Diseño e implementación de un robot móvil tele-operado con capacidades de reconocimiento de imágenes a través de Cloud Computing para la exploración de zonas distantes

Espinoza Castro, Miguel Alfredo y Nasimba Nasimba, Victor Fabricio

Departamento de Eléctrica, Electrónica y Telecomunicaciones

Carrera de Ingeniería en Electrónica, Automatización y Control

Trabajo de titulación, previo a la obtención del título de Ingeniero en Electrónica, Automatización y Control

Ing. Alulema Flores, Darwin Omar, PhD.

8 de noviembre del 2021

Informe de originalidad

NOMBRE DEL CURSO
Revisión de tesis

NOMBRE DEL ALUMNO
VICTOR FABRICIO NASIMBA NASIMBA

Firmado electrónicamente por:
DARWIN OMAR ALULEMA FLORES

NOMBRE DEL ARCHIVO
VICTOR FABRICIO NASIMBA NASIMBA - Tesis

SE HA CREADO EL INFORME
10 nov 2021

Resumen

Fragmentos marcados	6	0,9 %
Fragmentos citados o entrecomillados	3	0,5 %

Coincidencias de la Web

sawers.com.bo	6	0,9 %
agenciab12.com	1	0,2 %
amazon.com	1	0,2 %
aliexpress.com	1	0,1 %

DEPARTAMENTO DE ELÉCTRIA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

CERTIFICACIÓN

Certifico que el trabajo de titulación, "**Diseño e implementación de un robot móvil teleoperado con capacidades de reconocimiento de imágenes a través de Cloud Computing para la exploración de zonas distantes**" fue realizado por los señores **Espinoza Castro, Miguel Alfredo y Nasimba Nasimba, Victor Fabricio** el cual ha sido revisado y analizado en su totalidad por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 08 de noviembre del 2021

DARWIN OMAR ALULEMA FLORES

.....

Ing. Alulema Flores, Darwin Omar

C. C. 1002493334

DEPARTAMENTO DE ELÉCTRIA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

RESPONSABILIDAD DE AUTORÍA

Nosotros, **Espinoza Castro, Miguel Alfredo y Nasimba Nasimba, Victor Fabricio**, con cédulas de ciudadanía n° 0105392674 y n° 1722410543, declaramos que el contenido, ideas y criterios del trabajo de titulación: **Diseño e implementación de un robot móvil tele-operado con capacidades de reconocimiento de imágenes a través de Cloud Computing para la exploración de zonas distantes** es de nuestra autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Sangolquí, 08 de noviembre del 2021

.....	.....
Espinoza Castro, Miguel Alfredo	**Nasimba Nasimba, Víctor Fabricio**
C.C.: 0105392674	C.C.: 1722410543

DEPARTAMENTO DE ELÉCTRIA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

AUTORIZACIÓN DE PUBLICACIÓN

Nosotros, **Espinoza Castro, Miguel Alfredo y Nasimba Nasimba, Víctor Fabricio,** con cédulas de ciudadanía n° 0105392674 y n° 1722410543, autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **Diseño e implementación de un robot móvil tele-operado con capacidades de reconocimiento de imágenes a través de Cloud Computing para la exploración de zonas distantes** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de nuestra responsabilidad.

Sangolquí, 08 de noviembre del 2021

.....	.....
Espinoza Castro, Miguel Alfredo	**Nasimba Nasimba, Víctor Fabricio**
C.C.: 0105392674	**C.C.: 1722410543**

Dedicatoria

Este trabajo va dedicado:

A mis padres Janeth y Miguel, por su amor incondicional, sacrificio y estar a mi lado apoyándome para no rendirme sin importar la distancia ni las dificultades pues el haber cumplido esta meta es principalmente gracias a ellos.

A mis tíos y primos por siempre poder contar con su apoyo y cariño, quienes compartieron por muchos años un lugar en su mesa y tuvieron las puertas abiertas de su hogar.

A mis amigos que compartieron conmigo gratos momentos de felicidad y me ayudaron a superar varias adversidades, los considero parte de mi familia.

Miguel Alfredo Espinoza Castro

Dedicatoria

Virgen María, Madre de El Quinche, enséñame a amar, como lo hiciste tú, para vivir en paz y con serenidad; enséñame a esforzarme, como tú Madre, para cumplir las metas señaladas por Dios; enséñame a entregarme, y ser para los demás, las manos de Dios.

Esta tesis está dedicada:

A mis padres, Victor y María quienes con su amor, paciencia y esfuerzo me han permitido llegar a cumplir hoy un sueño más.

A mis compañeros de vida, Marisol y Mateo por ser mi motivación e inspiración para superarme y poder crecer juntos.

A mis hermanos, Danny y Gabriela por haber sentado en mí el deseo de superación e impulso para salir adelante.

A mis abuelitos paternos, Gustavo y Rosario por sus oraciones y palabras de aliento para poder lograr mis objetivos.

A mis abuelitos maternos, Alberto y Rosa por haberme enseñado que siempre hay que luchar hasta el final.

Al resto de mi familia, amigos y compañeros que de una u otra manera han contribuido para lograr este objetivo.

Victor Fabricio Nasimba Nasimba

Agradecimiento

Agradezco a mis padres por brindarme la oportunidad de cumplir este sueño, son el mejor ejemplo de vida y siempre serán para mí la mayor motivación para continuar avanzando sin rendirme y que puedan sentirse orgullosos.

A mis tíos y primos más cercanos que siempre me comparten sus experiencias de vida y por sus ánimos durante esta etapa de mi vida.

A mi tutor, Ing. Darwin Alulema, PhD, por darnos la confianza de realizar esta investigación bajo su guía al ofrecernos su tiempo y conocimiento para desarrollar un trabajo de muy alta calidad.

A mi compañero de tesis por su determinación y constancia durante el transcurso de la investigación dando lo mejor de sí y culminando con un trabajo muy bien realizado.

Finalmente agradezco a mis amigos que sin importar las circunstancias siempre puedo contar con su apoyo y hemos pasado momentos llenos de bromas e ironías que no se olvidarán nunca en la vida.

Miguel Alfredo Espinoza Castro

Agradecimiento

Agradezco a Dios y a la Virgencita de El Quinche por regalarme su bendición, sabiduría y salud para disfrutar del don de la vida y poder lograr este sueño.

A mi madre, por haberme forjado como la persona que soy, por brindarme siempre sus palabras de aliento e inculcar en mí el bonito valor de la humildad.

A mi padre, mi más grande ejemplo, por su amor, esfuerzo y sacrificio para hacer que no me falte nada en la vida y apoyarme incondicionalmente para lograr este objetivo.

A mi Sol, por haber estado conmigo durante toda esta etapa de mi vida brindándome sus palabras de aliento y por haberme regalado mi más grande motivación.

A mis hermanos, por acompañarme en este duro camino, siendo mi fuente de energía y motivación para no rendirme.

A mis abuelitos, ejemplo de sencillez y humildad, por brindarme siempre sus sabios consejos, oraciones y las más grandes enseñanzas de vida.

A mi tutor, por guiarnos en el desarrollo de esta gran investigación, a mi compañero de tesis por su esfuerzo y dedicación para culminar con éxito este trabajo.

Y a mis demás familiares, amigos, compañeros y conocidos por el apoyo moral que me han brindado a lo largo de esta etapa.

Agradezco por todo lo que he recibido y todo lo que aún está por llegar... GRACIAS.

Victor Fabricio Nasimba Nasimba

Contenido

Índice de tablas

Índice de Figuras

Resumen

Hoy en día el Internet es la base principal de comunicación donde los avances de la tecnología enfocada al Internet de las Cosas es una realidad que se ve involucrada en ámbitos de estudio e investigación para las industrias. Es así como las grandes empresas dedican gran cantidad de tiempo y recursos por la búsqueda de la mejor y adecuada implementación de un sistema conformado con nuevos protocolos de comunicación junto con la integración de servicios y la factible interacción del hombre con la tecnología. Por estas razones surge la propuesta de una arquitectura Cloud IoT orientada a robótica móvil utilizando para la comunicación, tecnología de largo alcance como lo es LoRa y además ofrecer la capacidad de reconocimiento de imágenes a través de Cloud Computing, el robot está destinado para exploración de lugares no aptos para el acceso humano. La arquitectura está estructurada por la capa Edge computing que integra al robot móvil, Fog computing con los protocolos de comunicación a larga distancia, el Cloud computing para la orquestación del uso de microservicios y mensajería MQTT para finalmente terminar con el Front End correspondiente a la interfaz de usuario. Tanto la funcionalidad, usabilidad y eficiencia fueron evaluadas a través de múltiples pruebas que resultaron ser satisfactorias en la implementación del proyecto.

PALABRAS CLAVES:

- **ARQUITECTURA ORIENTADA A SERVICIOS**
- **COMUNICACIÓN A LARGA DISTANCIA**
- **CLOUD COMPUTING**
- **MQTT**

Abstract

Nowadays the Internet is the main basis of communication where the advances in technology focused on the Internet of Things is a reality that is involved in areas of study and research for industries. This is how large companies dedicate a great amount of time and resources in the search for the best and adequate implementation of a system conformed with new communication protocols together with the integration of services and the feasible interaction of man with technology. For these reasons arises the proposal of a Cloud IoT architecture oriented to mobile robotics using for communication long-range technology such as LoRa and also offer the ability to image recognition through Cloud Computing, the robot is intended for exploration of places not suitable for human access. The architecture is structured by the Edge computing layer that integrates the mobile robot, Fog computing with long distance communication protocols, Cloud computing for the orchestration of the use of microservices and MQTT messaging to finally finish with the Front End corresponding to the user interface. Both the functionality, usability and efficiency were evaluated through multiple tests that proved to be satisfactory in the implementation of the project.

KEYWORDS:

- **SERVICE-ORIENTED ARCHICTETURE**
- **LONG DISTANCE COMMUNICATION**
- **CLOUD COMPUTING**
- **MQTT**

Capítulo 1: Marco metodológico

Antecedentes

Con el transcurso del tiempo es evidente el avance tecnológico en la industria manufacturera encontrando una relación entre las TIC (Tecnologías de la Información y Comunicación) y las Industrias por la búsqueda de mejorar el desarrollo del país con oportunidades de crecimiento que aseguren un impacto positivo en la economía nacional (MINTEL, Industria Manufacturera, 2019). Con la aplicación de los pilares de la Industria 4.0 en los procesos productivos aumenta la eficiencia y flexibilidad, mejora funcionalidades y como uno de los puntos principales se logra una mayor integración de la electrónica (SMIT, 2016).

En este sentido, el desarrollo de sistemas robotizados mejorando el rendimiento informático pretende alcanzar la capacidad de aprendizaje y adaptación en cuanto a las acciones que se realizan en un medio específico, como lo es la Inteligencia Artificial con un amplio campo de investigación con el fin de mejorar la calidad de vida del ser humano aplicando tecnología para aprender y procesar tareas que sean de necesidad para el hombre de forma eficiente con un inmenso análisis de datos creando herramientas de alta confiabilidad (MINTEL, Inteligencia Artificial y Machine Learning, 2019). Aunque se piense que las principales ramas a estudiarse para poder emplear la Inteligencia Artificial sean la informática y la robótica, no solo se queda ahí pues existe un aporte a varios campos como la parte social y empresarial, pues la obtención del análisis de datos que se realizan en tiempo real gracias a la Inteligencia Artificial permite un procesamiento mucho grande y rápido de datos para lo toma de decisiones que se deben llevar a cabo dentro de una Industria (Miaibe, 2018).

La Revolución Industrial 4.0 considerada como una nueva de la incorporación de la tecnología de automatización para facilitar la vida del ser humano tanto, a nivel industrial, doméstico, transporte y demás ámbitos, está constituido por el Internet de las

Cosas o IoT en donde se tiene planificado que todos los aparatos electrónicos se encuentren conectados a internet y puedan ser controlados por órdenes del usuario. Con el IoT todos los elementos que se encuentren conectados establecen una misma comunicación entre ellos, es decir la capacidad de coordinación de tal manera que las acciones que se realizan sean por decisiones que pueda ser de carácter simple o crítico ante un determinado evento (Gardaseive, 2017). Si se habla de expandir este concepto se podría estar ya hablando del Smart City es decir que en la red se podrán incorporar servicios que sean utilizados en los hogares sino también en las grandes industrias para los procesos de fabricación, en el transporte, sector agrícola, proyectos de investigación y exploración militar, en fin una amplia gama de aplicación en casi la mayoría de los servicios, siempre y cuando la disponibilidad de conexión al servicio de internet pueda ser desde cualquier lugar.

El puente para poder enlazar al mundo físico con el mundo cibernético se lo llama Edge Computing, es una conexión entre los usuarios y dispositivos mediante sistemas Edge que involucran estudios en computación, comunicación, base de datos, servicios y control, pero con el objetivo de la existencia de baja latencia, alto rendimiento, gran ancho de banda y garantizar la seguridad. El Edge Computing tiene una estrecha relación con el IoT debido a las conexiones y control de numerosos dispositivos por medio de una misma red de servicios de Internet. Existe la propuesta de múltiples plataformas para otorgar servicio de alta calidad y eficiencia, se hace uso de algoritmos genéricos que en dependencia del tipo de aplicación o finalidad necesitada exista la posibilidad de adaptación hacia los servicios de tal manera que se pueda trabajar de forma rápida y ordenada, en otras palabras, se puede definir niveles para caracterizar a los servicios a disposición (Hu & Chen, 2019). Pues la actual tecnología se basa en la forma de comunicación, conexión y procesamiento de información, es decir, realizar un gestionamiento del Cloud Computing, logrando notar como los pilares de la Industria 4.0

se van relacionando, ahora el tratamiento de datos es a través de la nube manteniendo todo enlazado (MINTEL, Edge Computing, 2019).

Otro de los pilares de la Industria 4.0 es el Cloud Computing que conforma un nuevo paradigma del procesamiento, control, almacenamiento y acceso a la información que se encuentra compartida en el medio cibernético, permitiendo establecer variedad de servicios de utilidad tanto para las grandes industrias como para el hogar. Con el Cloud Computing se puede tener accesibilidad a múltiples plataformas que se encuentran en la red las cuales permiten visualizar, procesar y controlar datos. Una ventaja a considerar de las plataformas es la división que las conforman según las necesidades y aplicaciones que se requieran teniendo así una mejor eficiencia y rapidez de uso de recursos entregando un servicio de estabilidad para un amplio número de clientes al mismo tiempo (MINTEL, Cloud Computing, 2019).

La aplicación de los robots para solución de tareas y actividades es una de las realidades actuales, principalmente los robots han sido implementados para realizar actividades de alta precisión o en entornos que puedan poner en riesgo la vida de las personas, pero los robots funcionan a base de programación y ésta a su vez también ha evolucionado conjuntamente, pues surge el concepto denominado robótica colaborativa que son técnicas y estrategias incorporadas en el robot dándole un grado de independencia y se pueda reducir accidentes, es decir, que sin intervención de un operario, el robot este en la capacidad de continuar con sus actividades, pueda interactuar con elementos a su alrededor o tomar decisiones (Molina, 2021). La robótica colaborativa tiene fuerte uso en el medio industrial, a pesar de esto no quiere decir que se encuentre delimitado su campo de aplicación pues se propone su uso para robots de exploración que son operados a larga distancia y los tiempos de latencia en la transmisión de información son bastante elevados a partir de esto nace la necesidad de que el robot sepa que secuencia de acciones ejecutar a partir de información externa mediante

interacciones de protección, monitoreo de prevención de colisiones cuando no disponga de ninguna instrucción por parte del operador y se encuentre en un territorio o ambiente desconocido.

Para poder satisfacer los requerimientos de robótica colaborativa es necesario un control. Es aquí donde se da a conocer el criterio de los sistemas de control de robots que tienen como objetivo resguardar la seguridad de relación humano - robot, robot - robot y robot - entorno mediante diseño de esquemas con los cuales se pueda integrar el control a través de niveles dinámicos con la capacidad de procesamiento de información, posicionamiento del robot, detección de colisión, cálculo del modelo dinámico, todo en tiempo real con el uso de controladores propios integrados en la infraestructura del robot (Tao, Li, Liu, Deng, & Xin, 2018). Los recursos tecnológicos para lograr conseguir el desarrollo de esta temática son aquellos que puedan aportar con amplios anchos de banda, alta velocidad de procesamiento, software y hardware actualizado y especializado en robótica de manera que el sistema de control del robot tenga alto rendimiento, calidad y seguridad.

Las formas con los cuales se diseñan los robots son en base a su funcionalidad, el caso de los robots móviles y tele - operados se tiene a los que pueden desplazarse a través de terrenos irregulares y zonas inaccesibles para el ser humano, por ejemplo los robots de exploración que pueden ser de uso militar en zonas de alto riesgo para desactivación de explosivos, o aquellos que son utilizados para exploración en la luna u otros planetas con el fin de obtener información del medio y muestras para análisis del terreno. Es una realidad que los robots se encuentran ya entre nosotros por lo que debemos aprender a adaptarnos y cooperar con avances de investigación a fin de mejorar el nivel académico no solo como persona sino también en representación del país.

Justificación e importancia

La implementación de la arquitectura Cloud IoT orientada a la robótica móvil, constituye una solución atractiva en la actualidad para diferentes campos, como la educación, la seguridad, ciudades inteligentes y principalmente la industria, entre otras, ya que de esta manera puede existir una interconexión e integración de los robots móviles instalados en el mundo físico con el mundo de la informática. La arquitectura Cloud IoT está cada vez más presente en la vida cotidiana, ya que cuenta con un gran potencial. Pues su objetivo es el desarrollo de aplicaciones y servicios para mejorar la calidad de vida de la sociedad, y aplicada a la robótica para automatizar actividades optimizando la ejecución, etapas de fabricación y costos. Como bien se ha mencionado, la implementación de una arquitectura Cloud IoT orientada a la robótica móvil abre las puertas a muchas y nuevas oportunidades de desarrollo, pero al existir un gran número de dispositivos y servicios interconectados, también se plantean algunos problemas y desafíos.

En el país ya no es ajeno el concepto de Cloud IoT y se empieza a notar una cierta tendencia de las empresas ecuatorianas a la adopción de esta tecnología. Sin embargo, existen pocos estudios que relacionen esta tecnología con la robótica, específicamente con la robótica móvil y por ende no se definen con claridad las formas de adoptar estas tecnologías para aplicaciones específicas, como consecuencia se tiene un desconocimiento de los beneficios de este tipo de servicios.

Algunos temas de investigación desarrollados, relacionados directamente con esta propuesta de investigación, son los que se describen a continuación. Como primer tema expuesto es presentado por (Alava Bravo, 2019), en el cual se hace uso del servicio de internet para integrar una arquitectura con modelos de servicio Cloud Computing con el objetivo de procesamiento de datos provenientes de una aplicación de teléfono celular, la información es compartida en una base de datos y analizada con la plataforma Amazon

Web Services. En este tema se establece un bajo flujo de datos, a diferencia de las aplicaciones actuales donde el flujo de datos es muy alto, especialmente en el manejo de robots móviles, ya que requieren grandes paquetes de información tanto en la transmisión como en la recepción. Es por eso que, a diferencia de la tesis mencionada con respecto a este proyecto de investigación, se comunicará a un robot móvil en un medio real para transmitir información tanto de GPS y Streaming para que con ayuda de los servicios de Cloud computing se pueda realizar un reconocimiento de imágenes, por ende, el flujo de datos es muy alto y como consecuencia se deben utilizar herramientas de hardware y software para que se adapten a tal fin.

El segundo trabajo aportado por (Córdova Lara & Medina Torres, 2018) detalla como un robot móvil es controlado a través de internet, mediante métodos de tele operación aplicados hacia la robótica y con el uso de visión artificial utilizando el algoritmo Viola Jones, presentan la información a través de un HMI y utilizando software a Python, Ubuntu y OpenCV. En la actualidad las necesidades del ser humano para el control de robots van más allá de únicamente controlar o visualizar su recorrido, sino también de poder manipular los objetos que están en el entorno del robot. Como consecuencia en este proyecto de investigación se implementará el control del recorrido y del manipulador del robot que se utilizará en este proyecto, a través de un mando a grandes distancias o mediante la generación acciones y rutas que posteriormente serán cargadas en el robot para que las ejecuten independientemente gracias a que está dotado de tecnología GPS, y para la dotación de visión artificial no se usarán librerías de programas para tal fin, ya que están propensos a no brindar un servicio continuo ni eficiente. Para tener un mejor servicio y dar una mayor robustez a la visión artificial, en este proyecto se usarán plataformas como Amazon Web Services, Always AI, Google, entre otros mediante el uso de los servicios de Internet con la interacción con el Cloud Computing.

El tercer apartado realizado por (Figueroa Chiriboga & Tiuma Chafla, 2019), presenta una estructura seccionada en módulos con un diseñ acorde a la cinemática y dinámica del robot. La tele - operación permite la comunicación entre el operario y la plataforma robótica a través de un módulo inalámbrico como es WI-FI. Para el caso del proyecto de investigación no solo se realizará video en tiempo real sino que también este video será utilizado para el reconocimiento de objetos, además el robot tendrá un manipulador para la realización de otras funciones, a la vez todos los datos serán almacenados en la nube mediante los protocolos detallados en la descripción de nuestro proyecto y no solo por WI-FI como es el caso del tema en cuestión, asimismo este proyecto dispondrá de una HMI que no solo controlará el robot a través del video, sino que también se podrá controlar el manipulador y verificar la posición del robot por las prestaciones de posicionamiento mediante GPS que brindan las tarjetas que serán usadas.

El cuarto tema es de (Chriboga Torres, 2020). A diferencia de esta tesis que solo se envían los datos a un servidor web para el monitoreo del vehículo, en este proyecto de investigación se enviaran datos de video, GPS, y a la vez el robot recibirá datos de instrucciones para poder manipular el robot mediante tecnología LORA.

Por último, se tiene el escrito de (Gonzales Bonifaz & Verdugo Cabrera, 2018), donde se diseña una arquitectura IoT capaz de recibir información de robots móviles, almacenarla, procesarla y determinar una acción adecuada. Como factor diferenciador claro de este proyecto de investigación con respecto a esta tesis es la tele - operación del robot a grandes distancias con la obtención del posicionamiento mediante GPS, y el control del manipulador que incluirá el robot, y otros servicios como el reconocimiento de objetos que brindará la estructura IoT.

A continuación, en la tabla 1, se resumen los trabajos analizados, con lo que se obtiene una visión más amplia con respecto a lo desarrollado o líneas de investigación que aún no se han abordado.

Como se puede notar los temas de investigación en el área del Cloud IoT orientados a la robótica móvil son muy escasos en el país, es por eso que mediante la realización de este proyecto de investigación con las prestaciones y beneficios anteriormente detallados, se brindará al país la posibilidad de contar con esta tecnología emergente e innovadora llena de oportunidades que permite mayor agilidad, elasticidad, capacidad de almacenamiento, redundancia y ahorros de costes en este tipo de tecnologías a largo plazo, que gracias a su versatilidad se le puede dar diferentes enfoques en los campos de aplicación, ya sea en el ámbito industrial para aplicaciones de robótica colaborativa o en el ámbito de la seguridad, como un aporte a los organismos de seguridad del estado ecuatoriano para la intervención en operaciones que signifiquen un alto riesgo para el ser humano.

El desarrollo del proyecto de investigación no solo será un aporte tecnológico al país, sino también será la apertura a una iniciativa para el cambio de la matriz productiva, ya que el cloud IoT orientado a la robótica móvil dentro de aplicaciones industriales, seguridad, entre otras, está relacionado directamente con la economía digital; ya que hoy en día por motivos de la pandemia la economía ha sido afectada pero las tecnologías de la información, la industria 4.0 y las plataformas IoT y sobre todo la robótica se han masificado en los países desarrollados, y es muy importante considerar la idea de que como nación tenemos el anhelo de volver a la “normalidad”, pero se debe reflexionar que el resto de países a tomado la situación de pandemia como una oportunidad para implementar las tecnologías antes mencionada, y gracias a ello se están adelantando tecnológicamente para adaptarse a la realidad actual.

Tabla 1

Resumen de trabajos analizados para la justificación del proyecto

Titulo	Base de datos Cloud	Amazon Web Services	Reconocimiento de imágenes	Robótica móvil	Tele-operación	Geolocalización	Comunicación mediante Tecnología WI-FI	HMI	Aplicación de Tecnología LoRa	Arquitectura IoT	Robótica colaborativa
Desarrollo de una arquitectura Cross-Device para Cloud Computing aplicada a un sistema EHealth [18]	X	X									
Diseño y construcción del prototipo de un robot móvil para telepresencia controlado a través de Internet [19]	X		X		X	X		X			
Diseño y construcción de un prototipo de robot todo terreno, tele-operado y dotado de visión remota para el Centro de Investigación Científica y Tecnológica del Ejército CICTE [20]					X		X	X			
Diseño e implementación de una solución con tecnología LORA para el monitoreo de ubicación vehicular con un aplicativo web [21]						X			X		
Diseño e implementación de una arquitectura IOT para robótica colaborativa [22]				X				X		X	X
Nuestro proyecto de investigación	X	X	X	X	X	X	X	X	X	X	X

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permitan movilizar al robot hacia adelante.

Además, se tendrá la posibilidad de contar con tecnología que permitirá un ahorro de procesamiento y masificación de datos, para de esta manera poder desarrollar aplicaciones que puedan desplegarse más rápidamente, ya que al tener los servicios ejecutándose en la nube se evita un procesamiento local, ahorrando recursos de hardware. Por otra parte, también se logrará hacer un despliegue con tecnologías inalámbricas de largo alcance para robots móviles permitiendo que no solo sea una aplicación ilustrativa o demostrativas, sino una tecnología que se pueden utilizar en entorno reales, y de esta manera se podrá lograr solventar los problemas descritos y brindar una nueva tecnología que tendrá una arquitectura estandarizada, que permita a usuarios y empresas integrarla de manera óptima para las aplicaciones que la requieran, y al ser una plataforma escalable pueden surgir muchos temas relacionados para proyectos futuros y de esta manera ampliar aún más la visión que se tiene con nuestro proyecto de investigación.

Alcance del Proyecto

El presente proyecto pretende alcanzar los siguientes hitos:

Se implementará una arquitectura Cloud IoT orientada a la robótica móvil que se observa en la figura 1, la cual estará compuesta por dos etapas como son: Back end, que aborda el Cloud Computing, Fog Computing y Edge Computing; y para interacción con el usuario se añadirá Front-End, que corresponde al HMI con el cual se interactúa.

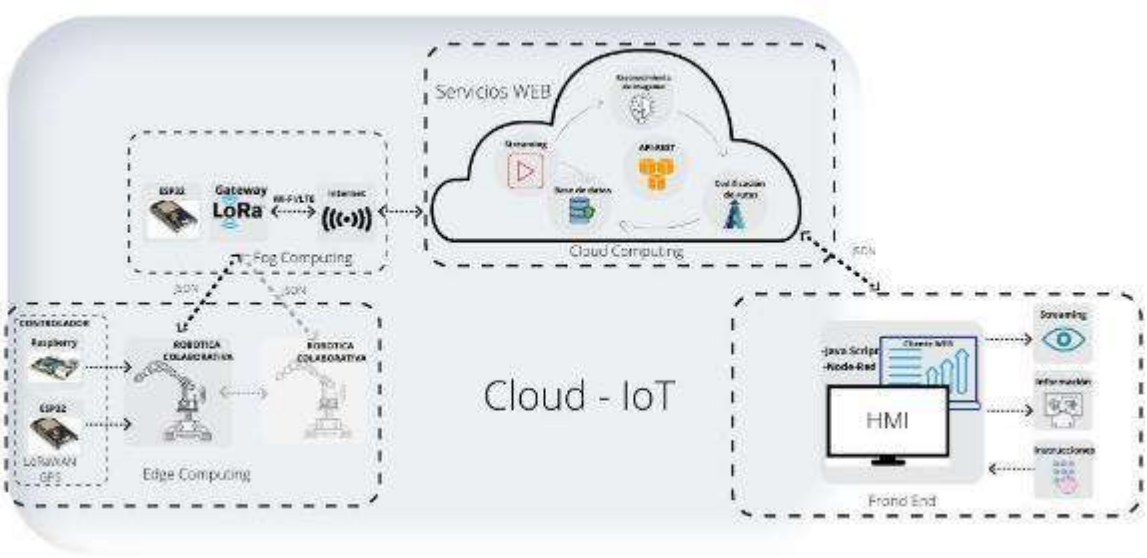
Edge Computing: La capa de Edge estará compuesta principalmente por un robot móvil que dispondrá de un manipulador (brazo robótico) para ejecutar diferentes instrucciones físicas, como es la manipulación de objetos, también contará con una cámara en la parte frontal para que pueda ser tele-operado y para su control integrará una tarjeta Raspberry pi y una tarjeta SP32. La tarjeta Raspberry PI realizará las funciones de controlar al robot, en la cual se procesarán los datos que van a dar la inteligencia

necesaria al robot para ejecutar las instrucciones recibidas y tomar sus propias decisiones dependiendo de las circunstancias en las que se encuentre.

