative air humidity should be below 95% (the temperature is 25°C).

1.4 Material: PP

2. Type and rated technical parameter

2.1 Type: guiding (single direction) solenoid.

2.2 Nominal voltage: Direct current: 12v, 24v ; Alternating current: 110v, 220v

2.3 Rated Power 5W

2.4 ON working: short-time (ON 3min. and OFF 5min.) (intermittent working period)

2.5 Insulation grade: E grade insulation

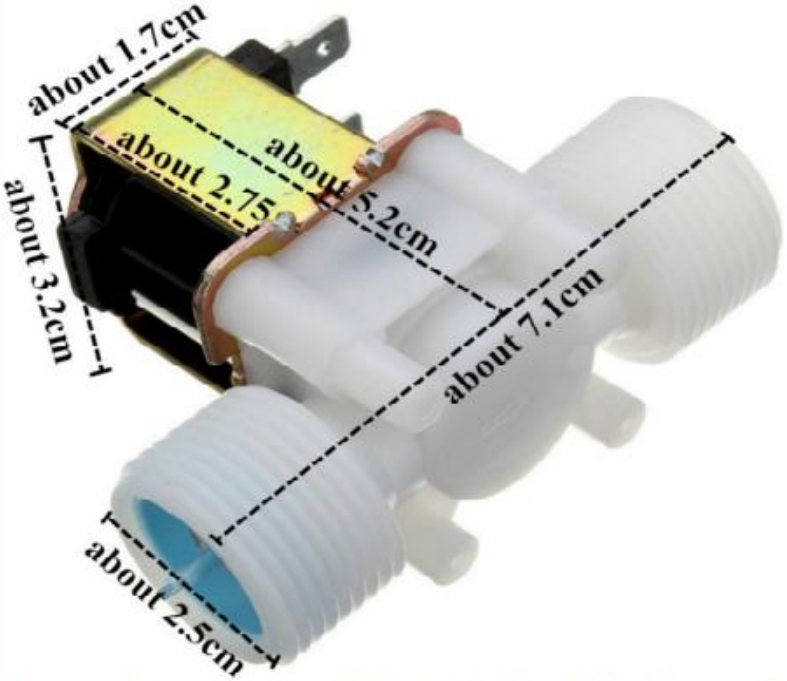
2.6 Fluid: (0-55°C) water

2.7 Water pressure: 0.02-0.8Mpa

2.8 Features of water flow

Pressure (Mpa)	Industry flow (L/min)	Domestic flow (L/min)
0.02	> 5	>2 - 3
0.1	>11	>8 - 10
0.3	>18	>13 - 13
0.8	>32	>25 - 28

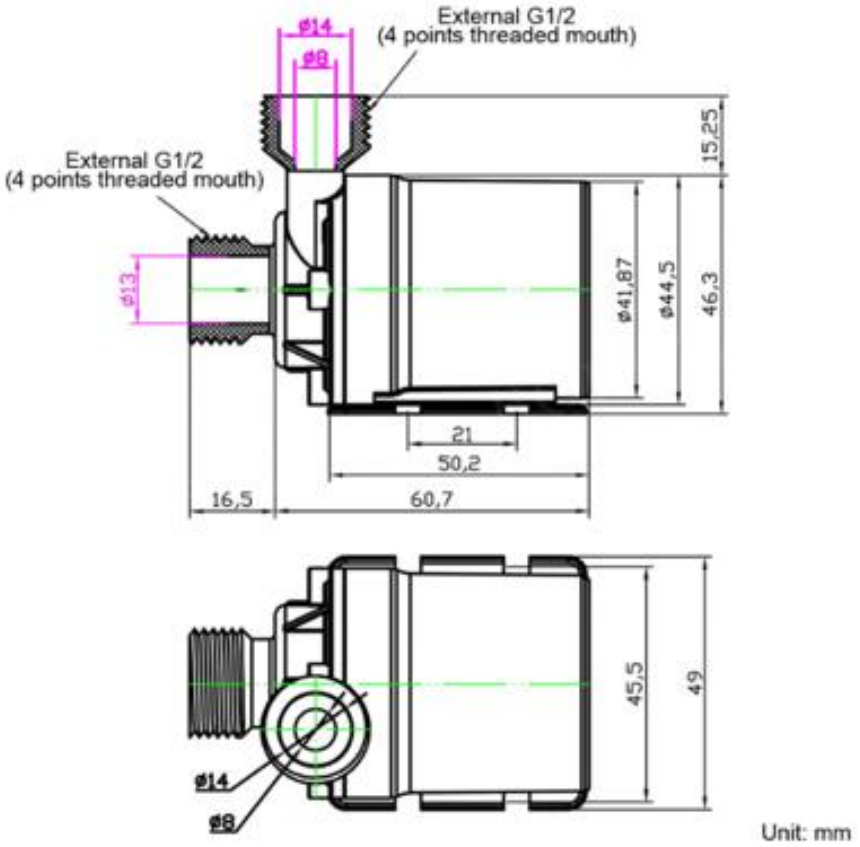
2.9 Longevity: over 100,000 times



Shenzhen City, Guangdong Science and Technology Co., Ltd. Address: China's Shenzhen Futian District, Shenzhen Huaqiang North Road