En cuanto a la tarjeta SP32 se controlará y administrará la comunicación del robot con la capa de Fog Computing por medio de su protocolo de red LoRaWAN, que debido a sus prestaciones de comunicación a grandes distancias (hasta 10 Km) se podrán enviar los paquetes de datos, video y posición ya que la tarjeta cuenta con GPS, estos datos serán enviados en paquetes tipo JSON con la finalidad de tener una comunicación a larga distancia. También en esta capa se dejará prevista la configuración necesaria para realizar una comunicación en malla entre robots en proyectos futuros.

Figura 1

Arquitectura Cloud IoT



Nota: Etapas y capas que conforman la arquitectura Cloud IoT.

Fog Computing: La capa de Fog computing está entre la capa de Edge Computing y la de Cloud Computing, ayudará a la comunicación entre estas últimas. Esta capa estará compuesta por un nodo central tipo LoRaWAN que funcionará como equivalente al protocolo IEE802.3 o al protocolo Bus CAN para poder recibir los datos

provenientes de la capa de Edge computing y poder transmitirlos hacia la nube o viceversa.

Se dispondrá de una tarjeta SP32 con su respectiva tecnología LoRa, tecnología escogida debido a sus prestaciones para comunicación a grandes distancias que funcionara como Gateway que mediante tecnología WI-FI o LTE que tienen un gran ancho de banda, se enviarán o recibirán los datos del internet. En esta capa los datos recibidos por el robot estarán en dos paquetes tipo JSON, uno de datos y otro de imagen, que serán manejados sobre HTTP que funcionara conjuntamente con MQTT.

Cloud Computing: La capa Cloud Computing es considerada una capa de control, funciona a partir el servicio de Internet que recibe información en formato JSON del Fog Computing, se encuentra conformada por un API REST con el fin de realizar requerimientos a múltiples servicios, mediante procesos de control con la orquestación de estos servicios como lo es una base datos, codificación de rutas, streaming y reconocimiento de imágenes, a través de servicios web de plataformas como Always AI, Amazon Web Service, Google, entre otros, con el fin de realizar el procesamiento de imágenes en el cloud.

Front-End: Por último, se tiene la capa Front-End donde se va a tener la HMI para poder interactuar con el usuario, compuesta principalmente por Java Script y Node-Red, mostrará al usuario el streaming e información recolectada por el robot en la capa de Edge Computing, por otro lado, cuenta con la disponibilidad de enviar instrucciones de control para el robot a ser ejecutadas, teniendo en cuenta que la información se enviará al Cloud Computing para codificar su ruta.

Una vez en funcionando el proyecto de investigación se pretende evaluar cada uno de los componentes creados tanto de hardware como de software mediante una serie de pruebas específicamente relacionadas con el ámbito de la seguridad para la tele-

operación en lugares peligrosos y el reconocimiento de objetos que pueden proporcionar un riesgo al ser humano.

El proyecto de investigación en el que se elaborara una arquitectura de Cloud IoT orientado a la robótica móvil requiere de un mayor esfuerzo de análisis, diseño, construcción e implantación del producto, con referencia a los temas mostrados en detalle anteriormente. Este esfuerzo adicional puede ser estimado bajo las siguientes métricas:

Desarrollo del Hardware: Se diseñará e implementará un robot que posea un brazo manipulador y una cámara para la visión artificial, su inteligencia será dotada por una tarjeta Raspberry pi y la comunicación será proporcionada por tarjetas SP32, Routers y módulos para comunicación LTE, y por último monitores para servir de HMI. Todos estos elementos deben formar Arquitecturas Orientas a Eventos, Arquitecturas Orientas a Servicios, Procesamiento de Eventos Complejos, Sistemas Ciberfísicos, Ingeniería Dirigida por Modelos. El dimensionamiento se lo realizara con el objetivo de que estos elementos y arquitecturas funcionen para la tele-operación y control a distancia, por lo que requiere un mayor esfuerzo de diseño y análisis, ya que, por la ubicación de nuestro país, se deben considerar factores climáticos y geográficos que influirán en el tiempo de desarrollo del hardware.

Desarrollo del software: Se manejarán varias herramientas de software como son: Python, Java Script, LoRa, Arduino IDE. Para la arquitectura Cloud IoT se utilizan plataformas como Amazon Web Services, Always AI, Google, entre otros. Debido a que cada una de las herramientas que se pretende utilizar en nuestro proyecto cuentan con su respectivo lenguaje de programación, o estructura de funcionamiento, se requiere una gran cantidad de conocimiento lo que conlleva un gran tiempo de estudio y esfuerzo.

Implantación del software y hardware: Para permitir la comunicación e interconexión entre el software y hardware que manejara nuestro proyecto de investigación se utilizaran varias tecnologías de comunicación como son: MQTT, HTTP,

REST, WI-FI, LTE. Así mismo, se utilizarán protocolos de comunicación como: LoRaWAN que funcionará como equivalente al protocolo IEE802.3 o al protocolo Bus CAN, que manejará paquetes de datos tipo JSON. El uso de estas tecnologías y protocolos requieren un profundo análisis para que se acople a nuestra aplicación y también la ejecución de pruebas continuas para verificar el correcto funcionamiento, lo que se traduce en una gran inversión de tiempo y estudio.

Pruebas de funcionamiento: El proyecto de investigación se evaluará en distintos escenarios, utilizando las métricas correspondientes, donde se dará un realce a las pruebas ejecutadas en el ámbito de la exploración, que verificará el correcto funcionamiento de la tele-operación a grandes distancias fusionado con todos los servicios anteriormente detallados.

Objetivos

Objetivo General

Diseñar e implementar un robot móvil tele-operado con capacidades de reconocimiento de imágenes a través de Cloud Computing para la exploración de zonas distantes.

Objetivos Específicos

- Investigar el estado del arte de sistemas Cloud Computing, reconocimiento de imágenes, robótica colaborativa, tele – operación y arquitecturas IoT.
- Investigar arquitecturas de comunicación a larga distancia para la transmisión y recepción de información a través de servicios de la nube.
- Analizar las tecnologías para aplicar los conceptos de robótica colaborativa, control de robots móviles y telecomunicaciones.
- Diseñar la arquitectura Cloud Computing IoT para el control de un robot móvil, con el propósito de definir los mecanismos para la transmisión de imágenes vía

streaming y tele-operación a través de servicios Cloud, además del reconocimiento de imágenes e interacción del usuario por medio de una interfaz.

- Implementar los componentes de hardware para la construcción del robot móvil e implementar los servicios Cloud para el control y tele-operación del sistema.
- Analizar los resultados sobre pruebas de funcionamiento, pruebas de usabilidad y pruebas de carga de servicios Cloud del sistema.

Estado del Arte

Metodología

Se establecen métodos existentes con el objeto de poder encontrar un repositorio con información requerida de las cuales según criterios en dependencia de los deseado se pueda reducir el número de documentos con el fin de fijar una cantidad adecuada que pueda definir, argumentar y aportar sobre las arquitecturas Cloud, microservicios, robótica móvil y protocolos de larga distancia, como metodología se pretende utilizar el SMS Systematic Mapping Study y el SLR Systematic Literature Review (Moguel Márquez, 2018).

Mapeo Sistemático de la Literatura SMS. La metodología SMS Systematic Mapping Study traducido como mapeo sistemático de la literatura pretende la revisión, identificación y valoración de aquellos artículos, tesis, proyectos y toda clase de publicaciones científicas referentes al uso de la tecnología con robots móviles y el uso de arquitecturas Cloud, lo cual radica en sí como una estrategia para seleccionar la información a citarse en el presente trabajo relacionado al IoT o Internet de las Cosas (Kitchenhan & Charters, 2004).

En la actualidad la finalidad es utilizar la tecnología para un bien común, con el uso de la arquitectura Cloud se pretende que el manejo de dispositivos representados como hardware esté representado en software a través de la Web en donde los usuarios

puedan manipularlos sin la necesidad de estar presente en el medio físico, el SMS procura recopilar todas aquellas publicaciones que mencionan al tema de tal forma que se defina un área de trabajo que aún no se la haya experimentado.

La forma de interpretar la estructura del mapeo sistemático de la literatura depende de los autores, sin embargo, los pasos en sí son los mismos y se los puede observar en la figura 2.

Figura 2

Estructura del mapeo sistemático de literatura

Nota: Figura tomada de (Moguel Márquez, 2018).

1. Planteamiento. Se establecen preguntas de forma conceptual que abarque los tópicos como microservicios, arquitecturas orientadas a servicios, robótica móvil, para que sirvan de guía para las siguientes fases del mapeo y determinar la parte investigativa, a continuación, se presentan las preguntas P con las motivaciones M (Panizzi, 2019).

P1. ¿Qué tecnologías y aplicativos tiene la robótica móvil?

M1. Conocer la morfología de los robots, sensores, control y modelos matemáticos

P2. ¿Cuáles son las tecnologías disponibles para el desarrollo de arquitecturas Cloud IoT?

M2. Conocer cómo realizar una arquitectura y el diseño de su estructura.

P3. ¿Qué tecnologías existen para servicios web?

M3. Conocer los proveedores de servicios web que son de libre uso.

P4. ¿Cómo se orquesta los microservicios web?

M4. Conocer plataformas o medios para orquestar los microservicios al desarrollar una aplicación.

2. Localización. En esta sección denotamos aquellos motores a disposición de los estudiantes universitarios para consultar información verificada y confiable, en este caso se utilizó IEEE Xplore. Repositorio ESPE, Springer Link y Scielo, luego para realizar la búsqueda la recomendación es ingresar el texto en forma de cadena y enlazando los tópicos mediante conectores booleanos como AND y OR teniendo así para cada pregunta las siguientes cadenas (Panizzi, 2019).

C1. “Mobile robotics” OR “robot morphology”

C2. “Cloud architectures” OR “Cloud architecture design”

C3. “Web Services”

C4. “Microservices orchestration”

Al colocar todos los términos en una sola cadena para la búsqueda se tiene (“Mobile robotics” OR “robot morphology”) AND (“Cloud architectures” OR “Cloud architecture design”) AND (“Web Services”) AND (“Microservices orchestration”), pero al colocar esto los resultados encontraron fueron insuficientes por lo que se realizó las mismas búsquedas solo que se organizó a las cadenas de 3 diferentes maneras teniendo así:

Cadena combinada 1: (“Cloud architectures” OR “Cloud architecture design”) AND (“Web Services”) AND (“Microservices orchestration”)

Cadena combinada 2: (“Mobile robotics” OR “robot morphology”) AND (“Web Services”)

Cadena combinada 3: (“Mobile robotics” OR “robot morphology”) AND (“Web Services”) AND (“Microservices orchestration”)

Posterior se definen criterios de inclusión y exclusión es decir seleccionar las condiciones que vamos a incorporar al momento de realizar la búsqueda para que puedan satisfacer la búsqueda de las cadenas previamente establecidas, destacando así:

Inclusión. Documentos en todos los idiomas y todo tipo de publicaciones.

Exclusión. Documentos de pago que imposibiliten el acceso a su lectura, anteriores a 2010, duplicados, no pertinentes o que no expliquen a detalle lo que se requiere.

3. Preanálisis. Las estrategias para extraer información depende a como el autor se acomode o encuentre facilidad por ejemplo (Moguel Márquez, 2018) propone el uso de fichas para registrar las publicaciones y a su vez valorar la relevancia, el tipo de estudio y aplicación. El autor (Navarro & Ramírez, 2018) establecen ya las condiciones para la búsqueda de información a través de los motores de búsqueda. En la tabla 2 podemos observar los resultados obtenidos de publicaciones totales basándonos en los criterios de inclusión

Tabla 2

Resultados de búsqueda con los criterios de inclusión

Motores de búsqueda	Cantidad de documentos
IEEE Xplore	2629
Springer Link	2838
Scielo	10
Repositorio ESPE	869
Total	11746

Nota: La presente tabla da a conocer todos los resultados de los motores de búsqueda y el total de documentos adquiridos.

En el momento de clasificar las publicaciones de interés, se descarta aquella información que se encuentre bajo los criterios de exclusión al número total de documentos con el objetivo de encontrar la información de mayor relevancia para el tema propuesto. A pesar de esto la cantidad de publicaciones aún es extensa por lo que se decide aplicar un filtro con mayor especificación para encontrar aquellas publicaciones

que den un ejemplo de una arquitectura Cloud implementada y explicada a detalle resultando tener una cantidad total de 45 publicaciones, el proceso se lo puede apreciar en la figura 3.

Figura 3

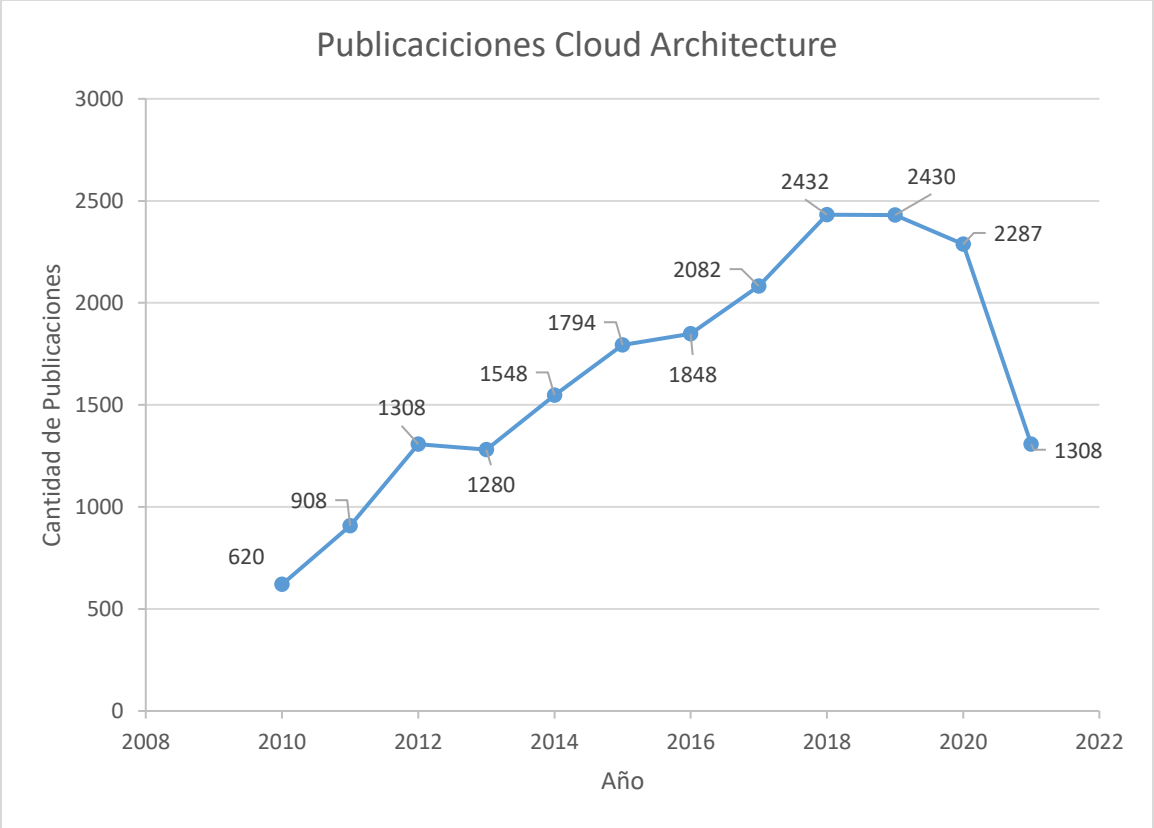
Aplicación de criterios de exclusión y un filtro de especificación en los documentos encontrados

Nota: Se indican los documentos resultantes después de pasar por cada criterio de exclusión y un último filtro exhaustivo.

4. Análisis. Definido ya cuales fueron los documentos para la extracción de información será nuestro pequeño repositorio de consulta. Por otra parte, podemos analizar, gracias al SMS, otros aspectos como el aumento de la cantidad de documentos por año sobre los temas de las preguntas impuestas, así como el tipo de publicación ya sea artículos, revistas, conferencias o demás que los motores de búsqueda facilitan para buscar (Moguel Márquez, 2018). Encontrando así en primer lugar la cantidad de publicaciones por año, en este caso no se aplican los criterios de inclusión ni de exclusión para un registro completo, se observan los resultados en la figura 4.

Figura 4

Publicaciones por año con el tema "Arquitectura Cloud"

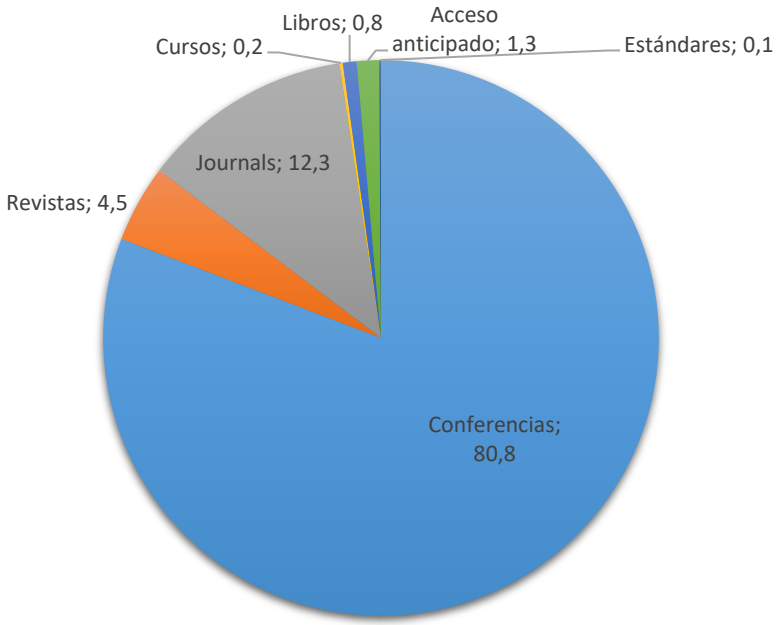


Nota: Se indica la gráfica de como aumenta el número de investigaciones sobre la Arquitectura Cloud hasta medio año del 2021.

De la figura 4 se deduce que en los años de 2018 a 2019 es en donde se tuvo la mayor cantidad de investigaciones referentes a la arquitectura Cloud lo cual apoya en gran parte al desarrollo del proyecto proporcionando información actualizada y confiable. Las publicaciones encontradas son de diferentes tipos dependiendo de la publicación por lo que se puede encontrar en diversas presentaciones teniendo en la figura 5 la respectiva representación donde claramente la mayoría de publicaciones son de conferencias con un 80.8%.

Figura 5

Tipos de publicaciones por año con el tema "Arquitectura Cloud"



Nota: La gráfica representa el porcentaje de publicaciones existentes para cada tipo.

Las preguntas propuestas hacen referencia a los tópicos principales tratados en el apartado de trabajos previos del presente trabajo por lo que se utilizó como un mini repositorio con el fin de obtener información que aportó para la sustentación del estado del arte evidenciándose en la sección bibliográfica, el uso del SMS permite seleccionar de toda la información disponible aquella de mayor relevancia que aporte con lo necesario y que no sea repetitivo desarrollando así un escrito con aportaciones aceptables.

Revisión sistemática de la literatura SRL. Con el SMS se pudo dar valor a la documentación relevante para responder a dudas sobre la arquitectura Cloud, los microservicios y la orquestación, aunque para el proyecto falta otras temáticas que faltan ser profundizadas como es la transmisión de video streaming y los protocolos de comunicación a larga distancia para lo cual se trabajó con el SRL Systematic Literature

Review el cual difiere del SMS en que la búsqueda es llevada a mayor profundidad pero con etapas casi iguales (Moguel Márquez, 2018).

Se pretende conocer cómo enlazar una arquitectura Cloud con un servicio de video Streaming y que se comunique con el dispositivo o hardware a través de protocolos de comunicación a larga distancia, como parte de sus tecnologías de acceso de manera que se complemente el SMS con información más acorde al tema propuesto y que sea de ayuda para el desarrollo del mismo.

El SLR se distingue del SMS en que se rige con 3 etapas fundamentales, planificación que es la coordinación y creación de un plan para el SLR, ejecución que es la selección junto con la edición de información y elaboración referido a la representación de los resultados encontrados (Moguel Márquez, 2018). A todo esto, los pasos a seguir son los mismos que se tenían en el SMS representado en la figura 6.

Figura 6

Estructura de la revisión sistemática de la literatura

Nota: Figura tomada de (Moguel Márquez, 2018).

1. Planteamiento. Las preguntas para el SLR son acordes al investigar una relación de la arquitectura Cloud utilizando servicios de video streaming, mensajería, reconocimiento de imágenes y protocolos de comunicación con el hardware a larga distancia.

P1. ¿Qué son los servicios de video streaming aplicado a la arquitectura Cloud?

M1. Conocer cómo se incorpora un servicio de video streaming directo desde el hardware para ser tratado en la arquitectura Cloud.

P2. ¿Cuáles son los microservicios web para el reconocimiento de imágenes?

M2. Conocer de proveedores para el reconocimiento de imágenes en streaming.

P3. ¿Qué tecnologías de mensajería existen?

M3. Conocer algún servidor con el servicio de mensajería.

P4. ¿Qué tecnologías utilizan los protocolos de comunicación a larga distancia?

M4. Conocer qué tecnologías se utilizan y cómo se pueden utilizar.

2. Localización. Al igual que en SMS se utilizaron los motores de búsqueda IEEE Xplore. Repositorio ESPE, Springer Link y Scielo, donde las cadenas unidas mediante conectores booleanos como AND y OR correspondientes a las preguntas son:

C1. “Video streaming” OR “Streaming services” OR “Streaming with Cloud architecture”

C2. “Image recognition” OR “Machine learning”

C3. “Messaging services” OR “MQTT”

C4. “Long distance protocols” OR “LoRa”

La cadena para la búsqueda por SLR sería (“Video streaming” OR “Streaming services” OR “Streaming with Cloud architecture”) AND (“Image recognition” OR “Machine learning”) AND (“Messaging services” OR “MQTT”) AND (“Long distance protocols” OR “LoRa”). Se definen los criterios de inclusión y exclusión de forma mejor detallada que en el SMS, teniendo de esa manera:

Inclusión. Documentos en todos los idiomas, publicaciones en conferencias, proyectos de grado o revistas, campo de la Ingeniería electrónica o software, contener protocolos, tópicos en robótica móvil, diseño de arquitectura Cloud o tecnologías de comunicación a larga distancia, publicaciones completas.

Exclusión. Documentos de paga, carreras que no tengan que ver con la ingeniería, documentos que no contengan información sobre el formato para la transmisión de datos, traten a profundidad los temas impuestos en las preguntas y no solamente como una breve mención, publicaciones anteriores a 2010, duplicados, no pertinentes

3. Preanálisis. Aplicando los criterios de inclusión, los resultados de cada motor de búsqueda y el total de publicaciones se muestran en la tabla 3.

Tabla 3

Resultados de búsqueda con los criterios de inclusión en el SLR

Motores de búsqueda	Cantidad de documentos
IEEE Xplore	503
Springer Link	620
Scielo	2
Repositorio ESPE	217
Total	1342

Nota: La presente tabla da a conocer todos los resultados de los motores de búsqueda y el total de documentos adquiridos por SLR.

A diferencia del SMS vemos como la cantidad de publicaciones encontradas con el SLR en los diferentes motores de búsqueda son mucho menores pues las cadenas de búsqueda originadas en base a las nuevas preguntas específicas, así como los múltiples criterios de inclusión son más complejos y específicos.

Luego al aplicar ya los criterios exclusión que son más estrictos a la hora de desechar información insuficiente hace que el proceso sea suficiente hasta este punto para que la cantidad de publicaciones se reduzca en primer nivel a 665, luego un segundo nivel a 323, aplicando además otros filtros hasta llegar a 83 y finalmente teniendo como resultantes 15 publicaciones, por lo que ya no es necesario de otro filtro que sea exhaustivo, pues se tiene la información más detallada y mejor argumentada para aportar al proyecto, esto se lo puede observar en la figura 7.

Figura 7

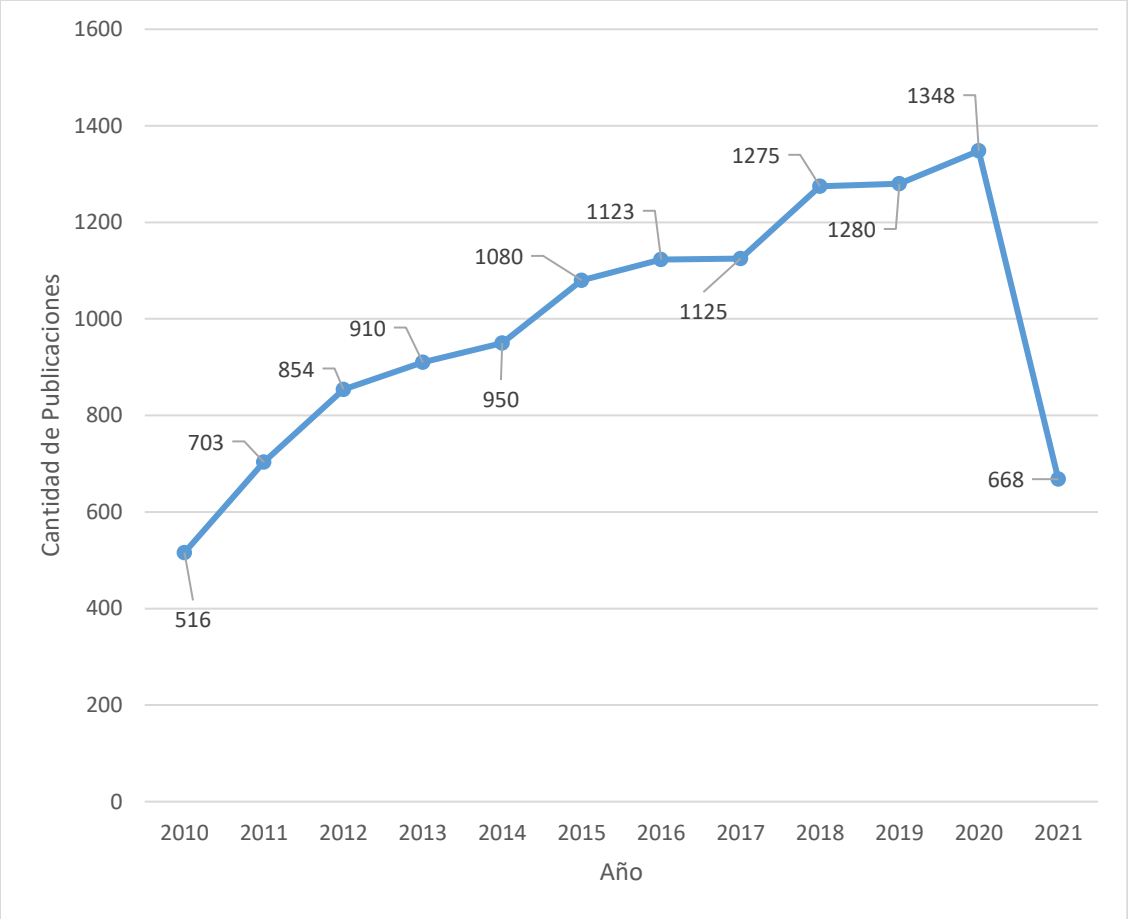
Aplicación de criterios de exclusión con SLR.

Nota: Se indican los documentos resultantes después de pasar por cada criterio de exclusión aplicados con la revisión sistemática de la literatura.

4. Análisis. Definida la cantidad de publicaciones en las cuales se va a realizar la consulta para el estado del arte se concreta que son las suficientes para poder argumentarlas y citarlas en el estado del arte. Como se hizo con el SMS también en el SLR se puede realizar un breve estudio de la cantidad de publicaciones anuales con las nuevas cadenas de búsqueda, así como el tipo de publicación, a continuación, en la figura 8 podemos apreciar en un gráfico de líneas la cantidad de publicaciones obtenido gracias a los motores de búsqueda sin aplicar ningún criterio (Moguel Márquez, 2018).

Figura 8

Publicaciones por año con el tema "Arquitectura Cloud" aplicando SLR

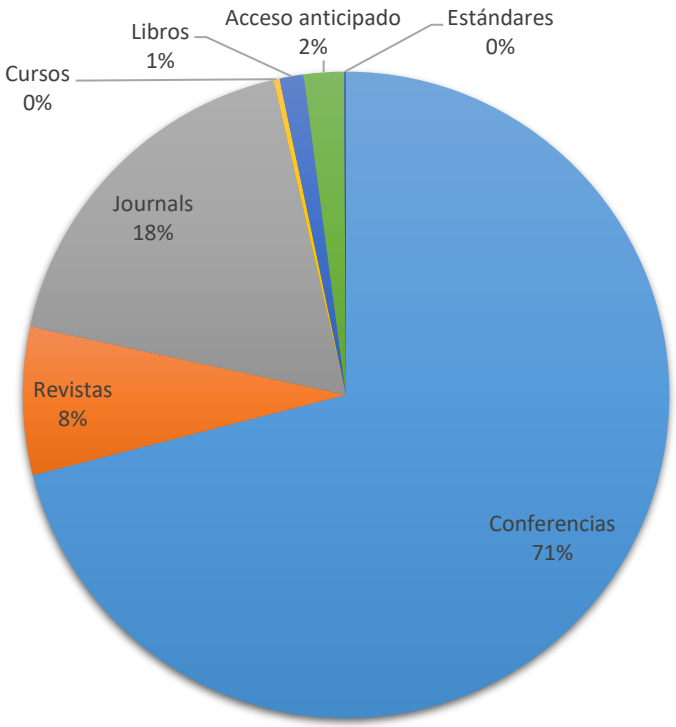


Nota: Se indican la gráfica de como aumenta el número de publicaciones anules hasta 2021 aplicando SLR

De la figura 8 se deduce que en el año 2020 es en donde se tuvo la mayor cantidad de publicaciones por lo que se puede concluir que las publicaciones de este año harán grandes aportaciones para el trabajo. Las publicaciones encontradas son de diferentes tipos dependiendo de la publicación por lo que se puede encontrar en diversas presentaciones representadas en la figura 9 donde la mayoría de publicaciones son de conferencias con un 71%.

Figura 9

Tipos de publicaciones por año aplicando SLR



Nota: La gráfica representa el porcentaje de publicaciones existentes para cada tipo en base a la búsqueda utilizando SLR.

Las preguntas expuestas con mayor afinidad hacia el trabajo redactadas utilizando las etapas del SLR son las que nos servirán también para complementar con el SMS para redactar el estado del arte en la sección de trabajos previos.

Trabajos Previos

La robótica móvil da la apertura a un campo de aplicación mediante arquitecturas estructuradas en capas para la ejecución de diversas tareas, activando cada tarea una a la vez pues cada una debe ser armoniosa evitando así la presencia de errores o ejecuciones incompletas, a su vez, como el Internet de las Cosas IoT lo requiere, se combina la parte de hardware, que son los robots físico, con el software, como la

arquitectura, fomentando la facilidad en el manejo de multiprocesos en tiempo real con criterios de programación robótica mencionado por (Santos, Gilmore, Hempel, & Sharif, 2017).

Para (Pecka, Zimmermann, Reinstein, & Svoboda, 2016) la morfología de los robots móviles es de un nivel de dificultad elevado pues este tipo de robots están destinado a exploración por ende los terrenos que atraviesan tienen muchas irregularidades, además estos robots al ser tele - operados elevan aún más su complejidad de manejo para el usuario, razón por la cual la morfología de los robots móviles se impone en manera que se establezca el estado en el cual el robot va a funcionar así como el terreno o área de movilización de manera que se garantice hasta cierto punto su funcionamiento.

El Cloud computing como mencionan (Hossain, Khan, Noor, & Hasan, 2016) se ha convertido en una aplicación de alto renombre en la industria con un despliegue de múltiples servicios con una cantidad ilimitada de recursos, pero esto es solo de forma teórica pues aún está en estaba de investigación en donde muchos investigadores utilizan servicios de varios proveedores para desarrollar aplicaciones con distintas finalidades, al ser el proveedor quien especifica sus propias configuraciones hace que su accesibilidad para cualquier usuario sea difícil y que no todos tengan acceso para trabajar con ello, convirtiéndose en una buena oportunidad para los empleadores y en donde los competidores buscará utilizar los servicios de aquellos proveedores con mejores recursos y precios más asequibles.

Como mencionan (Banijamali, Heisig, Kristan, Kuvaja, & Oivo, 2019) seguir adecuadas estrategias para diseñar las arquitecturas Cloud es lo que lleva al éxito o fracaso de una aplicación que se encuentra funcional en alguna plataforma, estas estrategias se crean a partir de decisiones, elección de atributos, incorporación de estilos y como ya se lleva mencionando las correspondientes evaluaciones de metodologías.

Estos autores proponen explicar cuáles son los diseños más comunes a implementar dependiendo del escenario y aplicación para ofrecer altos estándares en los atributos de calidad.

El trabajo de (Liu, et al., 2016) establece que los Web Services se crean como si fueran una aplicación para satisfacer una necesidad pues se dice que es un servicios el cual incorpora varios servicios individuales, se coordinan las solicitudes de los usuarios pues esto ya se conoce como una era del software orientado a servicios de ingeniería, aquí se detecta una dependencia entre los diferentes servicios en el momento de llevar a cabo la ejecución del servicio principal determinando las distintas rutas de los proveedores.

Varias investigaciones acordes al tema del IoT han vuelto tradicional el uso de las arquitecturas basadas en microservicios, lo que se alega en el artículo de (Song & Tilevich, 2019) es que una buena orquestación de los microservicios aumenta la confiabilidad y escalabilidad de un servicio, siendo esta orquestación de forma paralela o secuencial brindando un uso específico, sin embargo, es en la orquestación donde se puede volver tedioso el trabajo del programador por lo que utilizar compiladores es lo recomendable para aumentar su eficiencia y confiabilidad.

El video streaming es prácticamente una aplicación multimedia como se redacta en (Wang, Xu, & Wang, 2016) puede ser incorporada por dispositivos móviles pues con la evolución del internet y de las tecnologías de acceso se conta de redes inalámbricas como lo es el WiFi, volviendo más factible fomentar el desarrollo de la tecnología de video streaming, donde en sí es capturar un video y difundirlo en tiempo real utilizando alguna tecnología de transmisión, entrando a detalle de que existe una etapa de codificación y decodificación de datos, reproducción, transmisión y visualización, pero como en todo servicios las fallas como la presencia de latencia y pérdida de información son problemas que se pueden resolver dependiendo de la calidad de tratamiento de imagen así como de

la tecnología de acceso a disposición. Los servicios streaming son usados o implementados en el Cloud y como mencionan en el trabajo (Kim, Jang, Choi, Hwang, & Youn, 2016) el entorno de uso exige ser escalable, con alta disponibilidad informática y de almacenamiento, pues al ser datos receptados en tiempo real los recursos deben ser estables y de calidad con un costo aceptable, para lo que se ha creado modelos predictivos que son requisitos para garantizar un servicio óptimo.

Al juntar los servicios de videos streaming que ocupan gran capacidad y recursos del internet para su transmisión en tiempo real con las arquitectura Cloud los autores (Gamme, Immich, & Bittencourt, 2018) indican que los problemas de transmisión se corrigen relativamente involucrando que exista un mayor tiempo de latencia y congestión, como solución alternativa proponen una arquitectura multinivel con criterios que son aplicado en la capa fog computing a fin de presentar un streaming de alta calidad de experiencia QoE.

Los algoritmos aplicados para el reconocimiento de imágenes se han vuelto de suma importancia pues es un aplicativo de gran interés para varios tipos de industrias, estos algoritmos están basados en Machine Learning los cuales se van entrenando en medida de su funcionamiento, cabe mencionar como lo hacen en el documento (Deng, 2020) muchos factores afectan al reconocimiento como lo es la iluminación, oclusión, condiciones ambientales y hasta la misma calidad del dispositivo utilizado, por lo que la captura de las características esenciales en una imagen puede resultar insuficiente, pero el desarrollo en sí de un algoritmo para el reconocimiento de imágenes lleva tiempo en desarrollar pero en caso de ser un usuario que solamente desee utilizarlo como servicio este se encuentra disponible por varios proveedores a través de los microservicios web como los es Amazon, IBM o Microsoft, estos algoritmos ya han sido entrenados con millares de información por lo que su precisión se considera en un nivel muy elevado.

Como breve mención al machine learning en la publicación (Ray, 2019) se nos da a conocer varios de los algoritmos generalmente utilizados en diversas aplicaciones como es el reconocimiento de imágenes mediante el uso de redes neuronales, pues el campo de la inteligencia artificial cada vez es de mayor interés por la creación de algoritmos de aprendizaje automático que ya se encuentran utilizados por ejemplo como en la detección de tendencias de las personas al momento de utilizar un buscador, el machine learning propone un gran prospecto a futuro para una mejor toma de decisiones al momento de cumplir un tarea en base a un requerimiento específico pues el entrenamiento de un algoritmos ante miles de escenarios es lo que genera una respuesta predictiva de forma rápida.

Todo tipo de dispositivo móvil actualmente requiere de un servicio de mensajería para la transmisión y recepción de información con el internet, la forma de publicación de un mensaje es mediante el método push expuesto por (Yue, Ruiyang, Jianwei, & Kaifeng, 2017) ya que es la manera más rápida encontrando al protocolo MQTT que cumple este requerimiento siendo de consumo bajo y escalabilidad alta. El protocolo MQTT trabaja con Suscribe / Publisher especificando así el usuario y contraseña siempre y cuando se encuentren en una misma red, también otorga una facilidad de uso en la calidad de servicios OoS que puede modificarse en tres niveles dependiendo de la conveniencia (Sadeq, Hassan, Al-rawi, Jubair, & Aman, 2019).

Los protocolos de larga distancia se basan en tecnologías recientes requeridas para establecer comunicación en sectores que imposibiliten la instalación de redes de internet, antenas para LTE u otras, proponiendo parámetros como el funcionamiento inalámbrico, el bajo consumo, el uso de bandas libres y que cubran grandes distancias con un buen rendimiento (Darabkh & Muqat, 2018). Una tecnología destinada principalmente para el Internet de las Cosas y que utiliza un chip de espectro es LoRa low-power wide-area utilizada en la capa LoRaWAN como una red de área de largo

alcance, está constituido por dos dispositivos uno se conecta con los sensores o el sector donde se recolecta información y el otro que está alejado y hace de Gateway para transmitir esos datos a internet donde se pueden interpretar por el usuario, los dispositivos LoRa pueden medir la intensidad de señal receptada RSSI y en Ecuador se lo incorpora en la banda libre de 915 MHz (Gehadi, Shatagopam, Raghav, Sarkar, & Paolini, 2021).

Capítulo 2: Marco conceptual

Robótica

La robótica es un concepto que ha entrado en auge en los últimos años, pero no es un tema nuevo, porque se sabe que desde la antigua civilización griega se hablaba de individuos con cierta capacidad de generar movimientos a través de mecanismos que utilizaban bombas hidráulicas y poleas. Años más tarde este principio empezó a fortalecerse gracias a la civilización árabe, que le dio mayor importancia ya que les generaron mayores beneficios, especialmente aliviando el esfuerzo físico que algunas actividades conllevaban en esa época. A raíz de estas perspectivas antiguas que fueron llamando la atención de la humanidad, nace como ciencia la robótica a mediados del siglo XX.

Desde sus orígenes se fueron realizando mecanismos controlados que eran dotados de piezas móviles y engranajes, pero fue hasta la revolución industrial que el ser humano fue consciente de la importancia de estos sistemas, ya que se desarrollaron nuevos dispositivos para diferentes aplicaciones, especialmente aquellas que requerían un gran esfuerzo físico y tenían un alto grado de repetibilidad, que con el paso del tiempo se integrarían para la creación de la robótica. Fue hasta el año de 1921 donde Capek se refirió al término robot en su obra “R.U.R.”, que hacía referencia a trabajo forzado, mientras que Asimov fijo el término robótica en su obra “I, Robot” donde también menciona las tres leyes de la robótica. Desde entonces se destacó la robótica, desde la ficción hasta llamar la atención de investigadores que fueron involucrándose en su estudio y evolución.

Como consecuencia de estos antecedentes se tuvo la necesidad de generar el concepto de robot el cual se define como un conjunto de elementos tanto eléctricos como mecánicos que pueden ser programados cuantas veces sea necesario hasta que

cumplan con su objetivo de realizar tareas repetitivas, que conllevan un alto grado de precisión o esfuerzo físico (Barrientos, García, & Silva, 2017).

Robótica Móvil

En la actualidad podemos encontrar una amplia variedad de robots involucrados en todos los campos donde el ser humano puede participar o en algunos casos se han creado robots con el fin de acceder a lugares o realizar actividades que para el ser humano serían imposibles. A raíz de esta gran diversidad, nacen todo tipo de clasificaciones dentro de la robótica, una de ellas es la clasificación por su arquitectura donde se encuentran los robots móviles.

A inicios de la década de los setenta, como consecuencia de la necesidad del ser humanos de realizar operaciones de exploración en cualquier tipo de superficie, surge la investigación a gran escala de este tipo de robots, donde se buscaba especialmente dotarlo de algún tipo de inteligencia, para que pueda determinar los movimientos a realizar en base al entorno donde se encuentre. Los primeros robots de este tipo, eran dotados de ruedas, en algunos casos tres o cuatro, dependiendo de su aplicación, en donde se sustentaba su movimiento. Estos tenían la capacidad de seguir una trayectoria programada teniendo como referencia una línea en la superficie, que podrían usarse para el transporte de objetos de un lugar a otro donde la ruta ya está determinada con antelación, esta aplicación fue evolucionando hasta lograr tener un robot que podía discriminar y optimizar sus rutas.

A partir de los años noventa el campo investigativo de esta área, se enfocó principalmente en los robots cuya misión era simular el movimiento de animales de todo tipo, con esto se logró una mayor integración del entorno físico con el mundo de la robótica, ya que se necesitaba una mayor atención en los aspectos físicos y estéticos de los animales para aplicarlos mecánica y eléctricamente en los robots. De esta manera los robots móviles tuvieron aplicaciones muy amplias, desde aplicaciones de exploración en

lugares inhóspitos del planeta, hasta llegar a realizar misiones de exploraciones en el espacio exterior.

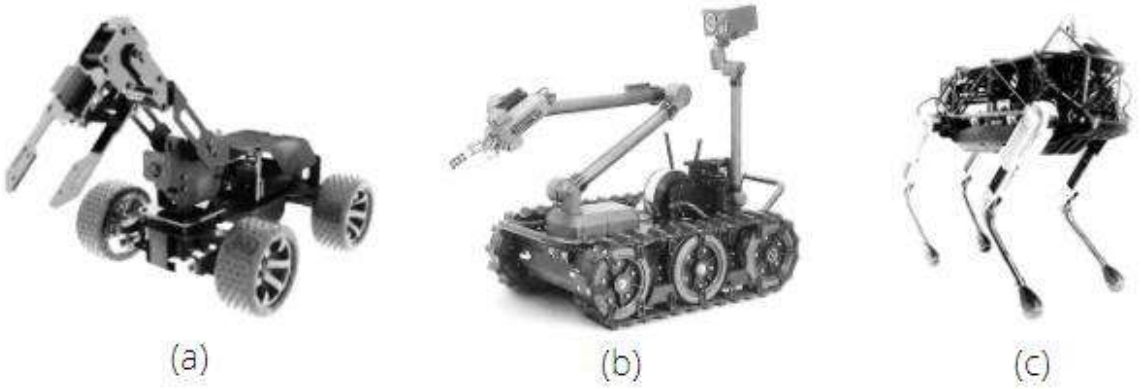
La robótica móvil en la última década se ha centrado en la exploración usando principalmente la tele operación como principio básico para este fin. Con el pasar del tiempo este campo seguirá evolucionando y con ello sus principios y conceptos con el fin de satisfacer las necesidades humanas. En la actualidad se define a la robótica móvil como el conjunto de elementos eléctricos y mecánicos, que tiene la capacidad de moverse de manera autónoma de un lugar a otro sin la necesidad de estar anclado en un solo punto, el mismo que está dotado de sensores que permiten un monitoreo continuo de su ruta y entorno, y de actuadores que son los encargados de realizar los movimientos y acciones que se requieran en su campo de exploración según corresponda. (Barrientos, García, & Silva, 2017)

Morfología de los robots móviles

Un robot móvil tiene como base fundamental su sistema de movimiento por lo que generalmente se los clasifica por el tipo de locomoción que posee, como son: robot móvil con ruedas, con patas o tipo oruga como se muestra en la figura 10.

Figura 10

Tipos de locomoción de los robots móviles

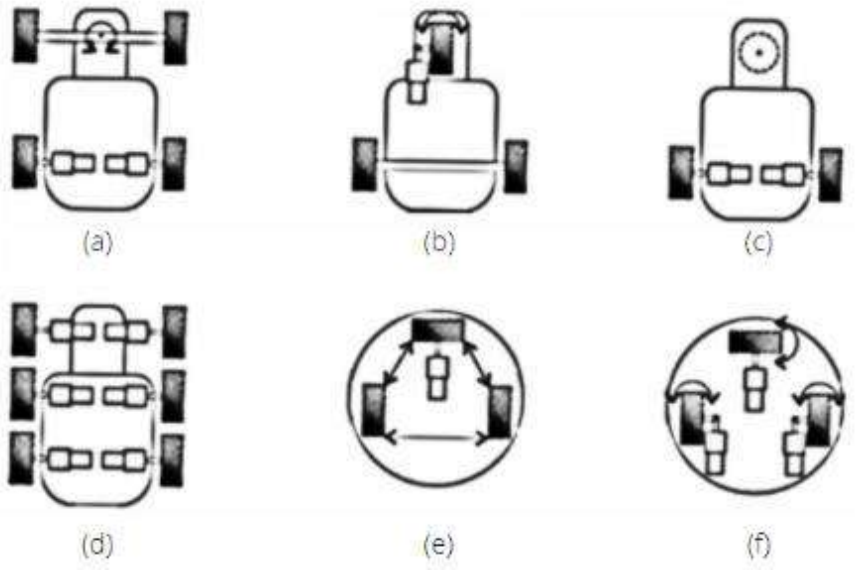


Nota: a) Robot móvil con ruedas, b) Robot tipo oruga, c) Robot con patas.

La locomoción por patas o tipo oruga ha sido un tema muy amplio de estudio, especialmente para investigadores que se han centrado en desarrollar robots cuyos movimientos son muy semejantes a los que realizan los animales, sin embargo, los robots móviles con ruedas o también denominados RMR han sido los que mayores ventajas han demostrado en cuanto a fiabilidad y eficiencia frente a los robots con patas y tipo oruga. Dentro de las principales virtudes que presentan los robots móviles con ruedas, se destaca su bajo desgaste electromecánico en aplicaciones de exploración ya que los mecanismos utilizados para el movimiento son muy inferiores en cantidad a los que utilizan los robots con patas y tipo oruga, por esta razón, tienen un evidente menor consumo de energía, especialmente en superficies firmes y lisas, lo que a su vez genera un menor desgaste del entorno donde se mueve.

Figura 11

Configuración cinemática de los RMR



Nota: En la configuración cinemática de los RMR tenemos: a) Ackerman, b) Triciclo clásico, c) Tracción diferencial, d) Skid steer, e) Tracción síncrona, f) Tracción omnidireccional.

La generación de movimiento de los RMR viene de la intervención de dos factores como son: el arreglo cinemático y el sistema de actuadores. Estos son los encargados de proporcionar movimiento a la estructura donde se encuentran fijados los demás elementos. Es importante considerar la aplicación de destino del RMR ya que de este factor depende la distribución interna de los demás elementos, con el objetivo de brindar la fiabilidad necesaria para el funcionamiento de los actuadores y principalmente la configuración cinemática del robot, que generalmente son las que se representan en la figura 11.

La configuración cinemática también está relacionada con la capacidad, en cuanto al estrés físico que deberá aguantar el robot, esto quiere decir que un robot equipado con múltiples sensores y actuadores, tendrá mayor volumen lo que generará un mayor estrés mecánico, teniendo que considerar la configuración cinemática más óptima para cada caso. En este aspecto también se debe manejar otro factor que influye de manera directa en la locomoción del robot, ya que está ligado directamente con la configuración, que es el tipo de ruedas, que se observa en la figura 12.

Figura 12

Tipos de ruedas

Nota: a) omnidireccional, b) Convencionales, c) Castor, d) De bolas.

El tipo de rueda es escogido principalmente por las razones anteriormente mencionadas, sin embargo, se consideran otros aspectos secundarios, pero no menos importantes que en su momento favorecerán o afectarán en la movilidad del robot. Tenemos tres factores secundarios, donde el primero considera que la dinámica de las partes que generan la locomoción del RMR son insignificantes y no tienen ningún tipo de flexibilidad favoreciendo a evitar desgastes innecesarios en los mecanismos rígidos que tiene el robot. El segundo factor nace a partir de la reducción máxima de la complejidad por lo que se requiere que las ruedas sean lo menos complejas posibles, en este caso una rueda que tenga no más de un eslabón, sería lo más conveniente. Por último, se debe tener una rueda que al moverse tenga una superficie de contacto grande, lo que evitará que el robot genere algún tipo de derramamiento, especialmente en superficies que son muy irregulares (Barrientos, García, & Silva, 2017).

Sensores de los robots móviles

Los sensores son la parte fundamental dentro la robótica móvil ya que son los encargados de brindar las señales del entorno para realizar la discriminación de acciones del robot. Cuando se habla de sensores, se hace referencia a otro término que es el transductor, el cual es el encargado de convertir la energía de un dominio al otro, entonces se puede decir que un sensor siempre hará uso de un transductor, posteriormente este brindará una señal que será útil para el sistema de medición. Haciendo referencia al argumento anterior se puede definir a un sensor como un dispositivo que abarca en su entrada una variable física para posteriormente brindar a la salida un dato manipulable (Corona, Abarca, & Carreño, 2014).

Sensor ultrasónico. Los sensores ultrasónicos están siendo muy utilizados ya que tiene la capacidad de evitar el contacto con la superficie de medición, bajo este principio se consideran sensores de medición no invasivos por lo que son ideales para aplicaciones de medición de alturas, profundidades y distancias que están sujetas a

cambios constantes, de igual manera son ideales para la medición de las variables antes mencionadas en cualquier tipo de superficie.

El principio del efecto Doppler radica en emitir una onda ultrasónica, la cual choca con la superficie a medir, una parte de esta es absorbida y otra reflejada, de esta manera el receptor percibe la señal y evalúa el tiempo que esta tomó en ir y volver, para posteriormente traducirla a distancia a partir de un cálculo matemático, o en algunos casos no tiene retorno, esto se discrimina como la nulidad de la superficie de contacto de enfrente. Este es el principio que usa el sensor ultrasónico para sus aplicaciones, ya que emite la señal ultrasónica que está por encima de los 20 KHz, la cual es imperceptible para el oído humano, esta rebota en la superficie a medir y es receptada en el mecanismo receptor del sensor, este valor es evaluado según la condición para obtener el dato de salida, que generalmente es un cálculo numérico (Corona, Abarca, & Carreño, 2014).

En el mercado existe una gran variedad de este tipo de sensores, enfocados a las diversas aplicaciones donde son codiciados. Para aplicaciones sencillas donde las condiciones de trabajo son normales y no se requiere de un gran desempeño se puede usar el sensor HC-SR04, sensor ultrasónico de distancia de bajo costo que mide distancias dentro de un rango de 2cm a 450cm y funciona a 5VDC.

Figura 13

Sensor ultrasónico HC-SR04

Nota: Figura tomada de (Mechatronics, 2021).

Acelerómetro. El acelerómetro es un sensor que se utiliza en diversas aplicaciones donde se requiere determinar la inclinación y vibración de algún elemento, donde el objeto no cambia de posición, caso contrario este sensor puede determinar la aceleración traslacional que le toma al objeto ir de un lugar a otro. Por su principio de transducción pueden clasificarse en capacitivos, piezoeléctricos o piezoresistivos, independiente del tipo de principio que se utilice, este tipo de sensores tendrán siempre una salida lineal, es decir que cualquier cambio en la entrada es proporcional en la salida, esto se debe a que este tipo de sensores usan básicamente la segunda ley de Newton y la Ley de Hook (Corona, Abarca, & Carreño, 2014).

En el mercado podemos encontrar una gran variedad de modelos que cuentan con la capacidad de decodificar los datos de un acelerómetro para poder trabajarlos en el sistema donde se desee emplear, desde modelos sencillos de utilizar hasta los más complejos o de gama alta, que son necesarios para aplicaciones críticas como determinación de choques, o monitoreo de aceleraciones de vehículos y aeronaves. El sensor MPU6050 es un módulo que integra un acelerómetro y un giroscopio que se comunica por el protocolo I2C que sirve para enviar la información que este genera en los tres ejes, tanto para el giroscopio como para el acelerómetro que integran un mismo chip que funciona a 3.3VDC, pero puede conectarse a 5VDC gracias a su convertidor incorporado.

Figura 14

Módulo MPU6050, acelerómetro y giroscopio

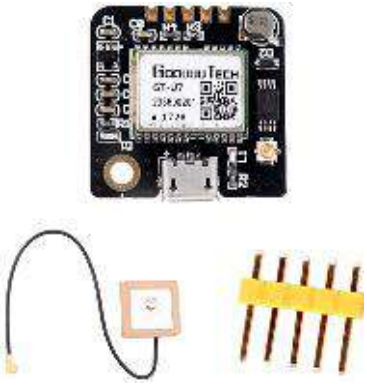
Nota: Figura tomada de (Mechatronics, 2021).

Sensor GPS. El GPS o sistema de posicionamiento global, es un sistema que proporciona una gran cantidad de datos referentes al lugar donde se encuentra, como son la latitud, longitud, altura, entre otros, así como también la fecha y hora actualizada del momento en que se está tomando la lectura del dato que entrega el sensor. Su funcionamiento en general se centra en la conexión con tres satélites que están orbitando el espacio, los mismos que forman una triangulación, de esta manera cada uno de los tres satélites forman una circunferencia que tiene como radio la posición del usuario, para posteriormente verificar la intersección de las tres circunferencias que será el punto donde se encuentra el usuario. Los datos son entregados al usuario por cadenas NMEA, que es un protocolo de comunicación de este tipo de sensores de posicionamiento, el cual trabaja a una velocidad de 9600 bps (Corona, Abarca, & Carreño, 2014).

En el mercado podemos encontrar una amplia gama de módulos GPS que radica su diferencia en el entorno a ser usado, uno de los que últimamente han salido al mercado gracias a sus grandes prestaciones en cualquier tipo de entorno es el módulo Geekstory GT-U7, que tiene un posicionamiento de alta precisión, con una alta sensibilidad y un bajo consumo de energía conectándose a 3.3VDC o 5VDC.

Figura 15

Módulo GPS GT-U7



Nota: Figura tomada de (Geekstory, 2021).

Sensores de visión. Los sensores de visión son dispositivos avanzados conocidos comúnmente como cámaras, que se encargan de obtener la información y datos del entorno a través de imágenes o videos, que posteriormente son procesadas según las necesidades del usuario. También este tipo de sensores son considerados inteligentes porque están compuestos por un grupo de sensores que en conjunto se encargan de extraer y procesar la información de modo que sea entendible para el usuario. Este tipo de sensores están en la capacidad de proporcionar a su salida imágenes o videos con diferentes características como son, a color, en escala de grises, entre otros. Para la transmisión de datos tiene varios protocolos de comunicación como USB, serial, SPI, etc. Todas las características antes mencionadas, muchas veces dependen de la marca o modelo del dispositivo que adquiera el usuario. Sin embargo, la composición de imágenes por píxeles es lo que tienen en común todos los modelos. Estos píxeles se descomponen en celdas unitarias que están formados por sensores de estado sólido con tecnología CMOS. El principio básico de este tipo de sensores se centra en la exposición a la luz, la misma que es captada y por medio del efecto fotoeléctrico, convertida en una imagen que muchas veces requieren de otro proceso para ser legibles por el usuario (Corona, Abarca, & Carreño, 2014).

Existe una variedad de cámaras para todo tipo de aplicación, donde siempre el usuario busca la facilidad de manipulación y esté acorde a la aplicación donde se la va a utilizar, por lo que es primordial analizar las características de funcionamiento. Para aplicaciones donde el procesador es una Raspberry Pi, es común usar la cámara Raspberry pi, que cuenta con todo lo necesario para conectarse directamente con la tarjeta, lo que facilita su uso, manipulación y programación.

Cámara Raspberry Pi. Es un sensor que se ha ido innovando con el pasar del tiempo, en la actualidad podemos encontrar esta cámara de segunda generación, la cual

tiene un lente de enfoque fijo que cuenta con una resolución de 3280x2464 píxeles, y pueden conectarse directamente a la tarjeta alimentada con 5V y un consumo de 2A.

Figura 16

Cámara Raspberry Pi V2, 8 megapíxeles

Nota: Figura tomada de (Geekstory, 2021).

Actuadores de los robots móviles

El actuador es un dispositivo que cuenta con la capacidad de brindar una fuerza que produce un cambio de estado, velocidad o posición en los elementos mecánicos que conforman el robot, gracias al principio de transformación de energía que se produce en este proceso. Existen variados tipos de actuadores, por lo general clasificados en dos grandes grupos que son: los actuadores por el tipo de energía que utilizan, que abarca los actuadores neumáticos, hidráulicos y eléctricos. Por otra parte, están los actuadores por el tipo de movimiento que generan, que contiene los actuadores lineales y rotatorios (Corona, Abarca, & Carreño, 2014).

Motor. La fuente de movimiento de un robot móvil puede ser de varios tipos, pero lo más común en la actualidad es encontrar motores. Gracias a los avances tecnológicos, se fabrican motores de gran variedad, pero los más utilizados en la robótica móvil son los motores DC por los beneficios que estos brindan y que se enuncia a continuación:

- El modelo de un motor DC es lineal, de esta manera simplifica el control, específicamente de aquellos motores que se componen de un imán

permanente porque el voltaje que realiza el control está conectado al circuito de armadura, mientras que el circuito de campo es excitado de forma independiente de los otros elementos del motor.