Copyright © 2010-2015 1688.com All rights reserved. Copyright and Trademark Notices

Anexo D: Especificaciones técnicas de la bomba de agua 5M 800L/H

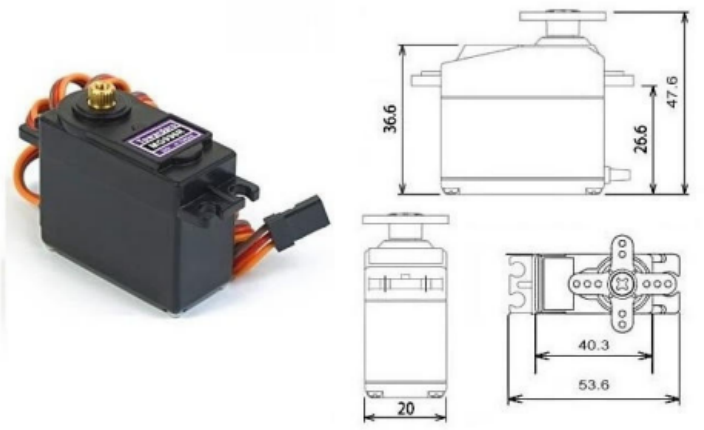


ESPECIFICACIONES TÉCNICAS

- Voltaje de Operación: 5-12V DC
- Corriente máxima: 350mA
- Potencia: 19W
- Caudal máximo: 800L/h (13L/min)
- Columna de agua máxima: 5m
- Conexión rosca: G1/2"
- Carcasa de plástico ABS negro
- Protección: IP68
- Ruido: <40dB a 0.5m
- Líquidos de trabajo: agua, aceite, gasolina
- Temperatura del fluido: 100°C máx.
- Dimensiones: 80*80*65 mm aprox.
- Peso: 220gr.

Anexo E: Especificaciones técnicas del servomotor MG996R

MG996R High Torque
Metal Gear Dual Ball Bearing Servo



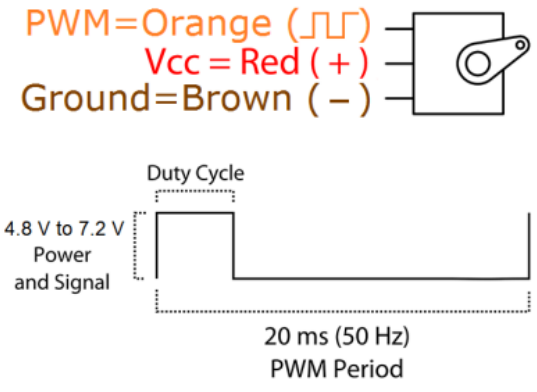
This High-Torque MG996R Digital Servo features metal gearing resulting in extra high 10kg stalling torque in a tiny package. The MG996R is essentially an upgraded version of the famous MG995 servo, and features upgraded shock-proofing and a redesigned PCB and IC control system that make it much more accurate than its predecessor. The gearing and motor have also been upgraded to improve dead bandwith and centering. The unit comes complete with 30cm wire and 3 pin 'S' type female header connector that fits most receivers, including Futaba, JR, GWS, Cirrus, Blue Bird, Blue Arrow, Corona, Berg, Spektrum and Hitec.

This high-torque standard servo can rotate approximately 120 degrees (60 in each direction). You can use any servo code, hardware or library to control these servos, so it's great for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. The MG996R Metal Gear Servo also comes with a selection of arms and hardware to get you set up nice and fast!

Specifications

- Weight: 55 g
- Dimension: 40.7 x 19.7 x 42.9 mm approx.
- Stall torque: 9.4 kgf·cm (4.8 V), 11 kgf·cm (6 V)
- Operating speed: 0.17 s/60º (4.8 V), 0.14 s/60º (6 V)

- Operating voltage: 4.8 V a 7.2 V
- Running Current 500 mA – 900 mA (6V)
- Stall Current 2.5 A (6V)
- Dead band width: 5 µs
- Stable and shock proof double ball bearing design
- Temperature range: 0 ºC – 55 ºC



Anexo F: Diagrama de bloques del control en cascada