- Existen motores DC de imán permanente, con escobillas y sin escobillas, donde los que más resaltan son los que no tienen escobillas, ya que no requieren de mantenimiento ni el reemplazo de las mismas. También son utilizados en ambientes que se consideran de alto riesgo inflamable ya que no generan chispas, gracias a su nula fricción interna, lo que acarrea una consecuencia favorable como es su alta velocidad, que en algunos casos llega hasta los 50000 rpm.
- La tendencia de uso de este tipo de motores en algunos casos se inclina a favor de los motores DC con escobillas, ya que su control de giro se considera sencillo porque únicamente se debe invertir la polaridad de los pines de alimentación y su vez porque en el mercado los podemos encontrar a muy bajo costo con respecto a los motores DC sin escobillas (Galvis & Madrid, 2016).

Figura 17

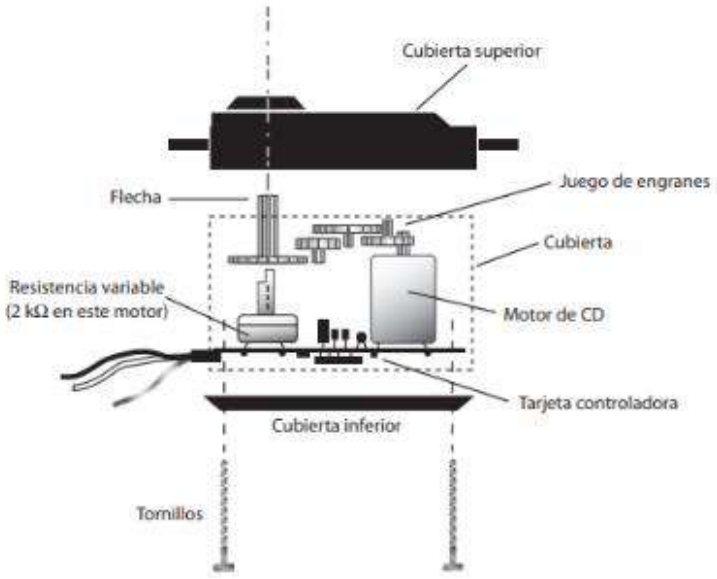
Motores DC

Nota: a) Motor DC con escobillas, b) Motor DC sin escobillas.

Servomotor. El servomotor es uno de los dispositivos que se usa con mayor frecuencia dentro de la robótica, especialmente en el campo de los brazos robóticos e integración de sistemas. Básicamente es un motor que cuenta con un reductor de velocidad que a la vez ayuda a que tenga una mayor fuerza, añadiendo a esto un circuito de control. Todos estos elementos están encapsulados dentro de un armazón que lo hace más ergonómico. Generalmente los servomotores cuentan con la capacidad de giro sobre el eje de 180º, pero también existen los que tienen un giro completo de 360º, este factor dependerá del usuario acorde a la aplicación donde se lo utilizará. El movimiento que realiza un servomotor se lo mide en grados, por lo que las señales de control para su movimiento deben aplicarse en pulsos que tendrán una cierta duración y frecuencia específica. Por lo general estos actuadores cuentan con tres cables, donde dos corresponde a la alimentación en DC y el restante es por donde se suministra la señal de control para su desplazamiento (Corona, Abarca, & Carreño, 2014).

Figura 18

Componentes de un servomotor



Nota: Figura tomada de (Corona, Abarca, & Carreño, 2014).

Control y monitoreo de los robots móviles

Los robots móviles son dotados de altas capacidades de exploración, pero muy pocas veces se estudia a fondo el entorno donde recorrerá, por lo que se debe considerar los factores más importantes de los RMR como son el posicionamiento o la geolocalización, la capacidad de recorrido en entornos difíciles con una alta eficiencia en la evasión de obstáculos y el seguimiento de trayectorias.

Referente al posicionamiento se han realizado análisis que se centran en el estudio de los sensores ultrasónicos para este fin. La eficiencia de este principio radica en el método de la implementación de estos sensores que tendrán la capacidad de estudiar el ambiente de trabajo dando como lugar el conocimiento de la posición dentro de un entorno. En algunos casos también se puede abordar el problema de la estimación del posicionamiento desde esta metodología, ya que genera ciertos errores que al cuantificarlos se puede estimar la posición considerando los tipos sistemáticos de los errores de edometría. El Lidar es otro tipo de sensor usado en menor volumen como consecuencia de su elevado costo, pero que a diferencia del ultrasónico se puede obtener un mejor análisis de posicionamiento, ya que cuenta con la capacidad de realizar un mapeo dentro del ambiente, ya sea abierto o cerrado, el mismo que puede ser bidimensional o topológico, brindando una mayor eficiencia y precisión en el desplazamiento del RMR.

En lo que respecta a la evasión de obstáculos tenemos una variada lista de métodos para lograr este objetivo, pero los que han tenido mayor relevancia por las ventajas que brindan son:

- La evasión de obstáculos por detección de borde el cual es un algoritmo que le permite al robot detectar los bordes verticales de un obstáculo con lo que podrá evitar esa trayectoria.

- La evasión de obstáculos por descomposición de celdas, donde el sistema que dota de inteligencia al robot realiza un mapeo del entorno y posteriormente la divide en pequeñas celdas, donde cada una de ellas se clasifica en accesible o no accesible, donde las accesibles son las que no tienen obstáculo mientras que las no accesibles son las que tiene algún obstáculo, por lo que el robot no podrá pasar por la misma.
- El método por construcción de mapas, donde se utiliza un agente externo que puede ser algún tipo de servicio web, que teniendo como datos la posición inicial y final, este genera la ruta bidimensional que no cuenta con obstáculos, para posteriormente cargarla en el robot y este pueda seguirla.

Por su parte el seguimiento de trayectorias, está ligado directamente al controlador que posee el robot, ya que este generará los lazos de control necesarios para tal fin. Lo más común es encontrar una estrategia de control que está centrada en dos niveles, donde en el primer nivel se tiene un lazo de control PID interno que está ligado a los motores del robot, mientras que en el segundo nivel se tiene un lazo de control externo ligado al modelo cinemáticos del RMR que brindara la velocidad adecuada para la trayectoria independientemente en cada motor donde la principal consideración es que la inductancia de cada motor es igual a cero. (Barrientos, García, & Silva, 2017).

Modelos matemáticos del robot móvil

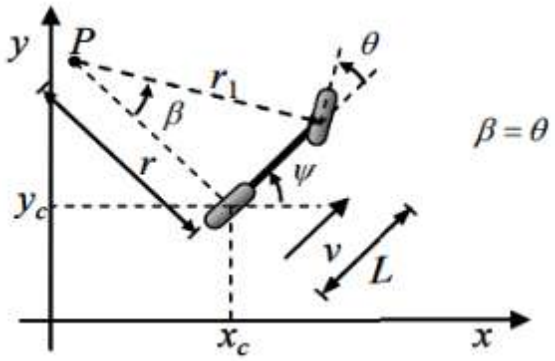
El modelo matemático de un robot es la forma de representar a través de fórmulas matemáticas, la relación que existe entre las diferentes variables con las que el robot cuenta. Esto sirve para analizar el vínculo entre dos o más variables con las que cuenta el robot y de esta manera entender los fenómenos que se producen cuando están involucradas estas variables. Existen modelos matemáticos para todo tipo de robots

móviles, pero únicamente nos centraremos en el modelo matemático del robot Ackerman, siendo el caso de estudio de este trabajo.

Modelo cinemático cartesiano. El modelo matemático del robot Ackerman puede simplificarse a un modelo tipo bicicleta como se puede observar en la figura 19, donde radica la importancia en el comportamiento de la rueda trasera, la cual tiene coordinadas en "x_c, y_c" y una orientación "ψ" que es el ángulo de orientación con respecto al eje "x".

Figura 19

Representación del robot Ackerman, simplificado en el modelo tipo bicicleta



Nota: Figura tomada de (Teodovich Sosa & Carelli, 2008).

La rueda trasera genera una velocidad lineal "v" y una distancia del eje posterior con respecto al eje frontal "L". Un robot tipo Ackerman cambia de dirección cuando la orientación de la rueda frontal cambia un ángulo "θ", que esta en un rango de $-\frac{\pi}{4} \leq \theta \leq \pi/4$.

Considerando estos detalles podemos representar la cinemática del robot Ackerman de forma simplificada en coordenadas cartesianas de la siguiente manera:

$$\begin{aligned} \dot{x_c} &= v * \cos(\psi)\,, & -\pi < \psi \leq \pi \\ \dot{y_c} &= v * \sin(\psi)\,, & |v| \leq v_m \\ \dot{\psi} &= \frac{v}{L}\tan(\theta) \quad , & -\theta_m \leq \theta \leq \theta_m, \theta_m = \frac{\pi}{4} \end{aligned} \tag{1}$$

Donde v_m representa a la velocidad máxima de la rueda posteriores, las variables de estado del sistema están dadas por $[x_c \quad y_c \quad \psi]^T$ y las variables de entrada corresponde a $[v \quad \theta]^T$ que hace referencia a la velocidad lineal del vehículo y al ángulo de la rueda frontal que da la dirección respectivamente.

También se pueden deducir las siguientes relaciones de la figura 19.

$$r_1 * \sin(\theta) = L \quad (2)$$

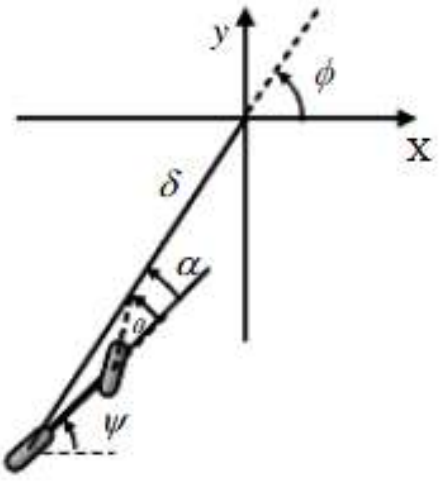
$$r * \tan(\theta) = L => \dot{\psi} = \frac{v}{r}$$

Donde r representa la distancia que existe entre el centro de rotación del vehículo "P" y el punto que se obtiene de las coordenadas "x_c, y_c" que representa el punto odométrico (Teodovich Sosa & Carelli, 2008).

Modelo cinemático polar. Debido a las condiciones de estabilidad en un punto de equilibrio del sistema que también se le conoce como condición de Brockett's, se tiene ciertas limitaciones en la representación cartesiana por lo que es preciso la representación de la posición del modelo Ackerman en coordenadas polares.

Figura 20

Representación del modelo del robot Ackerman, en coordenadas polares



Nota: Figura tomada de (Teodovich Sosa & Carelli, 2008).

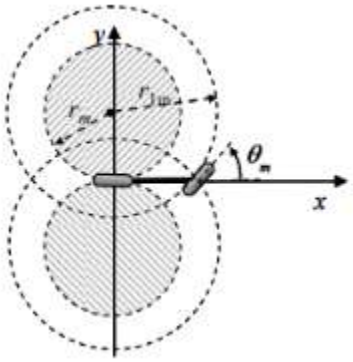
El esquema del sistema polar del modelo reducido de un robot Ackerman se representa en la figura 20 donde "δ" representa al vector de error que está orientado "ϕ" con respecto al origen. Por otra parte, se tiene el ángulo "α" que está comprendido entre el vector de distancia "δ" y el eje principal del vehículo. Considerando todos los términos mencionados, este modelo puede representarse de la siguiente manera:

$$\begin{aligned} \dot{\delta} &= -v * \cos(\alpha) && , \quad -\theta_m \leq \theta \leq \theta_m \\ \dot{\alpha} &= -\frac{v}{L} * \tan(\theta) + \frac{v}{\delta} * \sin(\alpha) , && -\pi < \alpha \leq \pi \\ \dot{\phi} &= \frac{v}{\delta} \sin(\alpha) && , \quad -\pi < \emptyset \leq \pi \end{aligned} \tag{3}$$

Mínimo radio de giro. El modelo del robot Ackerman, se caracteriza básicamente por tener en la parte frontal dos ruedas fijadas a un eje móvil para proporcionar la dirección, lo que genera una restricción para el recorrido del espacio de trabajo del robot, ya que el ángulo de giro no puede orientarse en cualquier dirección.

Figura 21

Representación del mínimo radio de giro, con la respectiva región no alcanzable



Nota: Figura tomada de (Teodovich Sosa & Carelli, 2008).

Cuando el modelo del robot Ackerman quiere cambiar de dirección, ya sea a izquierda o derecha, giran las ruedas delanteras, por lo que al avanzar formarán dos círculos respectivamente que representan la región no alcanzable por la estructura del robot. Este

recorrido forma dos circunferencias, la circunferencia externa de radio "r_{1m}", que describe el recorrido de la rueda frontal, mientras que la circunferencia pequeña de radio "r_1" representa el área no alcanzable.

A continuación, se muestran las ecuaciones que representan los radios descritos anteriormente:

$$r_m = \frac{L}{\tan(\theta_m)}, \quad r_{1m} = \frac{L}{\sin(\theta_m)}. \quad (4)$$

Controlador de los robots móviles

EL controlador es el elemento fundamental que constituye la arquitectura del robot, el cual está encargado de procesar la información que es captada por los múltiples sensores que componen el sistema de monitoreo del robot, que según el algoritmo de control que esté programado en el mismo, regulará las acciones que ejecutarán los actuadores que componen el robot (BIOLOID, 2021).

Por lo general, los fabricantes de robots, son los encargados de crear los respectivos controladores, en base a las características de sus robots, pero por otra parte existe una amplia gama de controladores que el usuario podrá adaptar a cualquier tipo de robot, teniendo en cuenta los protocolos de comunicación para la transmisión de datos. Existen controladores que se adaptan fácilmente a entornos industriales, como es el caso de los controladores de los brazos robóticos, o controladores basados en tarjetas de adquisición que son los que se analizarán a continuación ya que están orientados a aplicaciones con un volumen bajo de sensores y actuadores, o aplicaciones didácticas orientadas a la investigación. Entre estos últimos tenemos Arduino, Raspberry Pi, entre otros.

Raspberry Pi. La Raspberry Pi es una minicomputadora, que fue creada con la convicción de ser usada en cualquier tipo de aplicación en todo el mundo. Su diseño y desarrollo se realizó en los laboratorios de computación de la Universidad de Cambridge

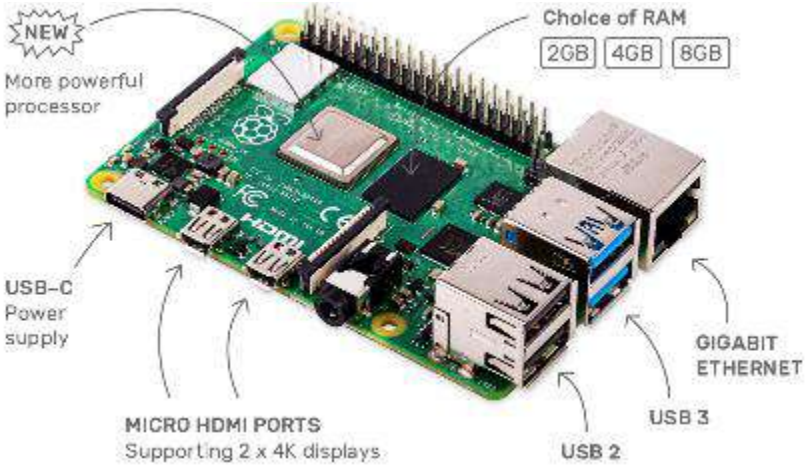
para posteriormente ser lanzada al mercado en el año 2012 por la Fundación Raspberry Pi con el objetivo de revolucionar la interacción entre computadoras y niños de todas las edades gracias a su fácil manejo. En la actualidad, esta minicomputadora se la puede adquirir a un bajo costo y varias entidades como Google están promocionando su uso, con lo que numerosos sitios web han presentado proyectos en código abierto basados en Pi.

La Raspberry Pi, que se muestra a continuación, cuenta con un potente procesador ARM además de un procesador de gráficos 3D capaz de generar y procesar videos de alta definición, 2GB, 4GB o 8GB de RAM y un conector de tarjeta de memoria SD para los archivos y contenido. Para su interacción con el usuario, cuenta con un conector HDMI que funciona con cualquier tipo de monitor que cuente con conexión HDMI, además de un par de conectores USB para el mouse y teclado, incluyendo su respectivo conector Ethernet. La tarjeta también está en la capacidad de manejar varios GPIO E/S de propósito general que le permiten comunicarse con sensores y actuadores para aplicaciones de integración. (Brock, Bruce, & Cameron, 2013)

El software de la Raspberry Pi, tiene como núcleo principal su sistema operativo que puede ser Linux, o Raspbian, que fue el último en ser lanzado pero que ha tenido la mayor aceptación gracias a sus grandes prestaciones, especialmente por su interfaz de fácil manejo. Dentro de este sistema operativo se pueden instalar diversas aplicaciones para satisfacer las necesidades de los proyectos, pero el que no puede faltar es el editor de código de Python para la realización de programas basados en este lenguaje, que, dentro de la Raspberry Pi, resulta sencillo manejar gracias a la cantidad de código abierto que existe para todo tipo de aplicaciones que se deseen realizar y para cualquier usuario que tenga acceso al internet. Así mismo, esta empresa brinda la posibilidad de adquirir periféricos tanto de entrada como de salida, para poder complementar la arquitectura de esta tarjeta de desarrollo.

Figura 22

Raspberry Pi 2021



Nota: Figura tomada de (Pi, 2021)

Python. Python es un lenguaje de programación que fue creado a inicios de los años noventa por Guido van Rossum, cuando se encontraba en el proceso de diseño e implementación de un nuevo sistema operativo llamado Amoeba. En sus inicios Python tenía el deber de controlar las excepciones y generar interfaces para el desarrollo de programación dentro de Amoeba, que en ese entonces funcionaría como sucesor del lenguaje ABC.

Después de varios avances en el estudio de este lenguaje de programación, que en un principio funcionaba para otros fines, el 16 de octubre del año 2000 se procede a lanzar Python 2.0 que fue creado básicamente con el fin de recolectar basura y generar un soporte completo para Unicode; por otra parte, la comunidad de desarrollo interesada en este nuevo lenguaje fue la encargada de darle un mayor realce y desarrollo. Años más tarde, específicamente en el año 2008 sale públicamente Python 3.0 que contaría con una amplia gama de funcionalidades dentro de la programación, lo que llamaría la atención de los programadores para poder crear sus proyectos. Python cuenta con varias

características que lo hace sobresalir sobre otros lenguajes de programación como se enuncia a continuación:

- La sintaxis de Python es relativamente fácil, ya que no necesita de complejas invocaciones como se realiza en otros lenguajes de programación, tanto que en ocasiones se podría considerar esta sintaxis como pseudocódigo.
- La consola de Python permite probar programas de alta capacidad de procesamiento sin tener que crear otras extensiones como se realizaría en otros lenguajes de programación.
- Python cuenta con una de las librerías más completas ya que de esta manera permite a los usuarios suplir la mayoría de necesidades que surgen dentro de la programación. Su fortaleza se encuentra en su código abierto y de fácil descarga para la implementación dentro de los programas.
- Python relaciona su librería con el lenguaje C para tener una mayor robustez acoplándose fácilmente a la amplia gama de editores de código existentes, es decir puede involucrarse fácilmente con otro tipo de lenguaje con el uso de la librería correspondiente.
- Es un lenguaje de programación que cuenta con una alta capacidad de extensibilidad lo que ayuda a reutilizar el código que comúnmente es escrito en C, de esta manera las limitaciones se reducen al máximo en la interlocución de diferentes tipos de programas.
- Python es software libre, posee Python License y está certificada por el movimiento Open Source para que los usuarios sean los encargados de dar el realce y avances que el código ha llegado a tener hoy en día (Challenger, Días, & Becerra, 2014).

Arduino. Arduino tiene sus orígenes en el Instituto de Diseño Interactivo de Ivrea, Italia, en el año 2005, que aparece bajo la necesidad de tener un dispositivo que ayude a los estudiantes en su aprendizaje, que a la vez sea de bajo coste y se adapte a un computador que maneje cualquier sistema operativo. Sin embargo, el mismo año, el instituto se vio obligado a cerrar por razones internas, por lo que tomaron la decisión de liberar a la comunidad todo el trabajo desarrollado hasta la fecha, para no perderlo, y de esta manera beneficiarse de las mejoras y sugerencias de los usuarios para poder mantenerlo “vivo”. Este es el principio que se ha manejado hasta la fecha, con lo que ha logrado innovarse, a tal magnitud de tener usuarios a nivel mundial que día a día aportan tanto a nivel de software como de hardware.

Para entender el principio de Arduino, inicialmente debemos conocer sobre los microcontroladores. El microcontrolador es un conjunto de componentes electrónicos que se integran dentro de un solo encapsulado para en conjunto formar un dispositivo electrónico, por lo que se lo conoce comúnmente como circuito integrado, que tiene la cualidad de ser programable, en otras palabras, es capaz de ejecutar un conjunto de instrucciones definidas con antelación por el usuario, de forma autónoma. Básicamente el microcontrolador cuenta con tres elementos fundamentales, la unidad central de proceso que se encarga de la ejecución de instrucciones del programa, los diferentes tipos de memoria que almacenan datos e instrucciones y por último los pines de entrada/salida que tienen la función de comunicar al microcontrolador con la funcionalidad exterior.

Entendido esto, se puede definir a Arduino como una placa de hardware libre, que tiene un microcontrolador programable como elemento principal, complementada con pines de entrada/salida que comúnmente son de tipo hembra para facilitar la conexión con los elementos del exterior que pueden ser sensores y actuadores. El software de Arduino es de libre acceso, lo que quiere decir que es gratis, libre y de multiplataforma,

básicamente se caracteriza por ser didáctico e intuitivo, lo que conlleva una fácil creación de programas basados en lenguaje C desde su interfaz, para posteriormente se cargarlos en la tarjeta Arduino. En referencia al hardware, hasta la actualidad existe una amplia variedad de placas Arduino oficiales, que sus características difieren principalmente del número de pines entrada/salida, así como de su tamaño físico, y modelo del microcontrolador. Existen otras placas que no son oficiales de Arduino, pero están adaptadas, eso quiere decir que son compatibles con Arduino tanto en software como en hardware, este es el caso de la ESP32, que es una tarjeta adaptada con características especiales como se enuncian más adelante (Torrente Artero, 2013).

ESP32. El ESP32 es un SoC por las siglas en inglés, System on Chip, que cuenta con módulos de comunicación incluidos como: Wifi, Bluetooth, LoRa, entre otros. Estas tarjetas se caracterizan por ser de bajo consumo de energía y bajo costo, a la vez el hardware está dedicado a integrar todo el sistema en un solo chip. Comúnmente el ESP32 realiza las operaciones del sistema en tiempo real, operaciones que dependen de la programación cargada en la tarjeta, la misma que se puede hacer de varias maneras, donde las más destacadas son:

- Formulario completo, donde se introduce el código en lenguaje C a la tarjeta, a través del SDK.
- Formulario introductorio o de aprendizaje que utiliza el IDE de Arduino para introducir el código en lenguaje C o C++.

Como se menciona anteriormente, el ESP32 tiene incorporado varios módulos de comunicación, pero vale la pena resaltar el módulo LoRa, el cual tiene un mayor realce para aplicaciones orientadas al IoT que a la vez también puede crear arquitecturas LoRa WAN para establecer enlaces de comunicación a largas distancias, que es el caso de estudio de este trabajo (Bertoleti, 2019).

Figura 23

ESP32 Heltec Wireless Stick Lite

Nota: Figura tomada de (HELTEC, 2021)

LoRa. LoRa proviene del inglés Long Range, es una tecnología de radiofrecuencia que permite realizar comunicaciones a larga distancia, específicamente, de unos pocos kilómetros, sin la necesidad de muchos recursos y con un bajo consumo de energía. La frecuencia que utiliza la tecnología LoRa para la transmisión de datos varía dependiendo de la región del planeta donde se encuentre el usuario como se puede observar en la tabla 4.

Tabla 4

Bandas de frecuencia licenciadas por LoRa en las principales regiones del planeta.

Región	Banda
Estados Unidos y Américas	Desde 902Mhz hasta 928Mhz
Europa	Desde 863Mhz hasta 870Mhz
China	Desde 779Mhz hasta 787Mhz

Nota: Se presenta la región con su respectiva banda de frecuencia (Bertoleti, 2019).

Esta tecnología puede utilizarse en forma de red, que funciona en la topología punto a punto o estrella, dependiendo de la cantidad de puntos LoRa que se tengan. En la comunicación LoRa no hay un direccionamiento de red, porque la información que un dispositivo transmite, puede llegar a varios puntos implicados dentro de la Red LoRa, por lo que se detalla a continuación los diferentes tipos de flujo de datos que permite LoRa.

- Simplex: Un solo punto transmite los datos a otros N puntos que están dentro de la red configurados como receptores.
- Half dúplex: Todos los puntos de la red pueden transmitir o recibir datos, pero nunca pueden hacer las dos cosas simultáneamente.
- Full dúplex: Todos los puntos de la red pueden transmitir o recibir datos, y pueden hacer las dos cosas simultáneamente.

LoRa tiene grandes prestaciones, lo que ayuda en aplicaciones robustas que requieran transmisión de datos a bajo costo y de largo alcance, algunas de sus ventajas se detallan a continuación.

- LoRa puede tener un gran alcance con un bajo consumo de energía a comparación de otros dispositivos que están desarrollados para realizar la misma función.
- Al operar en bandas diferentes a la gran mayoría de dispositivos de radiofrecuencia, LoRa tiene una gran inmunidad a las interferencias, que es algo fundamental en aplicaciones, especialmente con entornos urbanos.
- En comparación con otros dispositivos que puedan dar solución a aplicaciones que también puedan ser solucionadas con LoRa, la diferencia de costos es muy amplia, ya que LoRa ha crecido a tal punto de tener en el mercado una gran variedad de dispositivos que incorporan esta tecnología, como consecuencia se a echo muy económico.

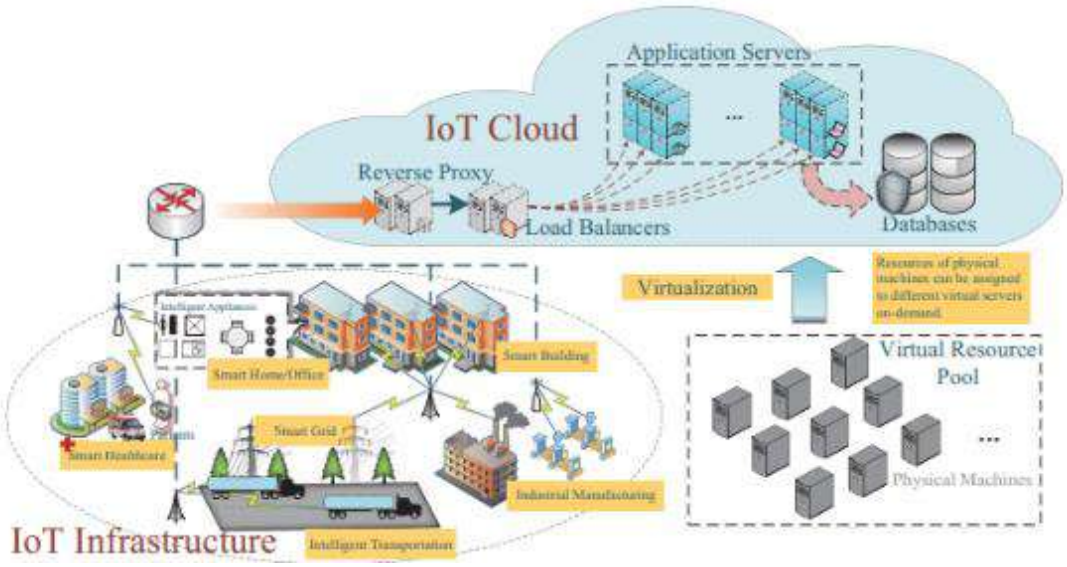
En conclusión, LoRa significa tecnología y enlace de radio. Otro término que nos encontraremos en el estudio de esta tecnología es LoRaWAN, que, a diferencia de LoRa, es un protocolo de red que está definido en software y que utiliza LoRa como su canal de transmisión, es decir, la capa física. (Bertoleti, 2019)

Arquitectura Cloud IoT

La arquitectura Cloud IoT es un sistema que está conformado por diferentes módulos o capas que se organizan con el objetivo de realizar el procesamiento de datos desde el mundo físico hasta llegar a la computación en la nube. Una arquitectura IoT robusta nos facilita en gran medida la transmisión de datos a través de los servicios y nuevas tecnologías que brinda el Cloud IoT. Este tipo de arquitectura puede admitir una gran cantidad de dispositivos IoT, lo que conlleva una infraestructura que no solo está basada en software, sino que también incluye el soporte de hardware a gran escala, lo que conlleva a una virtualización de estos recursos. En consecuencia, se hace necesario emplear servidores, que comúnmente son MQTT y HTTP, ya que los MQTT garantizan una alta cantidad de conexiones en tiempo real entre varios dispositivos, mientras que los HTTP proporcionarán los servicios a los dispositivos y usuarios finales, adicional a esto la arquitectura necesita de bases de datos, que son el soporte para brindar una alta funcionalidad, rendimiento y disponibilidad al sistema (Hou, Zhao, Xiong, & Zheng, 2016).

Figura 24

Arquitectura Cloud IoT



Nota: Figura tomada de (Hou, Zhao, Xiong, & Zheng, 2016)

Características esenciales

Infraestructura IoT. Es el diseño o la forma inicial de todo el sistema de IoT, ya que se deben considerar todos los argumentos para que se pueda detectar y realizar acciones desde y hacia el medio ambiente, así como el envío de información a la nube. La infraestructura de IoT está conformada por todos los dispositivos orientados al internet de las cosas y las redes de acceso de apoyo. El primero se implementa en la aplicación del entorno, mientras que este último proporciona comunicaciones entre los dispositivos de IoT y la nube. Los sensores, actuadores, dispositivos inteligentes, entre otros, son los que conforman básicamente la arquitectura Cloud IoT, y pueden generar grandes cantidades de datos que se transmiten a la nube, a través de redes de acceso fiables y eficientes. Además, los mensajes de control se pueden transferir a los dispositivos de IoT desde la nube a través de las mismas redes de acceso.

Cloud IoT. Como podemos observar en la figura 24, la nube consta de varios componentes, cada uno de los cuales está compuesto de múltiples servidores que realizan diferentes funciones. Los servidores están establecidos como máquinas virtuales (VM) utilizando tecnología de virtualización. Son independientes entre sí incluso si se ejecutan en la misma máquina física. Con estas máquinas virtuales, reguladores de carga, servidores proxy inversos, bases de datos y servidores de aplicaciones, se puede configurar la arquitectura Cloud IoT. Las funcionalidades de cada componente se describen a continuación:

- **Máquinas virtuales:** En este tipo de arquitecturas, los recursos de hardware de la máquina física (como la CPU, la memoria y conectividad de red) no se puede utilizar completamente, por lo tanto, hay un sobredimensionamiento de recursos, así como el problema de la baja escalabilidad de los servidores. Por esta razón se utiliza la técnica de

virtualización que proporciona soluciones viables que tienen como objetivo mejorar la utilización de recursos para la nube de IoT. Los servicios en la nube pueden implementarse en máquinas virtuales en lugar de directamente en máquinas físicas, y de esta manera se puede reducir el uso de máquinas y, por lo tanto, puede ofrecer un alto rendimiento a bajo costo.

- **Servidores de aplicaciones:** Los servidores de aplicaciones son comúnmente considerados el componente más importante de la nube, ya que son responsables de ofrecer servicios comerciales a los clientes. Son necesarios para proporcionar un entorno apropiado para ejecutar múltiples aplicaciones basadas en ciertos protocolos de aplicación. Los servidores de aplicaciones en la nube generalmente se basan en HTTP. Los servidores HTTP funcionan de manera de solicitud-respuesta a través de las conexiones TCP con los clientes. Cuando se establecen las conexiones, un servidor HTTP puede escuchar en ciertos puertos las solicitudes de los clientes y enviar las respuestas adecuadas a las solicitudes recibidas. Sin embargo, HTTP no es adecuado para la nube de una arquitectura IoT, ya que los dispositivos de esta arquitectura están limitados por sus recursos informáticos, de comunicación y energéticos. En consecuencia, otro tipo de protocolo de aplicación es más atractivo para la nube de IoT, como lo es el protocolo MQTT. MQTT está diseñado para dispositivos IoT con recursos limitados como un protocolo de transporte de mensajería liviano que opera a través de un modo de publicación-suscripción basado en temas. Esto significa que cuando un cliente publica un mensaje sobre un tema en particular, todos los clientes que se han suscrito al mismo tema pueden recibir este mensaje.

- **Base de datos:** Las bases de datos relacionales y no relacionales, también conocidas como lenguaje de consulta estructurado (SQL) y bases de datos NoSQL, son opcionales en la nube de IoT. SQL está diseñado como un tipo de lenguaje de programación para bases de datos relacionales que pueden almacenar datos en forma de tablas bidimensionales. Sin embargo, el rendimiento de las bases de datos SQL es el principal cuello de botella para la implementación de aplicaciones de IoT en tiempo real. En consecuencia, las bases de datos NoSQL se utilizan para proporcionar servicios en tiempo real y de alta eficiencia para el almacenamiento de datos. Estas bases de datos permiten que los datos se almacenen directamente en la memoria o en los discos duros y, por lo tanto, la velocidad de entrada / salida se mejora significativamente.
- **Equilibrio de carga y proxy inverso:** Una arquitectura Cloud IoT maneja una gran cantidad de dispositivos y usuarios de IoT, por lo tanto, los servidores de aplicaciones deben manejar millones de solicitudes simultáneas o transferir una gran cantidad de mensajes. Estas solicitudes o mensajes se procesan sin programación, si no se aplica el equilibrio de carga. Como resultado, algunos servidores pueden estar muy congestionados debido a cargas excesivas y es posible que las nuevas solicitudes o mensajes enviados a los servidores congestionados sean rechazados o descartados. Mientras tanto, otros servidores pueden estar inactivos, aunque los recursos sobrantes pueden ser suficientes para procesar estas solicitudes, lo que resulta en un desperdicio significativo de recursos. Por lo tanto, el equilibrio de carga es imperativo para la distribución uniforme de la carga de trabajo en múltiples servidores

> backend y lograr la utilización completa de todos los recursos disponibles (Hou, Zhao, Xiong, & Zheng, 2016).

Una arquitectura Cloud IoT puede variar en distribución y cantidad de elementos dependiendo de la aplicación a la cual está enfocada, pero detrás de esta arquitectura tenemos las capas de aplicación que no van a variar, es por eso que a continuación se conceptualiza las capas que conforman la arquitectura Cloud IoT de manera generalizada.

Sistema ciberfísico. Son aquellos sistemas donde la importancia recae en la interacción junto con la dinámica entre un medio físico y un medio computacional buscando tener respuestas rápidas y fiables ante instrucciones en tiempo real, sujeto a un control de una red de sensores y actuadores. Algunos ejemplos en los que podemos encontrar utilidad para los sistemas ciber físicos son en sistemas de transporte, redes de energía, robots en medicina, robots de exploración, procesos industriales entre otros. (Villalonga, 2016).

Capa de Edge Computing

La capa de borde o también conocido como Edge Computing nace hace unas pocas décadas atrás, a raíz de la evolución de las tecnologías de la información, las arquitecturas de comunicación cliente/servidor y la información en la nube, donde notaron que el manejo de extensas cantidades de datos, generaba inconvenientes en los sistemas IoT, y en la mayoría de casos, aquellos datos eran innecesarios o poco importantes para el sistema que estaba en funcionamiento. Desde esa perspectiva, nace la necesidad de analizar los datos y depurarlos antes de ser enviados a la nube, lo que ayudaría a optimizar el sistema, tanto en procesamiento como en espacio de memoria, a esta idea se la denominaría Edge Computing.

El Edge Computing es el conjunto de tecnologías que realizan el análisis y procesamiento de datos en el borde de la red, es decir fuera de los elementos que están

conectados al internet. De esta manera cualquier elemento que está ubicado entre el usuario y la nube, es decir elementos físicos o tarjetas de adquisición programables mediante su software interno, que en muchos casos son necesarios para la toma y procesamiento de datos a baja escala son considerados parte de la capa de Edge Computing. Esta capa tiene la finalidad de aportar significativamente en la arquitectura Cloud IoT, principalmente brindando una mayor velocidad de procesamiento de los datos a nivel de aplicaciones, y un menor consumo de recursos que como consecuencia genera un menor consumo de energía (Merchan, 2020).

La capa de Edge Computing hoy en día es muy utilizada gracias a las prestaciones y grandes beneficios que aporta al diseño de arquitectura IoT, como se enuncia a continuación:

- Al utilizar la computación de borde se reducirán los tiempos de latencia debido ya que el procesamiento de datos se realiza localmente.
- En esta capa, los datos son almacenados de forma local, ya que muy pocos de estos dispositivos están conectados al Internet, lo que reduce la cantidad de datos que pueden ser filtrados, obteniendo de esta manera grandes prestaciones de privacidad y seguridad de Datos.
- Esta capa envía los datos ya analizados y procesados, es decir solo datos válidos, lo que va ayudar a la nube a descartar el almacenamiento de datos innecesarios, que se traduce en la reducción de costos asociados con el almacenamiento y procesamiento de la nube.
- La arquitectura Cloud IoT está propensa a fallos debido a varios factores de red, por lo que esta capa puede ser programada para ejecutar acciones fuera de línea, y así, los elementos que están en contacto con el usuario sigan funcionando de manera autónoma.

Capa de Fog Computing

Bajo el estudio de la capa de Edge Computing realizada anteriormente, esta capa nos llegaría a parecer igual, tanto en características como en prestaciones, pero debemos tener muy claro que el Edge se ocupa más de los objetos y dispositivos que están en contacto o son manipulados por el usuario mientras que la computación en la niebla o Fog Computing se enfoca más a la infraestructura.

Generalmente la capa de Fog Computing se define como una infraestructura donde interactúan una gran cantidad de dispositivos descentralizados y heterogéneos, con la función de comunicarse, cooperar entre ellos y con los elementos de la red, con el objetivo de ejecutar labores de procesamiento y almacenamiento, sin la participación de terceros. Esta labor está centrada en mejorar los servicios y las funciones básicas que se ejecutan en la red. Esta capa en la actualidad se enfoca principalmente en tres campos, como son: el procesamiento eficiente de big data, el control de realidad aumentada, y control de plataformas digitales de contenido multimedia.

El Fog Computing está asentado en dos pilares como son los NFV y SDN.

- Las funciones de red virtualizadas o NVF tienen el deber de sustituir los dispositivos físicos que componen la arquitectura por otros elementos virtualizados.
- La red definida en software o SDN cuenta con la capacidad de mejorar la calidad del servicio en el envío de paquetes de datos a la red.

En algunos diseños de arquitectura Cloud IoT deciden no utilizar esta capa, ya que consideran que su funcionalidad se ve solapada con la capa de Edge Computing, pero se debe tener claro que no es así, ya que esta capa principalmente esta liga a brindar las siguientes prestaciones:

- En la transmisión de datos entre la capa de Edge Computing y la de Cloud, pueden existir desfases que podría generar problemas en la calidad de los

datos almacenados, es ahí donde la capa de niebla se vuelve importante ya que protege al usuario de desfases o también conocidos como delay's.

- Debido a la importancia que se ha dado a esta capa en los últimos años, se han desarrollado módulos tanto en software como hardware que ofrecen una conectividad sencilla y adecuada.
- Para la transmisión de datos hasta la nube se necesita gran fiabilidad en el envío, por lo que esta capa ayuda a que las conexiones sean confiables y la capacidad de conexión sea adecuada.

Para la implementación de esta capa, debemos concentrarnos en que todos los nodos tengan una misma interfaz y sistema operativo similar, lo que ayuda a coordinar de mejor manera los diferentes recursos de la capa. Así mismo, es importante dimensionar la capacidad de la arquitectura de manera adecuada, ya que esta debe ser en función de los dispositivos que la componen. Y por último se debe tener una correcta gestión de los recursos de la arquitectura (Merchan, 2020).

Capa de Cloud Computing

La capa de Cloud Computing es el elemento principal de una arquitectura Cloud IoT ya que todos los datos y recursos tecnológicos tanto en software como en hardware están centrados en ella. A esta capa se la conoce como una estructura o modelo que permite el acceso a través de la red a una gran cantidad de recursos compartidos y muchos de ellos configurables, por enlistar algunos de ellos: aplicaciones, servicios, redes, servidores y capacidad de almacenamiento. Estos pueden ser seleccionados por el usuario de una manera ágil y rápida a través de la red para que el proveedor del servicio pueda liberarlos y asignarlos al usuario rápidamente con una mínima gestión.

El diseño e infraestructura de esta capa se centra en la aplicación para la cual está destinada, por lo que es muy importante considerar todas las condiciones de

funcionamiento y especialmente sus características, en la mayoría de casos esta capa cuenta con cinco características esenciales:

- Sin la necesidad de tener una interacción humana entre los múltiples proveedores de servicios Cloud, el usuario está en la capacidad de hacerse con los servicios que se encuentran disponibles en la nube de manera automática a medida que este los vaya requiriendo.
- Los usuarios hoy en día manejan una amplia variedad de dispositivos, desde portátiles hasta teléfonos móviles, por lo que, gracias a esto, pueden acceder a través de la red a los recursos que brinda el Cloud Computing.
- Los proveedores tienen la capacidad de compartir los recursos como almacenamiento, ancho de banda, máquinas virtuales, memoria, entre otros, de forma dinámica, según las peticiones y requerimientos de los usuarios.
- El Cloud Computing ha logrado que los usuarios tengan la impresión de que los recursos en la red están siempre disponibles y son ilimitados, gracias a la gran eficiencia que han demostrado los proveedores a la hora de asignar y liberar los servicios que requiere el usuario.
- Existe transparencia entre los usuario y proveedores ya que este último, tiene la capacidad de medir el servicio efectivamente entregado, de forma que los usuarios tienen un acceso transparente al consumo real de los servicios que adquirió.

Los modelos de servicio que hoy en día los proveedores promocionan a los usuarios de esta tecnología son:

- **Cloud Infrastructure as a Service:** El usuario adquiere del proveedor servicios, grandes recursos, de esta manera, está en la capacidad de crear cualquier tipo de software, desde aplicaciones hasta sistemas operativos.
- **Cloud Platform as a Service:** El usuario tiene la capacidad de desplegar aplicaciones propias, sin embargo, la infraestructura de Cloud le pertenece al proveedor, porque es el encargado de brindar las herramientas de programación y la plataforma de desarrollo.
- **Cloud Software as a Service:** El usuario no está en la capacidad de controlar la infraestructura ni la propia aplicación (con algunas excepciones) del servicio que le brinda el proveedor, únicamente tiene la facultad de manipular el servicio para su beneficio, como es el caso de los servicios webmail. (Cierco, 2011).

Capa de Front End

La arquitectura Cloud IoT, comprende dos partes fundamentales, que son el Front-End y el Back-End, dentro del Back-End están constituidas las capas anteriormente mencionadas, mientras que el Front-End funciona como cliente de la arquitectura.

La capa de Front-End es la que incluye todos los elementos con los que interactúa el usuario final. En otras palabras, es la unión de varios subcomponentes que en conjunto ofrecen la interfaz del usuario. Es considerada una de las partes fundamentales de la arquitectura ya que a través de esta capa el usuario final se conecta a la infraestructura de computación en la nube. El Front-End incluye componentes como redes locales, navegadores y aplicaciones web (Memon, Banerjee, Nguyen, & Robbins, 2015).

Los componentes más significativos la capa de Front-End se describen a continuación:

- **Interfaz de usuario:** Es aquella a la que tiene acceso el usuario final para enviar solicitudes y realizar cualquier tarea en la nube según lo establecido en el Back-End. Como referencia de algunas interfaces de usuario populares que están basadas en la nube son Gmail, Google Doc, etc.
- **Software:** La arquitectura del software en el Front-End es el software que se ejecuta en el extremo del usuario, esta arquitectura comprende principalmente las aplicaciones o navegadores del lado del cliente.
- **Dispositivo cliente:** Este dispositivo se refiere al hardware del lado del usuario final, que puede ser cualquier dispositivo de entrada o PC. En la arquitectura Cloud IoT, el dispositivo del lado del cliente no requiere una gran capacidad de procesamiento ya que la nube es la que soporta toda la carga de proceso (Pandey, 2019).

Machine Learning

Con el pasar de los años y la inmensa concentración en el conocer conductas que sigan un mismo patrón para cumplir actividades en un fin predeterminado es lo que impulsa a la tecnología a recopilar una gran cantidad de datos que vuelva factible el predecir y automatizar actividades mediante algoritmos de aprendizaje, esto es lo que se conoce como Machine Learning aplicable a numerosas áreas como la mecánica, finanzas, robótica entre otras (Hinestroza, 2018). Estos algoritmos posibilitan que las máquinas estén en la capacidad de entrenarse en base a registros anteriores que ayuden a determinar un patrón clave para encontrar soluciones a problemas (The Royal Society, 2017).

Los programas comienzan a volverse autónomos mientras mayor sea el entrenamiento gracias a modelos de programación que pueden encontrarse en cámaras, buscadores de información, bases de datos, en fin, toda actividad que es cotidiana en la vida de las personas (González García, 2018).

Tipos de aprendizaje

Con la cantidad adecuada de información para poder entrenar a los diferentes modelos cada programa es capaz de interpretar cada escenario o evento y establecer una respuesta adecuado. Según (Sandoval, 2018) se tiene dos tipos de aprendizaje el supervisado y no supervisado.

Supervisado. El aprendizaje de la máquina depende tanto de la entrada y de la salida del sistema, es decir, que se tiene que entregar las dudas, inquietudes o preguntas en la entrada y también los resultados a su salida para que luego del entrenamiento la máquina sea capaz de predecir la salida ante una entrada determinada (Sandoval, 2018). A su vez el aprendizaje supervisado consiste en *clasificación* cuando la respuesta sólo puede tener dos únicos valores en otras palabras valores discretos y *regresión* que se espera a la salida un valor numérico o valor continuo (Leivi, 2019).

No supervisado. A diferencia del anterior el algoritmo es entrenado únicamente con las entradas del sistema y lo que se espera es la agrupación de estas entradas en base a criterios de semejanza entre las mismas (Leivi, 2019).

Reforzado. Este tipo de aprendizaje lleva al sistema a un entorno en el cual no va a tener dependencia de una entrada o salida determinada previamente, sino más bien el sistema es entrenado al resolver problemas en base a consecuencia de decisiones lo que permite encontrar la mejor ruta para solventar un cuestionamiento (Rojas, 2020).

Algoritmos de aprendizaje

Los algoritmos de Machine Learning son varios y depende de la aplicación que se vaya a tener para poder implementarlos y especificarlos, a pesar de ello existen los principales o los algoritmos bases de los cuales se derivan el resto o se toman como referencia al ser consideradas ya como técnicas o modelos principales para la toma de información (Rojas, 2020). Se tienen los siguientes algoritmos o modelos:

Deep Learning. En búsqueda de facilidad de aprendizaje para mejora del entrenamiento se presentan algoritmos que sean capaces de figurar o reproducir aquellos eventos o escenarios indeterminados (Bengio, Goodfellow, & Courville, 2015). Active Learning. Estos algoritmos pertenecen al grupo del tipo de aprendizaje no supervisado y es aquel que tiene la capacidad de realizar consultas o interacciones con otros medios de información para poder encontrar la solución a una tarea o problema (Settles, 2009).

Support Vector Machines. Se acoge un método en el cual las muestras o datos recolectados sobre un problema planteado pueden ser ubicados de forma lineal siguiendo un patrón (Burges, 1998).

Modelos Lineales. Están comprendidos por algunos algoritmos frecuentemente escuchados como el de logística y el método de mínimos cuadrados, son algoritmos sencillos de aplicar pues su objetivo es linealizar o encontrar un ajuste adecuado al seguimiento de los datos (Sandoval, 2018).

Algoritmos o modelos de árbol. Estos modelos tienen mayor eficiencia que los anteriores en donde su estructura se conforma como su nombre lo dice en forma de árbol, es decir se tiene una ramificación de varias posibilidades y no de forma lineal (Sandoval, 2018). Como ejemplo está el algoritmo árbol de decisión.

Redes Neuronales. De gran influencia y aplicación actual por su tratamiento de información por su estructura basado en las neuronas del cerebro humano que permite una inmensa interconectividad entre las entradas y salidas del sistema, es aplicado al reconocimiento de imágenes debido a la dificultad que conlleva y el alto grado de precisión que se requiere para el tratamiento de todos los pixeles de una imagen (Negrete Fernández, 2021).

APIs Web para machine learning

Actualmente existen muchos proveedores de APIs para Machine learning que ofertan servicios para diversos aplicativos en donde el usuario está en la capacidad de elección ya sea por afinidad, facilidad de manejo u oferta de procesos complejos, en fin, según (Corral Plaza, Resinas, & Boubeta-Puig, 2018) algunas de las opciones más populares son BigML Amazon Machine Learning, Microsoft Azure Machine Learning y Google Cloud Machine Learnin Engine.

Una comparativa entre estos proveedores representado en la tabla 5 permite conocer cuáles son las diferencias entre las APIs y cuál es la mejor aplicable a proyecto de investigación desde el punto de vista de un estudiante universitario.

Tabla 5

Comparativa APIs Web con mayor renombre.

APIs Web	Plan de Suscrpción Básico	Tamaño fuentes de datos	Predicciones	Trabaja con formato JSON
BigML	150$ - 300$	16 MB	Batch Tiempo Real	✓
Amazon Machine Learning	0.42$ / hora	100 GB	Batch Tiempo Real	✓
Microsoft Azure Learning	1$ / hora	10 GB	Tiempo Real	✓
Google Cloud Machine Learnin Engine	0.54$ / hora	---	Tiempo Real	✓

Nota: Se presenta un resumen de las ventajas y desventajas que se podrían considerar en cada una de las APIs Web (Corral Plaza, Resinas, & Boubeta-Puig, 2018).

Se evalúa las características de cada una de las APIs y es claramente visible como existen ventajas de una sobre otra, sin embargo todo dependerá de la aplicación que se

vaya a desarrollar y de los recursos económicos disponibles, pues las mejores opciones en este caso son Amazon Machine Learning y BigML que con los mejores recursos a disposición, aunque BigML resulte ser una mejor opción para que la eficiencia de los proyectos sea alta su precio es un factor en contra y es de gran relevancia a considerar, mientras que Amazon resulta una opción de mayor conveniencia pues si ofrece una calidad un poco menor el costo a pagar es considerablemente menor y es por lapsos cortos de tiempo siendo una ventaja ya que en aplicaciones de investigación los tiempos de uso suelen ser limitados y cortos. A todo esto, siempre la decisión se toma en base a las necesidades.

Reconocimiento de imágenes

Existen ya muchos aplicativos para el uso del reconocimiento o procesamiento de imágenes utilizado para medio de investigación, así como en el medio industrial, con una imagen se puede tener varios parámetros, elementos y atributos, en términos sencillos información que necesita ser procesada e identificada para solventar una problemática previamente definida. Las APIs Web ofrecen como servicio el trabajar con el reconocimiento de imágenes utilizando lo que se conoce como IA Inteligencia Artificial y el Deep Learning (Aimacaña & Columba, 2021).

Las imágenes son procesadas mediante los algoritmos previamente indicados sobre Machine Learning principalmente redes neuronales, los cuales están en constante entrenamiento para que tengan la capacidad de realizar una mejor predicción con un porcentaje de acierto cercano al 100%, las imágenes son almacenadas y procesadas principalmente en 2 regiones X y Y, lo cual permite obtener un patrón de parentesco (González García, 2018).

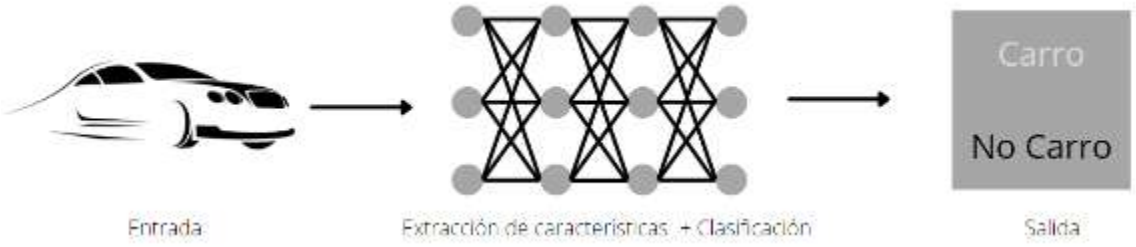
Deep Learning. Es un algoritmo con la capacidad de aprendizaje independiente con un requerimiento de gran cantidad de información, se basa en el uso de redes neuronales para un mejor procesamiento con una mejora del análisis predictivo. Está

estructurado en 3 partes donde primero destaca la entrada del sistema que es la clasificación de datos en un grupo y cuando se detecta diferencias u otros aspectos alternos los aprovecha como diferenciadores para una mejor clasificación, segundo se tiene la capa oculta o de procesamiento y como tercero está la salida que en sí es la respuesta (Casanova, 2021).

El Deep Learning tiene un nivel de complejidad elevado y requiere de numerosos recursos informáticos para su funcionamiento, en la figura 25 podemos entender de forma visible como trabaja este algoritmo.

Figura 25

Estructura Deep Learning



Nota: Figura tomada de (Casanova, 2021).

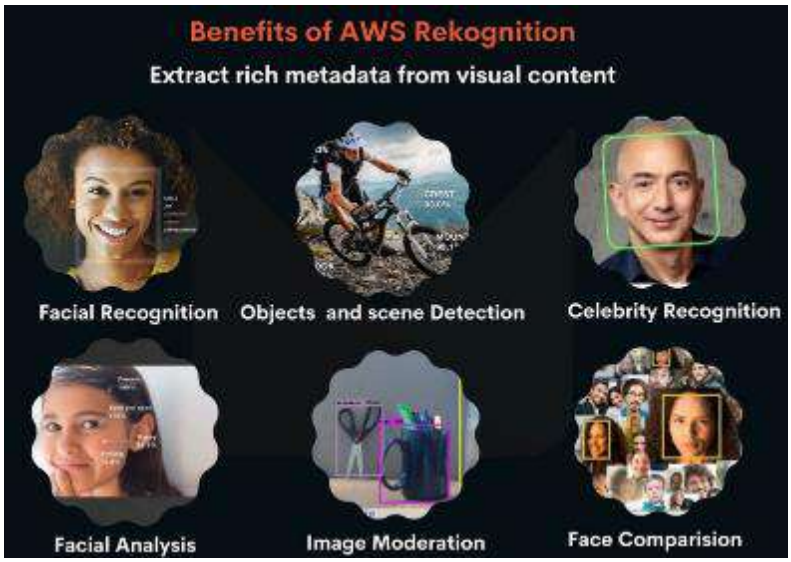
AWS Rekognition. AWS Rekognition es una API Web para el reconocimiento de imágenes que capta objetos, personas, ambientes y demás elementos percibidos, este servicio utiliza el Deep Learning para su entrenamiento y al ser utilizada por gran cantidad de usuarios es una herramienta con una eficiencia muy alta, además no se requiere un conocimiento previo de redes neuronales o Deep Learning pues ofrece un fácil y rápido manejo de la herramienta en el cual nosotros colocamos nuestra entrada que sería una imagen o video y a la salida obtenemos las etiquetas de los elementos encontrados en formato JSON (Diaz, 2020).

AWS Rekognition viene en una versión gratuita y otra de paga lo que da flexibilidad al usuario, pues en caso de no necesitarse por tiempos prolongados y solamente para un

entorno de pruebas la versión gratuita es más que suficiente pues presta un servicio de reconocimiento de 5000 imágenes por mes, también brinda diversas funcionalidad por si se requiere algo específico teniendo así las opciones detección de objetos y escenas, reconocimiento facial, análisis facial, comparación de rostros, detección de imágenes no seguras, reconocimiento de famosos, texto en imágenes y detección de equipo de protección personal (Diaz, 2020).

Figura 26

Opciones de detección de imágenes con AWS Rekognition



Nota: Figura tomada de (AWS, 2020).

Por estas razones AWS Rekognition tiene ventajas sobre otras APIs Web pues la cantidad de aplicaciones que se pueden tener son numerosas hay algunos ejemplos como que cada país tiene sus políticas de privacidad por lo que el moderar el contenido inapropiado se regula en tiempo real utilizado el servicio de Amazon, también se utiliza para la búsqueda de personas desaparecidas, control del equipo de protección para el personal dentro de las empresas y muchas otras aplicaciones (Casanova, 2021).

Servicios Web

Arquitecturas orientadas a servicios (SOA)

SOA representa el diseño de una organización o estructura para un sistema en donde se tiene como idea central la combinación de múltiples servicios los cuales se van a encontrar dispersos en la red, donde cada servicio está bajo un dominio propio con procedimientos, lenguajes y protocolos estandarizados para el cambio de información cumpliendo una función única, algunos de los que se pueden mencionar son los de reutilización de sistemas, los servicios universales, los compuestos, aquellos proporcionados por una organización externa y los nuevos. Todo se coordina por una plataforma para ser ejecutado a través de flujos con el fin de que se utilice lo que se necesita en el instante que se es requerido (Komoda, 2008).

Las arquitecturas orientadas a servicios son aplicadas a problemáticas en donde las decisiones pueden tomarse gracias al uso de las TI o tecnologías de la información las cuales están en interacción directa con los usuarios ya sea a través de registros en base de datos, procesamiento de datos, eliminaciones de datos y otras operaciones que pueden llevarse a cabo desde distintos puntos involucrando a diferentes proveedores de servicios (Laskey & Laskey, 2009).

El uso de SOA viene con notables beneficios a su favor pues es un sistema moldeable en donde los servicios implementados pueden ser removidos fácilmente, la combinación de los servicios a través de plataformas tiene costos relativamente bajos al igual que su mantenimiento y operación, existen estándares de uso para que el desarrollo del sistema. Aspectos negativos al utilizar SOA son los tiempos de respuesta bajos a requerimientos pues al trabajar con distintos proveedores puede existir una sobrecarga de análisis XML y la calidad entra en dependencia de cómo se estructure junto con los servicios (Komoda, 2008).

Microservicios

Esta definición se origina a partir de SOA pues en si es dividir a los sistemas para que los servicios funcionen con mayor eficiencia y de forma fluida, en otras palabras SOA se maneja en base a los proveedores de servicios mientras que los microservicios hacen uso de las APIs Web que son una parte del servicio de un proveedor (Larrucea, Santamaria, Colombo-Palacios, & Ebert, 2018). Los microservicios son aplicaciones que se puede implementar de forma independiente junto con su propia arquitectura, tecnología y plataforma, en consecuencia forma un patrón de aplicaciones compuesto de partes pequeñas que son otras aplicaciones, con el fin de cumplir diversas tareas que al agruparse crean un servicio completo que siempre se representa por medio de interfaces para que el usuario pueda acceder y manipular, por ejemplo es lo que se observa al momento de interactuar con una API Web . Normalmente un microservicio está conformado por 3 capas que son la de interfaz, la capa lógica empresarial y la capa de persistencia de datos (Xiao, Wijegunarathe, & Qiang, 2017).

Se utilizan protocolos de comunicación por Internet y el formato para el manejo de envío y recepción de información se lo aplica en formato JSON, el cual envía los datos en cadenas de texto organizada en parámetros, administrando así la información con carácter exclusivo para otro microservicio no pueda tener acceso a ella sin un permiso previo. Para mejor comprensión de cómo se diferencia al SOA de los microservicios se presenta la tabla 6 con una comparativa de que utiliza SOA y los microservicios para solventar ciertas funciones, es aquí en donde se ve que si utilizamos solo una parte del servicio completo es decir un microservicio los costos serán menores (Xiao, Wijegunarathe, & Qiang, 2017).

Tabla 6

Comparativa de SOA y Microservicios ante distintas funciones

Función	SOA	Microservicio
Punto central de administración. Servicios/APIs, buscador y reconocimiento	Servicio de registro, servicio de repositorio	API de registro empresarial, microservicio de repositorio empresarial
Donde servicios/API son hechos para consumo	Servicio bus empresarial	Capa de gestión API (API proxy)
Endpoint	Proveedor de servicio	Proveedor de microservicio
Otras aplicaciones	Usuario de servicio	Usuario de microservicio
Tiempo de ejecución de Servicio/API para monitoreo y administración	BAM, servicio de vigilancia	Monitoreo empresarial y seguimiento

Nota: La presente tabla muestra cómo se diferencia los componentes que utiliza un SOA de un microservicio (Xiao, Wijegunarathe, & Qiang, 2017).

Se observa como un microservicio puede suplir una función igual que un servicio, pero en cambio el nivel de complejidad en el uso es menor, solo se dispone de lo necesario sin desperdiciar recursos y por ende el costo será menor.

Formato JSON. Los microservicios utilizan el formato JSON JavaScript Object Notation que son un representación de los datos en una sintaxis de cadenas de texto que se conforman de un parámetro o variable y de su valor correspondiente, son utilizados para transmisión de información por la facilidad y rapidez, además de que al ser

estandarizado puede ser utilizado por toda plataforma pues solo se extrae la información que se necesite, cabe decir que se habla solo de un formato de envío de información mas no de un método.

API REST (Representational State Transfer)

Se expone como un conjunto de operaciones que están en la capacidad de llamar a través de una ruta base a un conjunto de recursos, estas operaciones son llamadas por un cliente HTTP o código en JavaScript que se ejecuta en un navegador web. La ruta base tiene como funcionalidad el aislar a la API REST de otras, entonces el cliente utiliza esta ruta para acceder a los recursos de la API REST aquí las rutas deben estar estructuradas de forma jerárquica con el fin de organizar los recursos a disposición (Surwase, 2016).

Según (Li & Wu, 2011) REST posee propiedades de usabilidad, simplicidad, escalabilidad y extensibilidad para reducir latencias en las comunicaciones con la red y para una API REST se establece a una agrupación de interfaces que se ponen en marcha por recursos.

Las operaciones tiene nombre y un método HTTP por ejemplo GET para consulta y lectura de recursos, POST para creación de recursos, DELETE que elimina recursos, PUT para editar recursos y PATCH para editar partes concretas de un recurso, siendo estos nombre particulares para su distinción, además de las operaciones también hay parámetros para pasar argumentos a la operación teniendo así al parámetro de ruta que identifica un recurso en particular, el parámetro de consulta que permite obtener un valor clave de una cadena de resultados final y el parámetro de encabezado que sirve de identificadores para un cliente HTTP (Surwase, 2016).

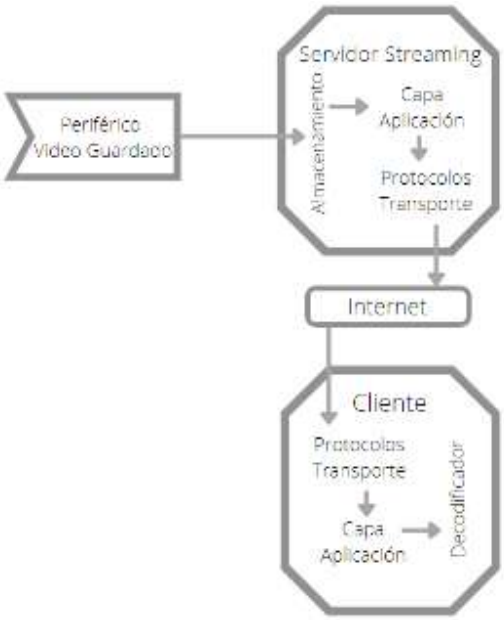
Servicios Streaming

Estos servicios indican la idea de una transmisión de información de forma continua lo que origina que se pueda ofrecer transmitir video en tiempo real usando una

arquitectura robusta gracias a las nuevas tecnologías y mejora de las redes. En la figura 27 se aprecia cámo se constituye la arquitectura de un video streaming, indicando el proceso de comprensión de los datos para que luego se almacenen estando a espera de un requerimiento de cliente en la capa de aplicación para que la información sea enviada ajustando sus parámetros como calidad y tiempo de respuesta en dependencia de la red utilizada, como consecuencia si la velocidad de la red no es adecuada se puede tener pérdida de información motivo por el cual se crean servicios para el control de embotellamiento y errores (Rebolledo & Guerra, 2015).

Figura 27

Arquitectura video streaming



Nota: Figura tomada de (Rebolledo & Guerra, 2015).

Los protocolos se aplican a la capa de red, transporte y sesión para un respectivo control, aquí podemos tener dos protocolos para el streaming de medios el primero es push que sirve para que el servidor este enviando constantemente información al cliente y éste sea quien decida seguir o dejar de recibirla cuando desee, se tiene en cuenta que el streaming se emite con un protocolo en tiempo real RTSP aplicado en la capa de

aplicación y el protocolo de transporte en tiempo real RTP aplicado en la capa de transporte; el segundo protocolo es pull a diferencia del anterior aquí para que se inicie la transmisión dependerá solamente del cliente pues si éste tiene una petición entonces en ese momento el servidor comienza a emitir la información (Rebolledo & Guerra, 2015).

Los tipos de servicios que se tiene un streaming son las de video en directo orientado a multidifusión, con protocolo push sin que haya interactividad cliente-servidor, presentándose dos tipos de transmisión de streaming uno es el unicast en el cual se transmite un flujo de datos para cada usuario y el multicast transmite un único flujo de datos utilizado para varios usuarios, y el otro servicio es de baja demanda que funciona con protocolo pull, el flujo de datos es para cada usuario y existe interacción cliente-servidor (Suárez, 2011).

Orquestación de Servicios Web

Es un proceso para integrar y trabajar con diversos microservicios web, con el fin de que se puedan ejecutar tareas basadas en diferentes aplicativos, en sí controlar el uso de los servicios siguiendo un orden determinado pues se busca un entorno colaborativo, en la figura 28 se observa una estructura de orquestación de forma general.

La orquestación debe ser dinámica para un entendimiento tanto adecuado como sencillo, adaptable y flexible ya que está susceptible a cambios que se pueden ir generando dependiendo del aplicativo y necesidades, así una organización está en la capacidad de aumentar el flujo de información y ofertar un mejor servicio (Peltz, 2003).

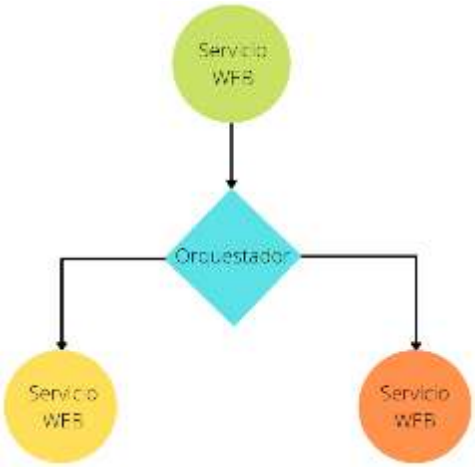
Ahora para que una orquestación sea considerada aceptable y estable debe tener en cuenta varios requisitos según (Mazzara & Govoni, 2008) como:

- Ser secuencial.
- Ser paralelo.
- Condicional.

- Usar un formato específico como JSON para el envío y recepción de datos entre servicios.
- Manejar errores.

Figura 28

Orquestación de Servicios Web



Nota: Figura tomada de (Peltz, 2003).

Existen varias herramientas como orquestadores disponibles en el mercado por varios proveedores con diferentes recursos y disponibilidades de desarrollo, para este caso se escogieron dos de las más conocidas que son Node-RED y Node.js

Node-RED. Esta herramienta ofrece un entorno gráfico para la interconexión de flujos en donde se puede programar la conectividad entre diferentes API y servicio, así como con hardware, ofrece también edición de funciones JavaScrpt y almacena los flujos en formato JSON.

Node.js. Está orientado a evento asíncronos, su ejecución se la realiza en JavaScript permitiendo incorporar diversos servicios para aplicaciones que sean escalables.

Tecnologías y Protocolos de comunicación

Servicios de mensajería ligera (MQTT)

El protocolo MQTT es utilizado para comunicación de hardware en el IoT Internet de las Cosas, este protocolo es de suscripción y publicación, para su funcionamiento y dispositivo puede publicar un mensaje bajo un nombre específico de tal forma que los dispositivos alternos que se encuentren suscritos al dicho nombre receptarán el mensaje publicado por el primer dispositivo. MQTT dispone de una cabecera que puede contener un nombre y una contraseña como autentificación un déficit es que estos datos no se encuentran cifrados por lo cual su seguridad no se podría decir que es de lo mejor (Singh, Rajan, Shivraj, & Balamuralidhar, 2015).

MQTT es un protocolo abierto dispuesto a aplicaciones de telemetría, donde se asume que la red subyacente ya otorga un servicio para el transporte de información punto a punto como TCP/IP, los datos se transportan en cadenas de caracteres, algo importante a destacar en este protocolo es la existencia de un intervalo de tiempo entre envío y recepción de datos, por lo cual si este tiempo se ve superado entonces automáticamente se envía un mensaje con el ping para así evitar que exista una desconexión (Hunkeler, Truong, & Stanford-Clark, 2008).

El protocolo admite una calidad de servicio básico que lo difiere en 3 niveles, el primer nivel es QoS 0 es el más sencillo donde los mensajes se envían una vez y no existe una confirmación de recepción, el segundo nivel QoS 1 es más fiable pues existe una confirmación de que si se recibió el mensaje, pero existe un déficit de que se pueden enviar más de una vez cada mensaje, por último, está el tercer nivel QoS 2 que garantiza una confirmación y que el mensaje se haya enviado una sola vez (Hunkeler, Truong, & Stanford-Clark, 2008).

Protocolos y tecnologías en el front-end

El front-end es la parte donde va a tener contacto el usuario para realizar peticiones e interactuar con el sistema desarrollado, están representadas por interfaces gráficas o HMI que contienen indicadores, gráficas, botones con una distinguida distribución de colores y posición de elementos para que sea adecuado y cómodo de utilizar, y la forma de programarlo es enlazándose directamente al programa principal mediante comandos de interacción lo cual es de gran conveniencia pues al momento de ofertar un producto o servicio se debe comunicar con un lenguaje entendible para todo público, buscando ser algo innovador, atractivo y amigable, pues esto es lo que produce el primer impacto ya sea positivo o negativo sobre lo desarrollado (Markham, 2013).

El protocolo HTTP es el utilizado para la capa de aplicación basado en el envío de información a través de la capa de transporte pues el cliente es quien realiza alguna solicitud o requerimiento y el servidor da una respuesta a dicha solicitud. Los recursos identificados en HTTP son las URI Uniform Resource Identifier que son secuencia de caracteres para identificar recursos en base a protocolos y a su vez estos dan la información de localización y acceso a los recursos URL Uniform Resource Locator (Arias, 2020).

También se definen los métodos de mayor uso que son GET para obtención de un recurso y POST para modificar un recurso, explicado esto entonces se entiende que la estructura de una petición HTTP está compuesta de la URL, el método, una cabecera como complementos y el cuerpo que es la información a procesarse (Arias, 2020).

JavaScript. Es el lenguaje empleado para diseño de páginas web dinámicas que están conformadas de animaciones, texto, botones y demás elementos para que se logre visualizar una interfaz amigable, clara y simple con la cual interactuar. Varias plataformas para la creación de páginas web y las HMI ofrecen facilidades en cuanto a programación gráfica o por bloques para evitar usar el código de programación escrito de JavaScript, lo

que estas plataformas realizan es tener las funciones del código de JavaScript dentro de bloques en donde el usuario modifique a su conveniencia (Eguíluz, 2009).

Los programas creados por JavaScript son conocidos como scripts y se ejecutan en el navegador web, este lenguaje se originó tomando base ciertos criterios de Java, pero es un lenguaje diferente con otra finalidad, aunque de ser necesario un código de JavaScrpt si podría ser ejecutado en Java (Navarrete, 2008).

Tecnologías de acceso (WI-FI, LoRaWAN)

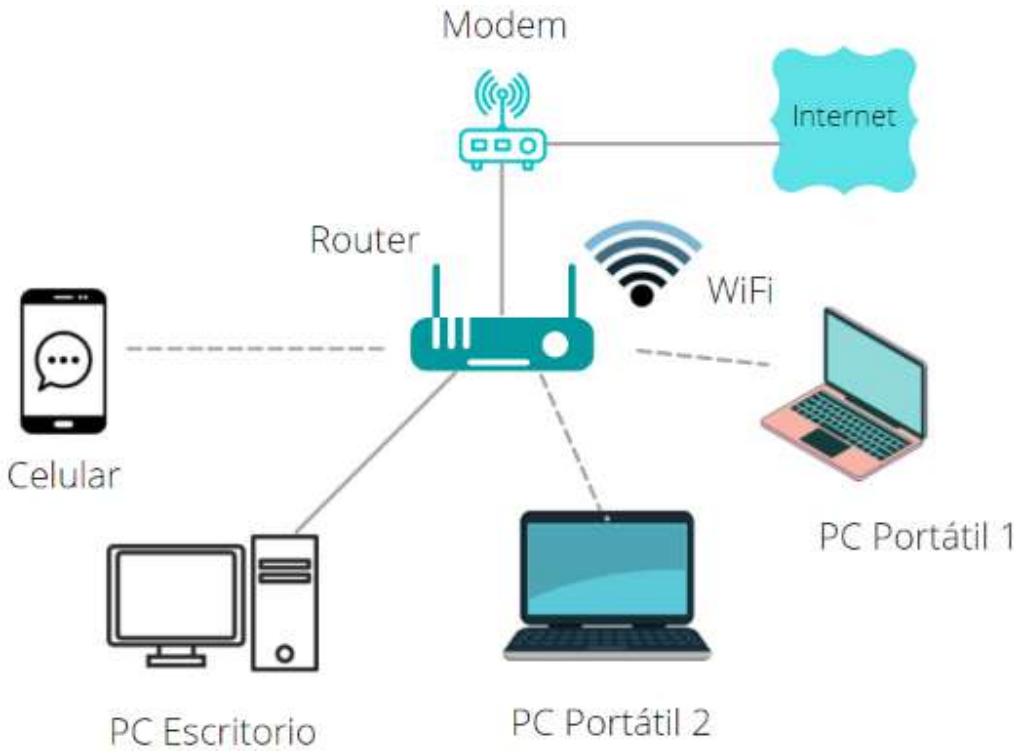
Estas tecnologías son aquellas que permiten establecer una conexión entre el cliente y el maestro para que la transmisión de datos a través de distintos anchos de banda con la capacidad de ofrecer una velocidad adecuada a ciertas distancias, en los inicios esta tecnología solamente era alámbrica encontrando aquí por ejemplo el cable coaxial que ya se ha ido reemplazando por la fibra óptica, se ha ido evolucionando mejorando así el servicio de transmisión pues cada vez la cantidad de datos aumenta y la calidad exigida por los usuarios es mayor. Otro punto a tener en cuenta es que las tecnologías de acceso en la actualidad se prioriza el trabajar de forma inalámbrica en el punto donde se conectan distintos periféricos como en hogares, oficinas y demás, por ejemplo, las utilizadas principalmente para el desarrollo del proyecto expuesto, están las tecnologías de WiFi y LoRaWAN (Restrepo Jaramillo & Restrepo García, 2014).

WiFi. Es una tecnología regida por el estándar IEEE 802.11 a la cual los terminales se conectan de forma inalámbrica formando parte de las WLAN que son las tecnología inalámbricas de área local, este estándar ha ido cambiado y mejorando la tecnología como es el estándar IEEE 802.11g se trabaja con tasas de transferencia de 54 Mbit/s, luego con la 802.11n la tasa era de 540 Mbps, con el estándar IEEE 802.11 AC la banda de operación es en 5GHz y por último el estándar 802.11 ac que permite una de 1.3 Gbps a través de 3 canales de 433 Mbps trabajando en la banda de 5GHz (Rodriguez, 2020). La disposición de esta tecnología es a través de un router,

adaptadores o puntos de acceso visualizando en la figura 29 de mejor manera como es la conectividad.

Figura 29

Conectividad red WiFi

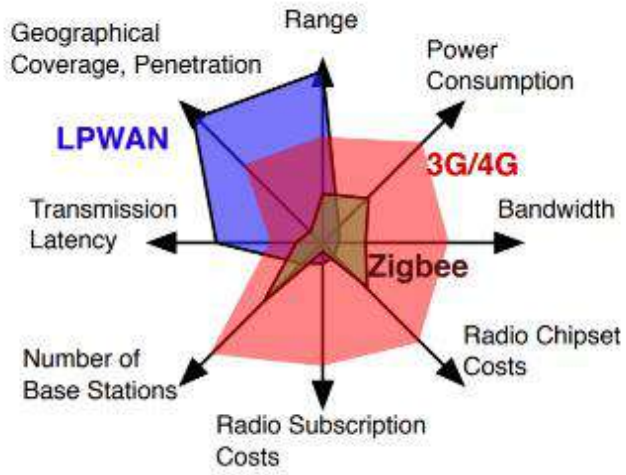


Nota: Figura tomada de (Rodriguez, 2020).

LoRaWAN. Esta tecnología pertenece a LPWAN que son las redes inalámbricas de bajo consumo y largo alcance, es aplicada en circunstancias donde se necesita realizar un rastreo, transmisión y recepción de datos en áreas donde no se disponga de otra conectividad como la móvil y no se tiene un costo adicional por el uso de un servicio de comunicación pues aún no existe un propietario o monopolio para las redes LoRa actualmente, en la figura 30 podemos observar la comparativa del LPWAN de LoRa con respecto a otras tecnologías (Adelantado, et al., 2017). El protocolo para LoRaWAN utilizada para aplicaciones IoT es la IEEE 802.11ah.

Figura 30

Comparación de LoRa con otras tecnologías de largo alcance



Nota: Figura tomada de (Hernandez, et al., 2017).

LoRa puede trabajar banda ancha estrecha por lo que la cantidad de información a tratarse no puede ser extensa lo cual es un limitante en los aplicativos que podría tener, el protocolo LoRaWAN es abierto siendo fácil de actualizar mientras está en funcionamiento algo a destacar es la bidireccionalidad pues puede emitir y receptar información volviéndolo apto para control y monitorización (Adelantado, et al., 2017).

Herramientas de desarrollo de pruebas

En el desarrollo de un proyecto de investigación, es inevitable el desarrollo de pruebas, ya que de esta manera se puede identificar errores tanto de diseño o funcionalidad como de requerimiento, para posteriormente poder corregirlos durante la marcha. De esta manera evitamos que el sistema genere errores en un entorno crítico, lo que traería graves consecuencias. Para este fin podemos realizar dos tipos de pruebas, las manuales que son las que realiza el desarrollador paso a paso y las automáticas que son realizadas con la ayuda de algún software diseñado para la finalidad de la prueba requerida (Gonzales , 2020).

Sistema de Escalas de Usabilidad (SUS)

Las siglas SUS qué hacen referencia a System Usability Scale o Sistema de Escala de Usabilidad, es una encuesta confiable, de distribución gratuita y de uso común que consta de 10 ítems. La puntuación de la encuesta produce una puntuación de usabilidad en el rango de 0 a 100. Su facilidad de administración y puntuación lo convierte en una opción popular entre los profesionales que desean saber la usabilidad de sus desarrollos. Investigaciones recientes han demostrado que el SUS proporciona evaluaciones superiores de la usabilidad de cualquier proyecto, en comparación con otros cuestionarios (por ejemplo, QUIS, CSUQ). Además, se ha encontrado que tiene una correlación de 0.8588 con el cuestionario SUMI de 50 ítems, que es uno de los más sofisticados en este ámbito.

El SUS fue desarrollado inicialmente por John Brooke mientras estaba en Digital Equipment Corporation en 1996. Además de ser una opción popular para las encuestas de usabilidad en línea, el SUS puede usarse como una medida subjetiva de seguimiento después de probar la usabilidad de los sistemas funcionales como un componente de las pruebas previas y posteriores.

Las preguntas que contiene esta encuesta son:

1. Creo que usaría este […].
2. Encuentro este […] innecesariamente complejo.
3. Creo que el […] fue fácil de usar.
4. Creo que necesitaría de una persona con conocimientos técnicos para poder usar este […].
5. Las funciones de este […] están bien integradas.
6. Creo que el […] es muy inconsistente.
7. Imagino que la mayoría de la gente aprendería a usar este […] en forma muy rápida.

8. Encuentro que el […] es muy difícil de usar.
9. Me siento confiado al usar este […].
10. Necesité aprender muchas cosas antes de ser capaz de usar este […].

Donde el espacio representado por […] puede atribuirse el término objeto, aplicación, sistema, dispositivo, etc, según sea el caso del proyecto que desea evaluarse.

Las diez preguntas pueden ser puntuadas de 1 a 5, donde 1 se interpreta como total desacuerdo, mientras que 5 significa total acuerdo. Para obtener los resultados bajo este método, se debe considerar que las preguntas impares tendrán el valor asignado por el usuario menos 1, mientras que para las preguntas pares será de 5 menos el valor asignado por el usuario. Por último, se realiza la sumatoria de estos valores y al resultado final, se lo multiplica por 2.5, por lo que el test completo tendrá una evaluación sobre 100. El valor final, será el promedio de la evaluación de todos los usuarios (Finstad , 2006).

Honeycomb

Honeycomb es un software muy versátil para la realización de pruebas de integración, ya que cuenta con la capacidad de hacer un seguimiento en tiempo real de los acontecimientos del sistema por largos periodos de tiempo, que gracias a su interfaz anclada en la red se puede realizar un monitoreo constante.

Por otra parte, este software ayuda a los desarrolladores a comprender de mejor manera los sistemas de software que en algunos casos son muy complejos. De esta manera se puede realizar una depuración rápida y mejorar el rendimiento de la aplicación. Cuenta con la capacidad de graficar todos los acontecimientos, lo que reduce el tiempo para encontrar problemas o casos atípicos en el sistema. Una característica esencial es que tiene una opción de supervisión que permite detectar cuellos de botella y errores de latencia que no se han tomado en cuenta durante el desarrollo.

Este software cuenta con una interfaz gratuita para pruebas de integración sencillas, ya que, de requerir un mayor análisis, cuenta con versiones pagadas que están

enfocadas a áreas de programación específicas. Su base de datos está dotada con scripts que pueden ser acoplados fácilmente a cualquier tipo de programa y lenguaje de programación que se conoce hoy en día (Honeycomb.io, 2016).

Apache JMeter

Es una herramienta en un software Java de código abierto cuya funcionalidad está destinada a conocer el rendimiento que tiene una aplicación web sometida a diferentes niveles de estrés, como lo es tiempos de respuestas, porcentajes de éxito y falla, donde simula un entorno con diferentes cargas sobre un servidor.

Es utilizado para diferentes protocolos como HTTP utilizan software como Java o NodeJS, también para probar servidores REST, bases de datos, correos, MQTT entre otros. Los datos los maneja en diferentes formatos como lo es HTML, JSON o XML para poder invadir a las aplicaciones sometidas a pruebas, Apache JMeter finalmente genera un reporte donde coloca parámetros como tiempos, porcentaje de fallos, cantidad de mensajes enviados y recibidos que permitan al evaluador identificar rápidamente en donde existen errores y luego tomar decisiones que puedan solucionarlo (JMeter, 2021).

Capítulo 3: Diseño

En este capítulo se detalla el proceso de diseño de los elementos necesarios tanto de hardware como de software que conformarán el sistema que cumpla con la finalidad de obtener un robot móvil tele-operado con capacidades de reconocimiento de imágenes a través de Cloud Computing para la exploración de zonas distantes. El diseño es la parte esencial de un proyecto, por lo que debe estar sustentado en una metodología que proporcione las garantías necesarias para lograr los objetivos, en consecuencia, se usó la norma de diseño VDI 2221.

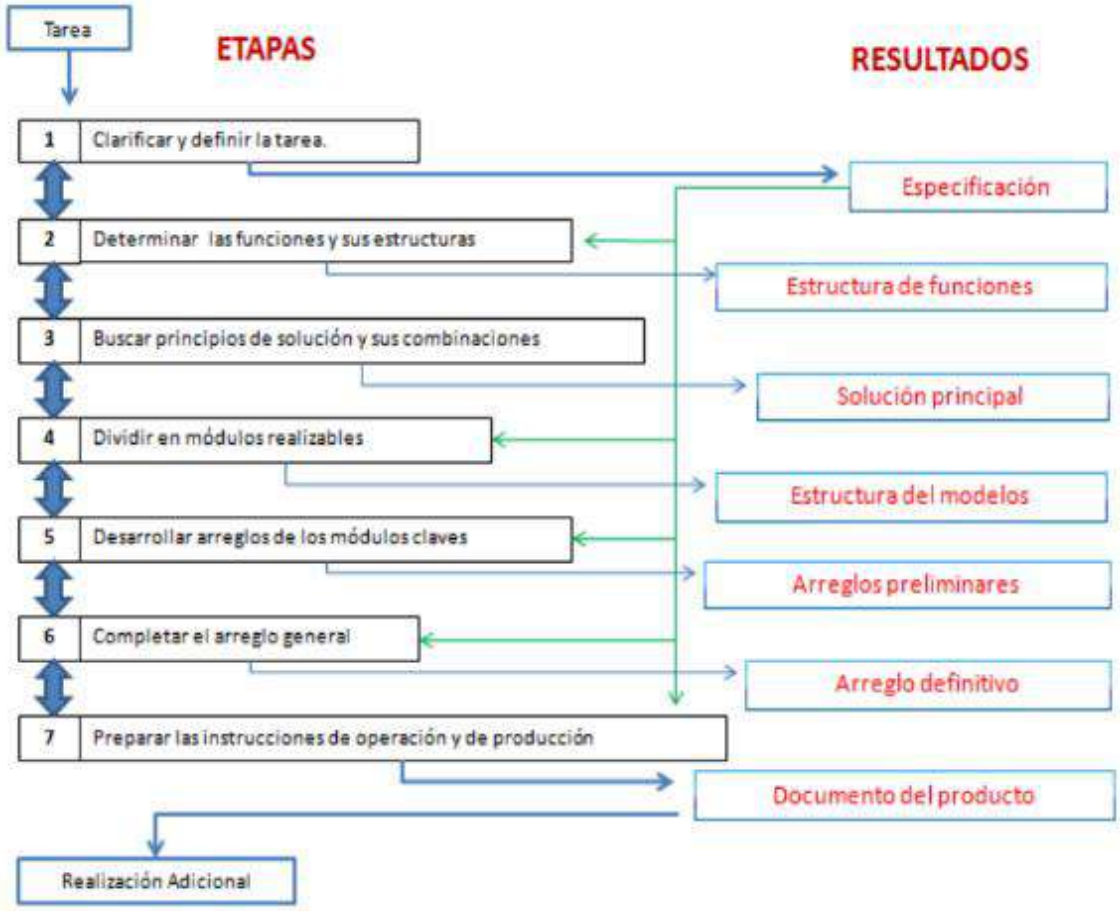
Metodología de diseño

La norma de diseño VDI 2221 es una metodología de diseño general, creada en Alemania por “La sociedad de ingenieros profesionales” y orientada a sistemas técnicos para apoyar un diseño metódico y sistemático, con el fin de lograr un trabajo más eficiente. La norma es aplicable independientemente de la rama que aborda el contenido y los aspectos organizativos del diseño. Se destaca la amplia aplicación de la norma dentro de la ingeniería mecánica, electrónica, desarrollo de software y la planificación de la ingeniería de procesos. La norma está basada en la ingeniería de sistemas y la resolución de problemas (Jansch & Birkhofer, 2006).

El proceso de diseño que establece la norma consta de siete etapas, las mismas que han sido modificadas a lo largo del tiempo adaptándolas al diseño de diversos elementos y proyectos, naciendo otras normas VDI de diferente numeración derivada de esta que sirven para la gestión de diseño y fabricación de elementos que generalmente se usan en la industria. En este caso se usará la norma VDI 2221, que es la base del diseño de la ingeniería de proceso y está distribuida como se muestra a detalle en la figura 31

Figura 31

Metodología de diseño según la norma VDI 2221



Nota: Figura tomada de (Sanchez Samaniego, 2014)

A continuación, se da un breve resumen de las etapas que componen la metodología para su posterior aplicación en el proyecto.

- Etapa 1. Requerimientos: Se establecen las especificaciones de los requerimientos de diseño, acordes al producto a ser desarrollado para de esta manera definir las especificaciones que pueden sufrir modificaciones a lo largo del proceso de diseño.
- Etapa 2. Funciones y estructuras: En esta parte se realizan diagramas de funcionalidad del sistema, así como de la estructura.

- Etapa 3. Principios de solución y variables: Se realiza la ingeniería de detalle de todas las soluciones posibles analizadas en las etapas anteriores.
- Etapa 4. Módulos: Se divide en módulos, que harán ver el sistema con menor complejidad y poder centrarse aún más en los detalles.
- Etapa 5. Esquemas de los módulos: Se analizan posibles inconvenientes o consideraciones que no han sido tomadas en cuenta de los diseños realizados.
- Etapa 6. Requerimientos de los módulos: Se establece el diseño final con un análisis de modelado y la integración del sistema.
- Etapa 7. Producto: Se integran los módulos y de esta manera se detallan los aspectos de funcionamiento que deben ser considerados en la fase de implementación.

Diseño del robot móvil

Etapa 1. Requerimientos

Tabla 7

Requerimientos del Usuario

Nº	Requerimiento
1	Servicio de Streaming desde la cámara del robot
2	Movilidad y exploración en cualquier tipo de superficie
3	Brazo robótico para manipulación de objetos
4	Buena autonomía
5	Dimensiones acordes a un prototipo
6	Control a largas distancias
7	Bajo Costo

Nota: En esta tabla se detallan los requerimientos del Usuario para referenciar el diseño

Tabla 8

Características Técnicas

Nº	Características técnicas
1	Control de cámara
2	Control de dirección y movilidad del robot
3	Control del brazo robótico
4	Sistema de alimentación de energía
5	Dimensiones del robot
6	Comunicación inalámbrica a larga distancia
7	Costo de elementos

Nota: En esta tabla se detallan las características técnicas del robot en base a los requerimientos del usuario.

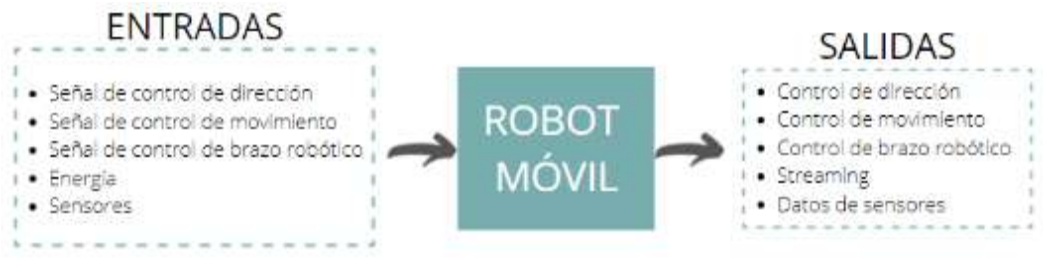
La función principal de robot móvil será de ejecutar las acciones que se le enviaran desde la HMI, así como también de enviar las señales de los diferentes sensores que contendrá para su posterior exposición en la HMI. Desde aquello se consideran dos aspectos fundamentales como son los requerimientos del Usuario y las características técnicas que se muestran en las tablas 7 y 8 respectivamente.

Etapa 2. Funciones y estructuras

Para establecer las funciones del robot se tomará en cuenta los requerimientos del usuario, así como también las especificaciones de entrada y de salida para de esta manera obtener los procesos técnicos que deberá proporcionar el robot, así como las características, subfunciones y la división en módulos para tener un diseño con mayores detalles.

Figura 32

Entradas – Salidas, que debe contemplar el robot



Nota: Detalle de entradas y salidas del robot con respecto a los requerimientos de diseño

Entradas.

- Señal de control de dirección: Señal de corriente continua de 5V.
- Señal de control de movimiento: Señal de corriente continua de 5V.
- Señal de control de movimiento: Señal de corriente continua de 5V.
- Energía: Fuente de voltaje DC de 5V

Salidas.

- Control de dirección: Activación de servomotor de corriente continua de 5V.
- Control de movimiento: Activación de motores de corriente continua de 5V.
- Control de brazo robótico: Servomotores de corriente continua de 5V.
- Streaming: Cámara con capacidad de transmisión de video continuo.
- Información de sensores: Sensor ultrasónico para determinar la distancia de obstáculos y sensor GPS para determinar la ubicación del robot móvil.

Procesos técnicos. Los procesos se determinan de acuerdo a la funcionalidad que se desea en las salidas dependiendo de las señales de la entrada, por lo que en base a esa referencia se ha logrado determinar los siguientes procesos que debe cumplir el robot para cumplir con los requerimientos del usuario.

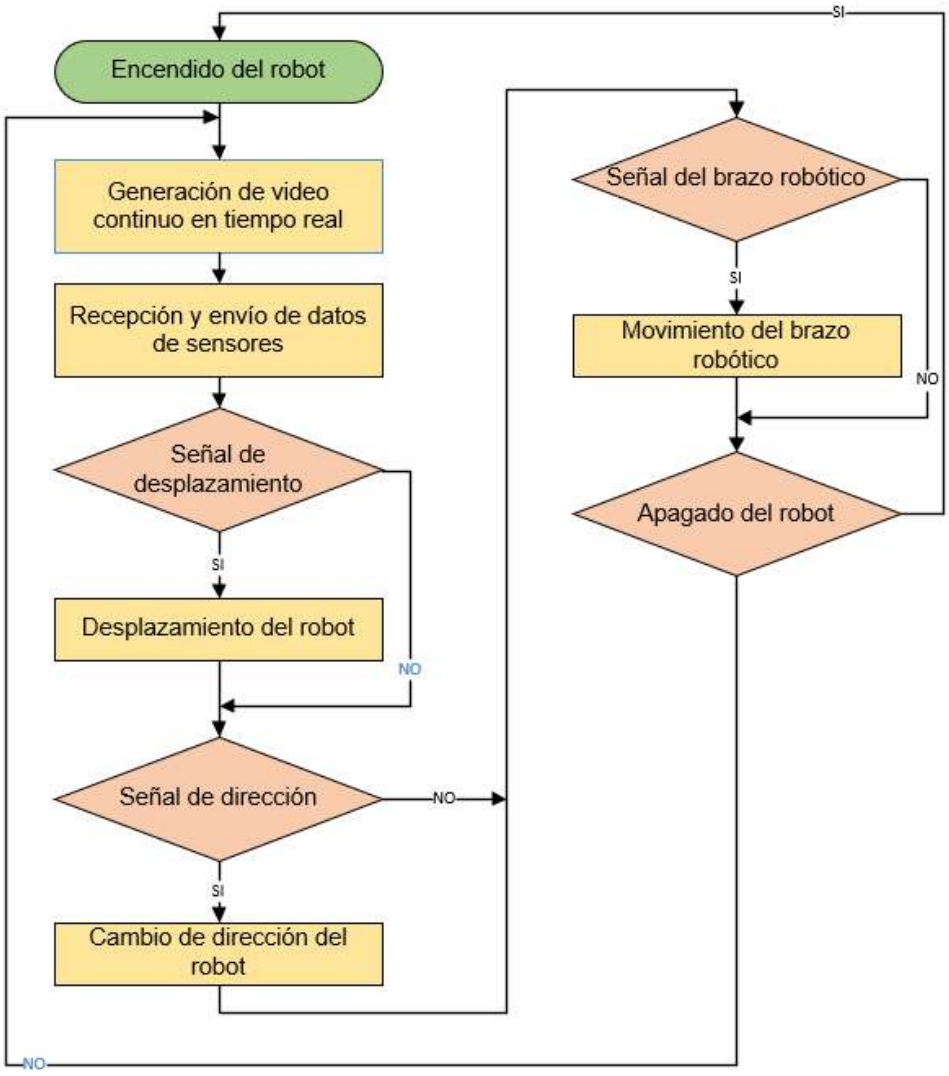
- Desplazamiento y orientación del robot móvil dependiendo de la señal de control para realizar la función de exploración de ambientes.

- Movilidad del brazo robótico dependiendo de la señal de control para la manipulación de objetos.
- Transmisión continua de video en tiempo real para monitorear el entorno.
- Recepción y transmisión de datos de los sensores

Estructura general de funciones.

Figura 33

Diagrama de flujo de la estructura general de funciones



Nota: En este gráfico se muestra el proceso de la estructura general de funciones que realizará el robot.

Etapa 3. Principios de solución y variables

Una vez identificados los requerimientos técnicos y los procesos que el robot ejecutará, se consideran todas las funciones que el robot debe ejecutar con lo que posteriormente se establecen las diferentes alternativas de solución, que adicionalmente brinden todas las garantías de funcionamiento en la aplicación para la cual está destinado.

Tabla 9

Matriz de alternativas de solución morfológica

FUNCIONES	Alternativa 1	Alternativa 2	Alternativa 3
Desplazamiento en cualquier superficie	Robot móvil con ruedas		
Cambio de dirección	Robot tipo Ackermann		
Manipulación de objetos	Brazo robótico Eléctrico	Brazo robótico neumático	Pinzas de sujeción
Transmisión de video	Modulo cámara	Cámara independiente	Cámara de celular
Exploración de zonas distantes	Radiofrecuencia	Wifi	LTE
Autonomía	Batería	Pilas	Panel solar
Ergonomía	Plástico	Aluminio	Fibra de carbono

Nota: En esta tabla se detallan las características técnicas del robot en base a los requerimientos del usuario.

En referencia a la tabla 9 se realiza un análisis de las alternativas relacionadas con su respectiva función para determinar la más adecuada y posterior elección

- Para el desplazamiento en cualquier superficie se consideró únicamente un robot móvil con ruedas, ya que son los más aptos para este tipo de aplicaciones considerando sus características físicas.
- El cambio de dirección se relaciona directamente por el tipo de robot que se usará, para el caso de un robot con ruedas lo más óptimo es considerar un robot tipo Ackerman, además de brindar beneficios en aplicaciones de exploración por la versatilidad en cualquier superficie.
- Para la manipulación de objetos es necesario considerar un brazo robótico que, a diferencia de unas pinzas, brindan mayores beneficios, por otra parte, se debe analizar la fuente que le suministrará la energía para la movilidad. Al ser un sistema que funcionara con la misma fuente de energía que es una fuente de corriente continua, el brazo elegido será un brazo robótico eléctrico.
- Entre las alternativas para la transmisión de video, vamos a considerar un módulo cámara, por su facilidad de uso, ya que viene adaptado con los elementos necesarios para únicamente ser conectado y configurado.
- Para la exploración de zonas distantes se considerará la radiofrecuencia por la cantidad de módulos existentes para tal fin, sus prestaciones gracias a los últimos avances y su bajo costo en el mercado.
- La autonomía del robot será dada por una batería, ya que entre las opciones es la que tiene una mayor durabilidad y puede ser recargada cuando sea necesario sin depender de otros factores.
- Los materiales con los que se pretenden establecer los mecanismos del robot serán plástico y aluminio, por su fácil manipulación y robustez, la fibra de carbono se descarta por su elevado costo y material extra para su manipulación.

Conceptos de solución.

Tabla 10

Conceptos de solución

Nº	Solución
1	Construcción a detalle de elementos mecánicos y electrónicos según las soluciones morfológicas seleccionadas anteriormente, que brinden las prestaciones necesarias para satisfacer la aplicación de destino.
2	Adquirir los elementos prefabricados adaptados a las soluciones morfológicas seleccionadas, para su posterior ensamblaje y adaptación a la aplicación que se requiera.

Nota: Se analizan dos conceptos de solución en base a las alternativas morfológicas.

Tabla 11

Conceptos de solución

INDICADOR	Solución 1	Solución 2
Menor costo		X
Reducción de tiempo de fabricación		X
Reducción de recursos de fabricación		X
Implementación corta	X	X
Mantenimiento	X	X
Personalización	X	
Mejoras periódicas	X	
Asistencia técnica	X	X

Nota: Se analizan dos conceptos de solución en base a alternativas morfológicas seleccionadas.

Análisis de solución. Teniendo los dos conceptos de solución establecidos se realiza una evaluación en la tabla 11, donde se analizan varios indicadores que generalmente las empresas utilizan para analizar la realización o compra de un producto, que posteriormente ayudarán a tomar una decisión de entre las dos alternativas.

Como se puede notar, la solución dos es la que presenta mayores ventajas con referencia a la solución uno, en base a los indicadores presentados, por lo que se decide la búsqueda en el mercado de un robot móvil que brinde las facilidades de adaptación a todos los requerimientos, funcionalidades y características antes mencionadas.

Solución. Se realiza la búsqueda en el mercado del robot móvil prefabricado, que cumpla todas las características tratadas a detalle anteriormente, encontrando el robot "PiCar Pro Smart Robot Car Kit 2-in-1" que se muestra en la figura 34 fabricado por la empresa Adeept, siendo el más óptimo para la aplicación.

Figura 34

PiCar Pro Smart Robot Car Kit 2-in-1

Nota: Figura tomada de (Adeept, 2021)

El robot Picar Pro es un kit didáctico que está basado en la Raspberry Pi, también es una ayuda en aplicaciones educativas orientadas a la electrónica por su fácil adaptación a las necesidades de aprendizaje. La estructura mecánica es simple de armar y de fácil adaptabilidad a variadas aplicaciones en comparación a robots similares. Algunas de sus características son:

- Cuenta con un brazo robótico para variadas aplicaciones
- Está fabricado en aleación de aluminio su carrocería y lámina de acrílico las partes restantes.
- Genera una fuerte tracción en sus cuatro ruedas con lo que puede ser usado con fines de exploración en cualquier tipo de superficie.
- Puede adaptarse a múltiples funciones como: transmisión de video, evasión de obstáculos, seguimiento de objetos entre otros que puedan ser requeridos por el Usuario.
- Entre otras. (Adeept, 2021)

Etapa 4. Módulos

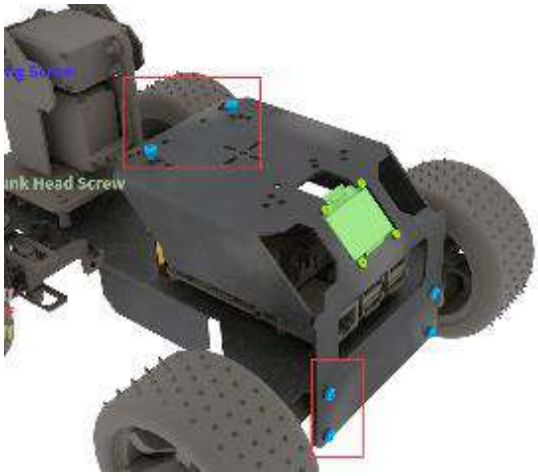
Sistema mecánico. El sistema mecánico del robot Picar Pro consta principalmente de dos partes, el chasis que contiene los sistemas de locomoción, y el brazo robótico para la manipulación de objetos, a continuación, se detallan las características de cada uno:

Chasis.

- Acoplados todos los elementos de locomoción tiene unas dimensiones de 27.18x18.8x10cm.
- Está fabricado con aleación de aluminio.
- Tiene un peso de 1.68 Kg.

Figura 35

Chasis y Sistema de locomoción del robot PiCar Pro



Nota: Figura tomada de (Adeept, 2021)

Brazo robótico.

Figura 36

Brazo robótico del robot PiCar Pro

Nota: Figura tomada de (Adeept, 2021)

- Es un brazo robótico de tres grados de libertad
- Tiene unas dimensiones de 12 cm el primer eslabón, 2.5 cm el segundo eslabón y 7 cm el eslabón vertical.
- Está construido en aleación de aluminio

- Su peso es de 0.37 Kg
- Su efector final es un Gripper construido de láminas de acrílico con unas dimensiones de 9 cm.

Sistema electrónico. El sistema electrónico es la parte fundamental del kit PiCar Pro, ya que de este depende la movilidad de cada una de las partes mecánicas detalladas anteriormente. El Kit PiCar Pro contiene sensores y actuadores mostrados a detalle en la tabla 13, que pueden ser adaptados a cualquier aplicación que necesite el usuario. Su capacidad es expandible ya que de ser necesario se pueden aumentar sensores y actuadores que no están dentro del Kit gracias a los pines de entradas y salidas extras que contiene el Driver de control.

Controlador. El Kit PiCar Pro está desarrollado para controlarse por una tarjeta Raspberry Pi, por lo que se usará la tarjeta Raspberry Pi 4 modelo B 2019, Quad Core de 64 bits con Wifi y Bluetooth.

Figura 37

Raspberry Pi 4 modelo B 2019

Nota: Figura tomada de (Raspberry, 2019)

Fuente de energía. Los elementos descritos en el punto anterior funcionan con un voltaje de 5-12 VDC, por lo que más adelante se realiza un dimensionamiento del consumo de todos elementos que emplea el robot para la aplicación en desarrollo y de

esta manera determinar las características necesarias de la batería que se usará, para su posterior búsqueda en el mercado.

Etapa 5. Esquemas de los módulos

Tabla 12

Diagramas en 3D de los elementos del Chasis del Kit PiCar Pro

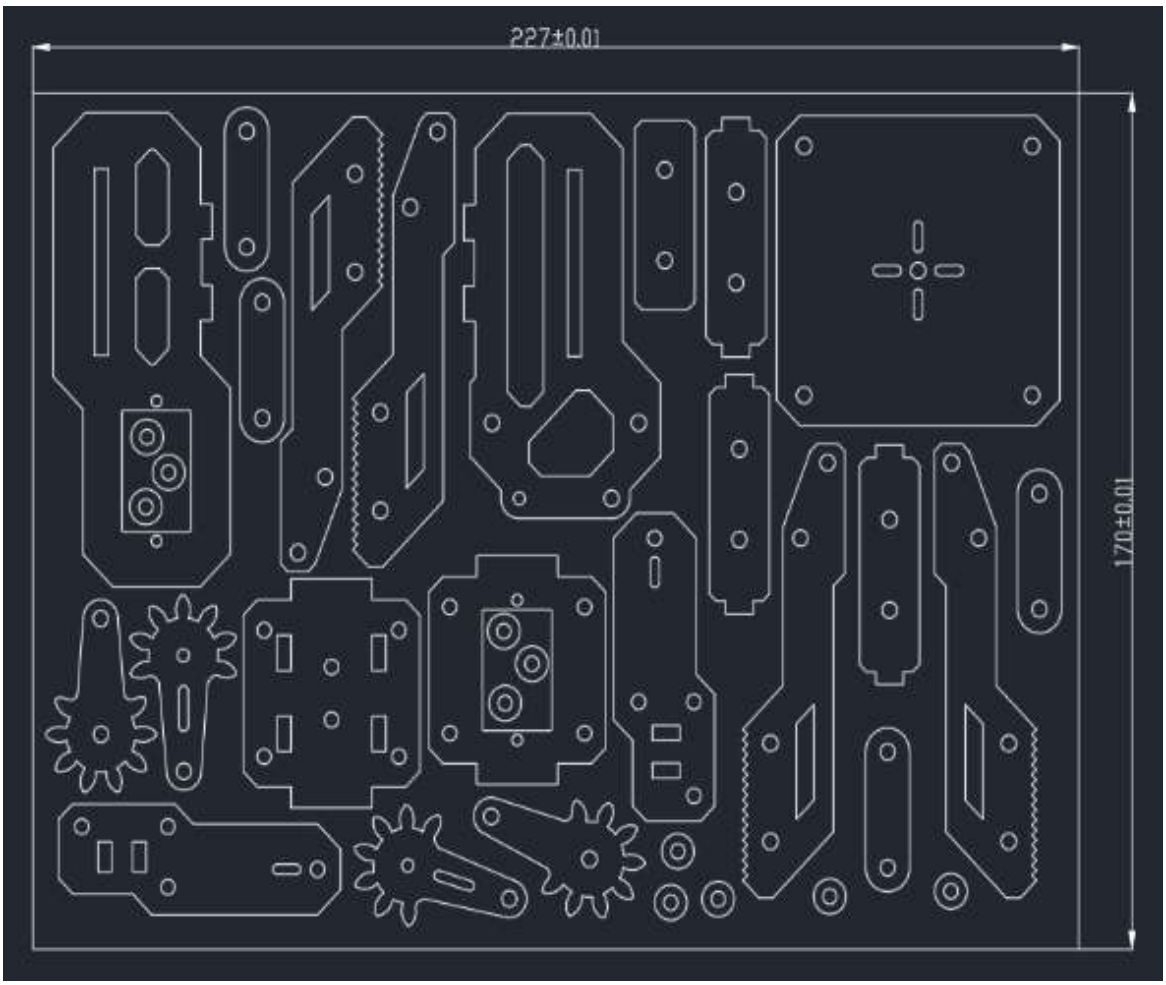
Diagrama 3D	Descripción	Diagrama 3D	Descripción
	Estructura principal del chasis del robot.		Soporte de las tarjetas, Raspberry y RobotHAT.
	Soporte móvil de las ruedas delanteras para ejecutar los giros.		Base móvil para acoplarse con la primera articulación del brazo robótico.
	Base fija de las luces de iluminación.		Acople de la primera articulación que une al robot con el brazo.

Nota: Se muestran los diagramas modelados en 3D de los elementos que conforman el chasis del robot. Figuras tomadas de (Adeept, 2021)

Sistema mecánico. Lo diagramas mecánicos de los elementos que proporciona el Kit PiCar Pro está distribuido en dos partes, la primera que es un modelado en 3D de los elementos que conforman el chasis construido en aleación de aluminio presentados en la tabla 12 y la segunda el modelado en 2D de la lámina de acrílico que contiene las piezas que conforman el brazo robótico mostrado en la figura 38.

Figura 38

Piezas que conforman el brazo robótico en lámina de acrílico.

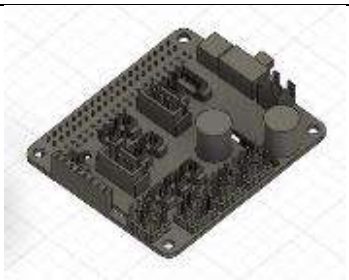
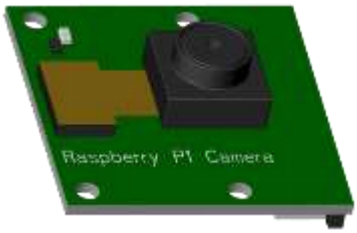


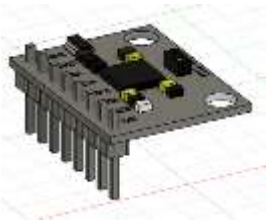
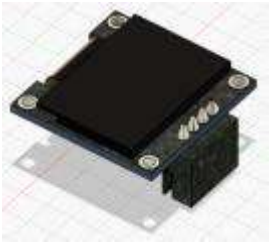
Nota: Figura tomada de (Adeept, 2021)

Sistema electrónico. El sistema electrónico con el que cuenta el Kit está formado por todos los elementos que se muestran a continuación:

Tabla 13

Elementos electrónicos del Kit PiCar Pro

Elemento	Descripción	Imagen
RobotHAT driver board	Esta tarjeta es la intermediaria de la comunicación entre el controlador del robot, en este caso una raspberry y los sensores y actuadores.	
Cámara	Módulo de cámara para Raspberry Pi	
Servomotores pequeños	Tres servomotores modelo AD002 con capacidad de giro de 180º	
Servomotores grandes	Cuatro servomotores modelo MG995 con capacidad de giro de 180º	
Motores	Dos motores modelo JGA25-370 DC12V 130RPM	

Elemento	Descripción	Imagen
Módulo ultrasónico	Un módulo ultrasónico, para medición de distancia	
LED Strip	Módulo de leds para combinación de colores	
Acelerómetro y giroscopio	Módulo que incluye acelerómetro y giroscopio modelo MPU6050	
Luces Leds	Dos luces leds grandes, para iluminación del entorno	
Pantalla OLED	Pantalla OLED para visualización de mensajes	
Cables de conexión	• Un bus de tres cables • Dos buses de cuatro cables • Cable de conexión de la cámara • Cables de servomotores	

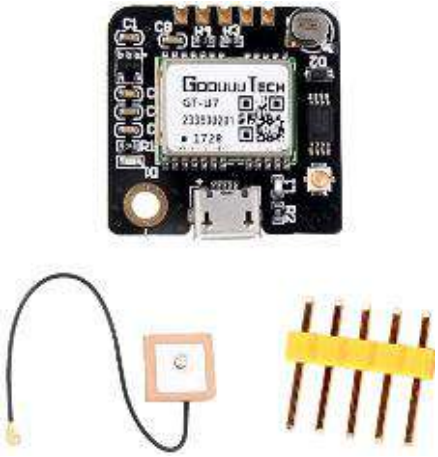
Elemento	Descripción	Imagen
	• Cables de motores • Extensiones de cables	

Nota: Se muestran los elementos electrónicos a detalle que vienen en el Kit PiCar Pro. Figuras tomadas de (Adeept, 2021).

Como se puede observar, el Kit Incluye sensores, actuadores, entre otros que forman parte de la instrumentación que el robot requiere para adaptarse a este aplicativo. A estos se añadirá un módulo GPS que se muestra en la figura 39, el mismo que no está incluido dentro del Kit, pero que se lo adaptará para de esta manera dar un mayor realce y robustez al sistema desde el punto de vista de orientación.

Figura 39

Módulo GPS GT-U7



Nota: Figura tomada de (Geekstory, 2021).

Fuente de energía. La fuente de energía será una batería DC, por lo que necesitamos dimensionar la cantidad de elementos que estarán conectados para determinar las características que deberá tener. En la tabla 14 se muestran los elementos a los que alimentará la batería con su respectivo consumo en corriente.

Tabla 14

Consumo de corriente de los elementos electrónicos que conforman el robot móvil

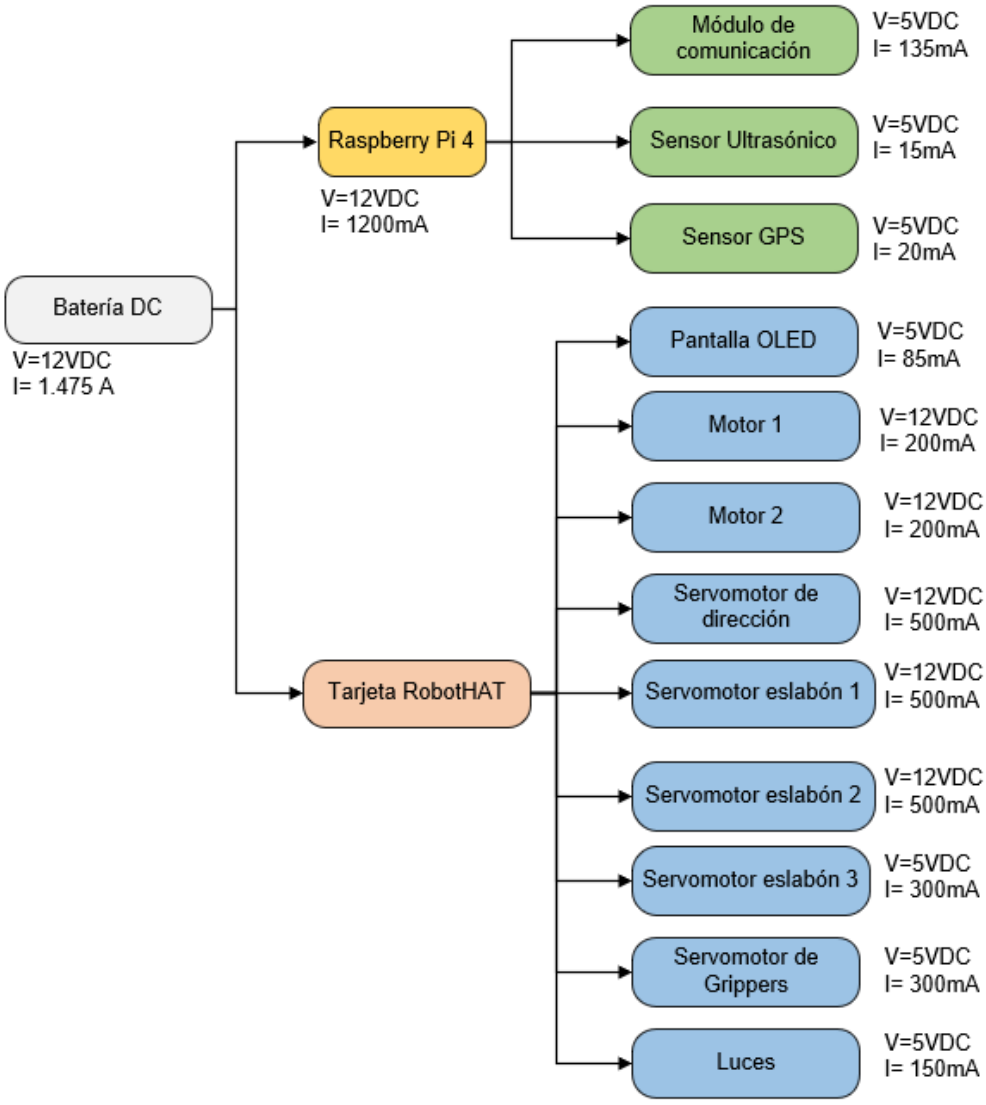
Elemento	Cantidad	Corriente individual	Consumo de corriente total
Raspberry Pi 4	1	1200mA	1200m A
Motor JGA25-370	2	200mA	400mA
Servomotor AD002	2	300mA	600mA
Servomotor MG995	3	500mA	1500mA
Cámara Raspberry Pi	1	30mA	30mA
Sensor Ultrasónico	1	15mA	15mA
Sensor GPS	1	20mA	20mA
Pantalla OLED	1	85mA	85mA
Luz LED	5	30mA	150mA
Circuito de comunicación (ESP32)	1	135mA	135mA
		TOTAL	**4.135A**

Nota: Se detalla el consumo de cada uno de los elementos electrónicos que formarán parte del robot móvil.

Realizado el análisis de consumo detallado anteriormente, se tendrá que buscar en el mercado una batería que cumpla con ciertos parámetros, como proporcionar mínimo una corriente de 4.135 A, y un voltaje de entre 5 VDC a 12 VDC, siendo este último el que mayor eficiencia brindaría a los elementos para que funcionen todos al 100% de su capacidad. El suministro de energía a cada uno de los elementos se distribuirá como se muestra en la figura 40.

Figura 40

Distribución de energía de la Batería



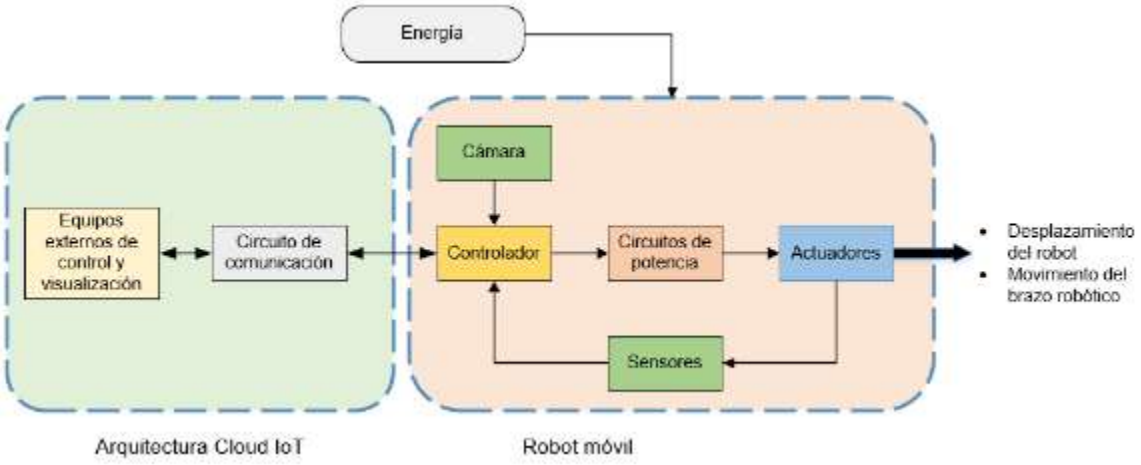
Nota: Se muestra la distribución de energía proporcionada por la batería, con los valores de consumo de cada elemento.

Controlador. Los elementos electrónicos detallados anteriormente deben interconectarse de manera adecuada para ejecutar los procesos de control que cumplan con los requerimientos del usuario en base a criterios técnicos. En la figura 41 se muestra

el lazo de control general que contendrá el controlador que en este caso será una Raspberry Pi.

Figura 41

Lazo de control del robot Móvil



Nota: El lazo de control está establecido por bloques, donde cada uno contiene uno o varios elementos electrónicos.

A continuación, se detallan el o los elementos que están dentro de cada uno de los bloques del lazo de control:

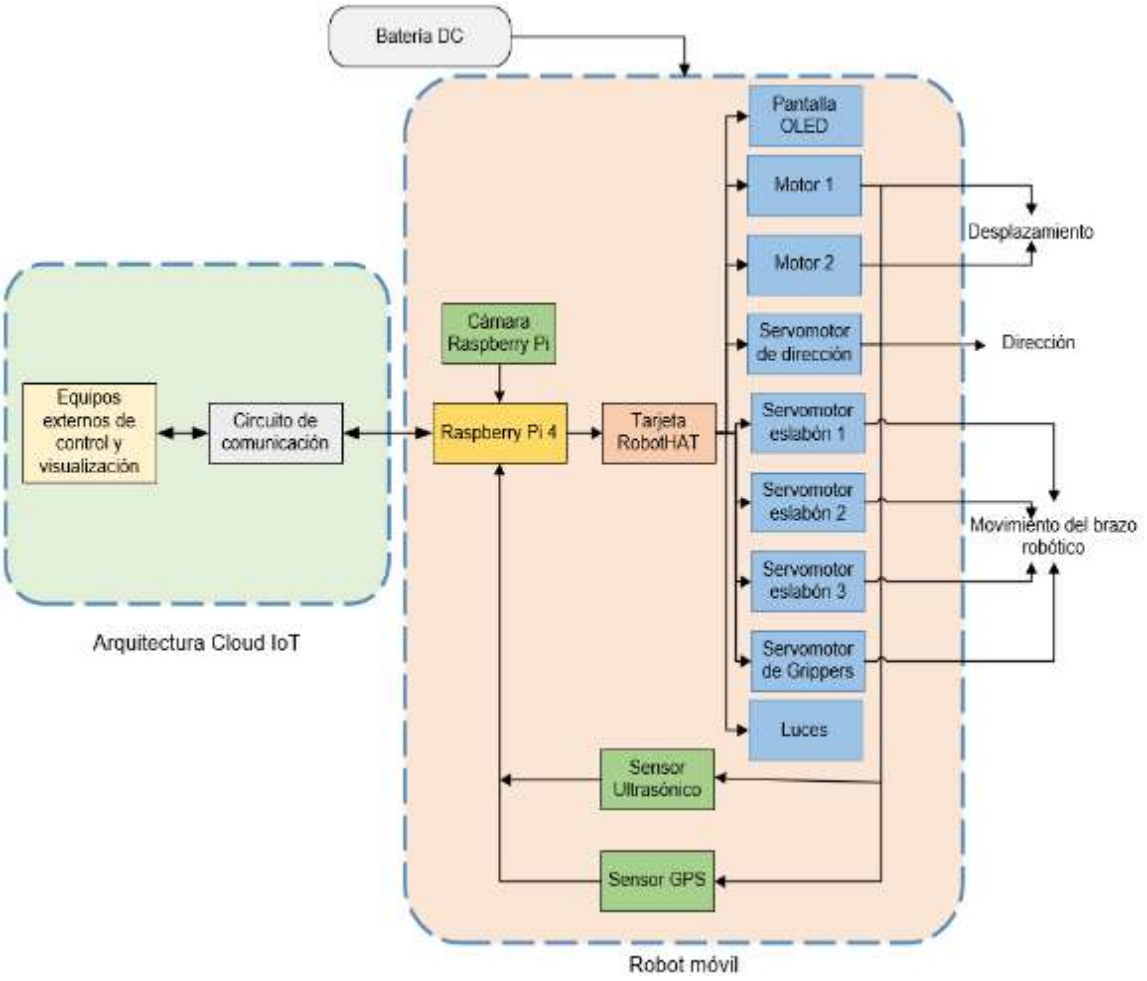
- **Controlador:** El controlador será la tarjeta Raspberry Pi 4 modelo B 2019.
- **Circuitos de potencia:** Los circuitos de potencia para los actuadores vienen incorporados dentro de la tarjeta RobotHAT por lo que esta será la encargada de funcionar como driver de los actuadores.
- **Actuadores:** Entre los actuadores tenemos los motores, luces y servomotores tanto del robot como del brazo robótico.
- Sensores: Los sensores que componen este bloque son el sensor ultrasónico y el sensor GPS.
- **Cámara:** Este bloque está formado por el módulo de la Cámara Pi.

- **Circuito de comunicación:** Este bloque se lo tratará a fondo más adelante ya que es parte de la arquitectura Cloud IoT.
- **Energía:** El robot tendrá la energía proporcionada por una batería externa con características que se enuncian anteriormente.

Teniendo los elementos que conforman cada uno de los bloques, en la figura 42 se muestran los elementos a detalle que conformarán la estructura electrónica del robot móvil.

Figura 42

Lazo de control a detalle del robot Móvil

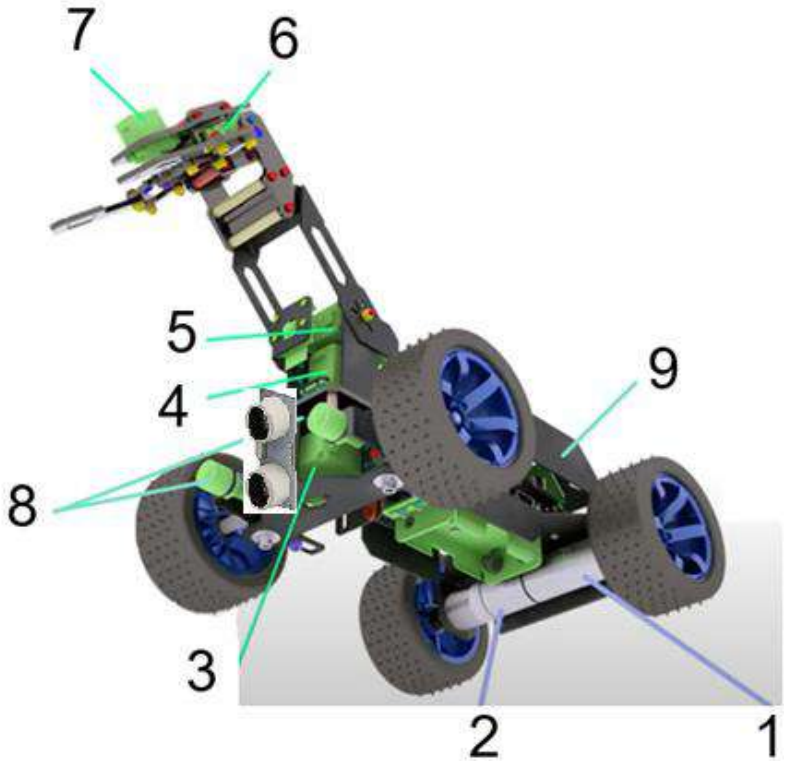


Nota: En este lazo de control se muestran a detalle los elementos que lo componen para posteriormente realizar la conexión donde corresponda.

Etapa 6. Esquema definitivo

Figura 43

Esquema final en 3D que acopla la estructura mecánica con los elementos electrónicos



Nota: Figura tomada y adaptada de (Adeept, 2021)

Tabla 15

Elementos electrónicos correspondientes a la numeración de la figura 43

Nº	Elemento
1	Motor JGA25-370 izquierdo
2	Motor JGA25-370 derecho
3	Servomotor de dirección
4	Servomotor de la primera articulación del brazo robótico
5	Servomotor de la segunda articulación del brazo robótico
6	Servomotor de la tercera articulación del brazo robótico

Nº	Elemento
7	Servomotor de los Grippers del brazo robótico
8	Luces LED y sensor ultrasónico ubicados en la parte frontal
9	Estructura que contiene la Raspberry, tarjeta RobotHAT, sensor GPS, módulo de comunicación, pantalla OLED, batería y conexiones necesarias.

Nota: En esta tabla se detallan los requerimientos del Usuario para referenciar el diseño.

Diseño de la arquitectura Cloud IoT

Como se expuso en la justificación, la arquitectura para el presente proyecto requiere de 4 capas que constituyen toda la arquitectura Cloud IoT las cuales son determinadas como Edge Computing, Fog Computing, Cloud Computing y la interfaz de visualización que es el Front-End, cada una tiene sus propios requerimientos, funciones y secciones donde se maneja la comunicación de datos a partir de un formato general de texto llamado JSON que se constituye básicamente del parámetro y el valor que corresponde al mismo, de esa manera se puede enviar información entre las diferentes capas y se utiliza a conveniencia de cada una. A continuación, se tiene el proceso de diseño de cada capa.

Etapa 1. Requerimientos

Los requerimientos están diferenciados para cada una de las capas que conforman la arquitectura Cloud IoT y se los observa en la tabla 16 para que desde ahí se puedan identificar sus características, funciones y módulos para entender cómo se estructuran de forma individual y luego unirlo todo.

Tabla 16

Requerimientos de la arquitectura Cloud IoT

Nº	Requerimiento
	Capa Edge Computing
1	Adaptar al robot un módulo de comunicación para largas distancias.
2	Recepción de instrucciones para el control de desplazamiento y dirección del robot.
3	Recepción de instrucciones para el control del brazo robótico.
4	Control y manipulación del robot móvil y brazo robótico.
5	Monitoreo del entorno y envío de datos de sensores del robot móvil.
6	Envío de datos de video para la generación de Streaming.
7	Diseño de una arquitectura expandible con el fin de adaptar más robots móviles.
	Capa Fog Computing
1	Establecer la comunicación entre las capas Edge y Cloud Computing.
2	Diseñar el software y hardware para la comunicación.
3	Comunicación a largas distancias
4	Comunicación inalámbrica
	Capa Cloud Computing
1	Modo de funcionamiento en el que se pueda utilizar más de un servicio a la vez en tiempo real.
2	Gestionar datos según peticiones.
3	Servicio de video streaming.
4	Servicio de reconocimiento de imágenes y base de datos.

Nº	Requerimiento
5	Servicio de geolocalización del robot.
6	Servicio para el tratamiento de información de sensores e instrucciones.
7	Servicio de mensajería.
8	Orquestación de los microservicios.
9	Fácil entendimiento, modificación y adaptación a nuevos servicios.
10	Costo de microservicios.

Nota: En esta tabla se presentan los requerimientos para cada uno de las capas que conforman la arquitectura Cloud IoT.

Tabla 17

Características de la arquitectura Cloud IoT

Nº	Características
	Capa Edge Computing
1	Parámetros de desplazamiento y dirección del robot más óptimos para aplicaciones de exploración de cualquier tipo de superficie.
2	Parámetros para el movimiento de las articulaciones del brazo robótico con el fin de manipular objetos.
3	Utilización de los protocolos de comunicación más óptimos entre los módulos (controlador, módulo de comunicación, sensores) para tener la mejor eficiencia.
4	El software y hardware deben ser escalables, con el fin de que la arquitectura pueda crecer con la interconexión de más dispositivos.

Nº	Características
5	Generación de video en tiempo real desde la cámara interconectada con el controlador, para su posterior envío a la siguiente capa de la arquitectura.
6	Lectura y empaquetamiento de datos de sensores para su posterior envío.
	Capa Fog Computing
1	Protocolos de comunicación a larga distancia con mayor eficiencia para la aplicación en este sistema.
2	Uso de módulos de comunicación garantizados técnicamente bajo normas.
3	Uso de protocolos de comunicación inalámbrica que brinde las prestaciones adecuadas para la aplicación.
4	Creación de Gateway's para establecer los enlaces de comunicación según se vaya requiriendo.
	Capa Cloud Computing
1	Forma de funcionamiento paralela donde los datos se utilizan a conveniencia por cada servicio y se den respuestas a los requerimientos en tiempo real, lo cual se posible de visualizar en el Front-End o en acciones en la capa Edge Computing.
2	Los datos llegan en formato JSON con la información tanto de sensores, peticiones e instrucciones que se utilizan por cada servicio solamente con la llamada de un parámetro específico.
3	El servicio de video streaming será tratado con protocolos HTTP para que se pueda llamar directamente con un requerimiento a la

Nº	Características
	ruta donde se tiene el servicio y se tenga como respuesta el video en tiempo real, se tiene que la tecnología de acceso para este caso es WiFi por mayor velocidad y ancho de banda.
4	Se realiza una captura del video streaming la cual es almacenada por un servicio de base de datos y luego con otro servicio realizar el reconocimiento de imágenes que funciona con una configuración de cliente REST pues estos servicios funcionan como una API REST en un sistema RESTful.
5	También es con el uso de un cliente REST pues el servicio funciona como un sistema RESTful al cual los requerimientos llegan en formato JSON con los parámetros de latitud y longitud para poder mostrar la localización del robot.
6	La información de sensores e instrucciones se envía en formato JSON para que pueda ser utilizado para visualizar en el HMI o para controlar el robot utilizando para el segundo el protocolo MQTT.
7	Servicio de mensajería utilizará el protocolo MQTT donde los datos son administrados desde los recibidos del Gateway utilizado en la capa Fog Computing y para enviarlos hacia el mismo en formato JSON.
8	La orquestación se realiza con herramientas que puedan conectar tantos servicios, API REST y hardware, es en base a un editor que agilice la interconexión de flujos y codificación de las rutas.
9	El entorno en donde se realice la conexión de flujos entre la parte hardware y servicios está estructurado por partes por lo que si un

Nº	Características
	servicio se desea modificar sea fácil de localizar y editar la sección donde se orquesta el uso de dicho servicio.
10	Los proveedores de servicios cobran por su uso, sin embargo, siempre tienen espacios de almacenamiento o limitantes de tiempo en ofertas gratuitas las cuales son las principales a utilizar en el presente proyecto.

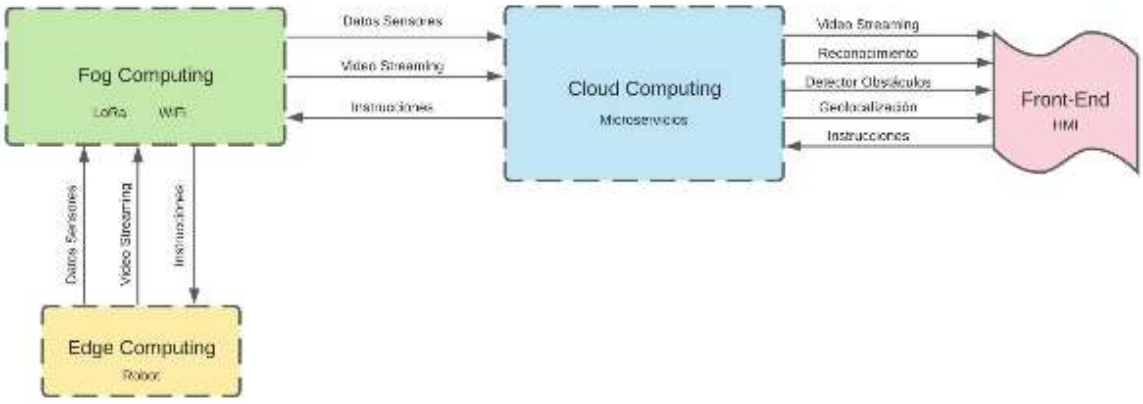
Nota: En esta tabla se presentan las características correspondientes a cada requerimiento para cada capa que conforman la arquitectura Cloud IoT.

Etapa 2. Funciones y Estructuras

Se determina el diagrama expuesto en la figura 44 en donde se puede observar cómo se encuentran comunicándose cada capa y las entradas y salidas con las que cuenta cada una para luego analizar los diagramas por separado de cada capa y conocer las subfunciones que se llevan a cabo en cada una.

Figura 44

Entradas y salidas de cada capa en la arquitectura Cloud IoT



Nota: Representación en bloques de cómo serían las entradas y salidas en forma general de las capas Edge Computing, Fog Computing y Cloud Computing.

Ahora se procede a realizar un análisis de las sub-funciones que se encuentran en cada una de las capas, es decir qué procesos se están dando en el interior de cada capa para conocer cómo se están incorporando y luego poder encontrar soluciones para poder ser implementadas.

Edge Computing. La capa de Edge Computing será la encargada de procesar y almacenar información a partir del robot móvil, sensores entre otros dispositivos IoT de forma local, es decir, lo más cerca posible de los dispositivos finales o de campo y sensores generadores de datos.

Otra de las funciones que debe ejecutar esta capa es la de recepción de instrucciones para el funcionamiento de los actuadores, que en este caso estarán acoplados a la parte mecánica del robot móvil, que servirán para dar el desplazamiento y dirección al robot, así como también la movilidad del brazo robótico para la manipulación de objetos.

El orquestador de las funciones, tanto de transmisión como de recepción será la Raspberry Pi, que de igual forma está establecida como controlador del robot. De esta manera todas las funciones necesarias de software para el procesamiento de datos estarán contenidas dentro de la Raspberry, que en otras palabras será la parte central de esta capa, es por eso que las conexiones de sensores y actuadores, así como también de otros módulos necesarios para la comunicación, estarán conectados de forma estructurada y organizada en las entradas y salidas de esta tarjeta.

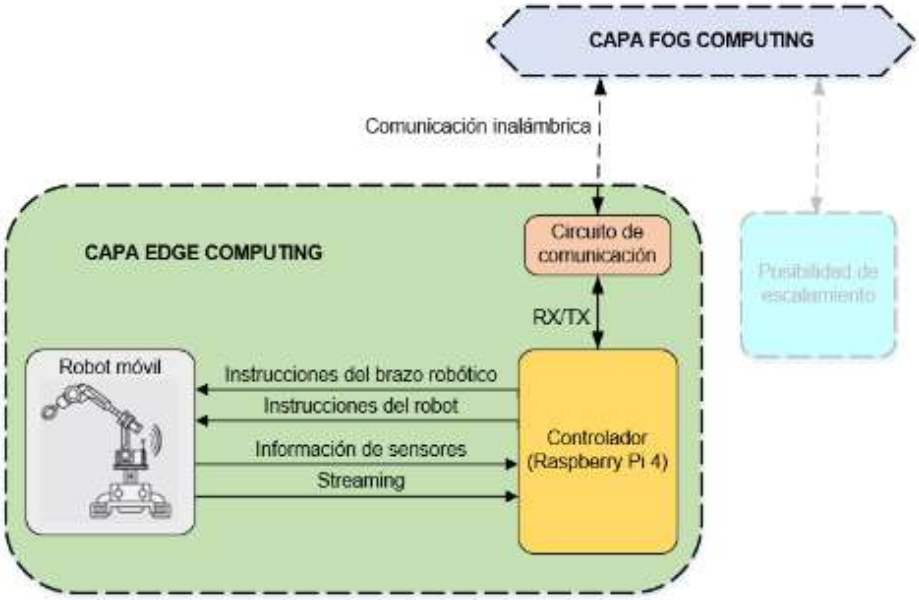
La estructura establecida de esta capa debe ser escalable, es decir el software y hardware deben estar diseñados con el propósito de poder interconectarse con más dispositivos a futuro, especialmente con otro robot móvil, para en investigaciones futuras lograr formar una red orientada a la robótica colaborativa.

El módulo de comunicación debe estar conectado directamente con el controlador para evitar tiempos de latencia causados por el hardware, en este caso será necesario

buscar un módulo que pueda ser utilizado para comunicación a largas distancias, tanto en transmisión como en recepción, para de esta manera lograr la tele-operación del robot a distancia, y a la vez del envío de datos desde esta capa hasta las capas siguientes de forma inalámbrica. En la figura 45 se muestra la estructura en forma de bloques, que debe manejar esta capa para cumplir con el funcionamiento requerido y ejecutar las funciones que se plantean anteriormente.

Figura 45

Estructura de la capa Edge Computing



Nota: Representación en bloques de las funciones que se llevan a cabo en la estructura de la capa Edge Computing

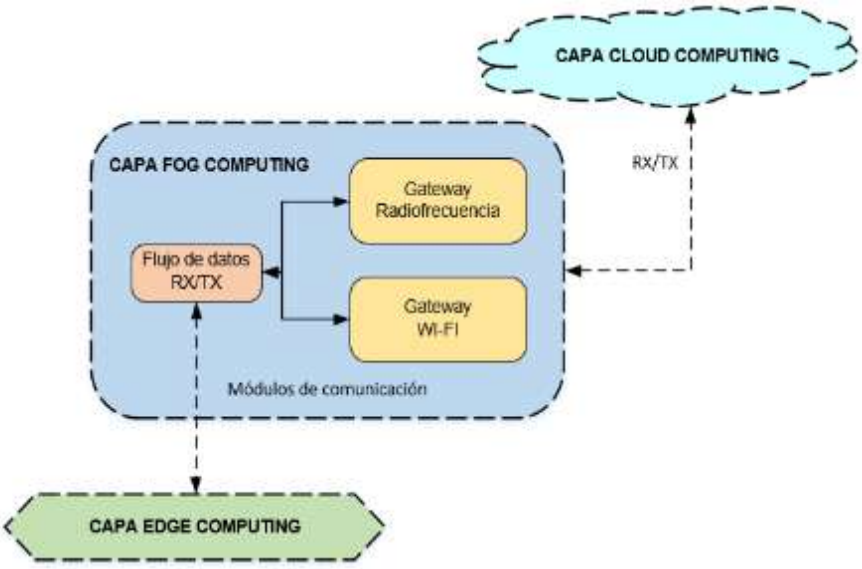
Fog Computing. La capa de Fog Computing se encuentra entre la capa de Edge y Cloud, siendo esta la encargada de comunicar las dos capas, es por ello que está compuesta por los módulos de comunicación. La comunicación debe ser inalámbrica, es por eso que se tiene previstos utilizar los protocolos de comunicación inalámbrica más usados, como son WIFI y radiofrecuencia, por lo que se debe levantar la infraestructura tanto en software como en hardware para la implementación de estos servicios.

El requerimiento principal de esta capa es que se pueda realizar una tele-operación a larga distancia, por lo que se traduce en que la comunicación debe establecerse en rangos de distancia altos. De esta manera los protocolos de comunicación deben ser estructurados para poder funcionar en grandes distancias, considerando todos los factores, como la ubicación, la geografía del lugar, la posición entre otros factores que favorecerán o perjudicarán este fin. Como consecuencia se debe realizar anticipadamente un análisis de los módulos en hardware más óptimos y con mayor eficiencia.

Las tecnologías de comunicación inalámbricas necesitan de puertas de enlace o Gateway's para la transmisión o recepción de datos, estos dispositivos deben estar implantados y configurados dentro de la arquitectura de esta capa, para posteriormente poder acceder a otra red, que en este caso permitirá el acceso a la capa de Cloud. La estructura de funcionamiento en base a estos requerimientos, ilustrados en la figura 46.

Figura 46

Estructura de la capa Fog Computing



Nota: Representación en bloques de las funciones que se llevan a cabo en la estructura de la capa Fog Computing

Cloud Computing. Esta capa se encuentra trabajando utilizando el Internet y es aquí en donde se va a realizar la orquestación de todos los servicios que se van a utilizar de manera que se supla la aplicación objetivo para este proyecto, en esta capa se encuentra involucrados:

Servicios para el video streaming. Funciona mediante protocolos HTTP pues necesita obtener el video a partir de la cámara del robot utilizando la tecnología de acceso de WiFi creando así un servidor local desde el robot utilizando la Rasberry Pi, para que este servidor streaming pueda ser utilizado en cualquier plataforma lo que se recomienda es utilizar un túnel o puente de acceso el cual usando nuevamente protocolos HTTP permite visualizar el video streaming de un servidor local desde cualquier red que no sea la local del robot.

Servicio de reconocimiento de imágenes y base de datos. Gracias al proveedor AWS de Amazon se obtuvieron de forma gratuita el servicio de base de datos Amazon S3 que ofrece un espacio gratuito de 5Gb y el servicio de reconocimiento de imágenes AWS Rekognition con un límite de 5000 imágenes gratis para analizar, lo propuesto en el diseño es tomar una captura del video streaming como un cliente API REST para guardar la imagen en la base de datos, luego llevar a analizar en el servidor de reconocimiento de imágenes y para terminar extraer los parámetros y el porcentaje de confiabilidad para poder mostrarlo en el HMI. No está demás mencionar que estos servicios funcionan como un sistema RESTful.

Servicio de Geolocalización. Con los datos de latitud y longitud obtenidos desde el sensor y en formato JSON se toman estos dos parámetros para luego poderlos enviar al servidor de mapa mundial al cual se lo puede llamar utilizando el nodo worldmap, el uso de este servicio no tiene costo alguno pues es de libre acceso.

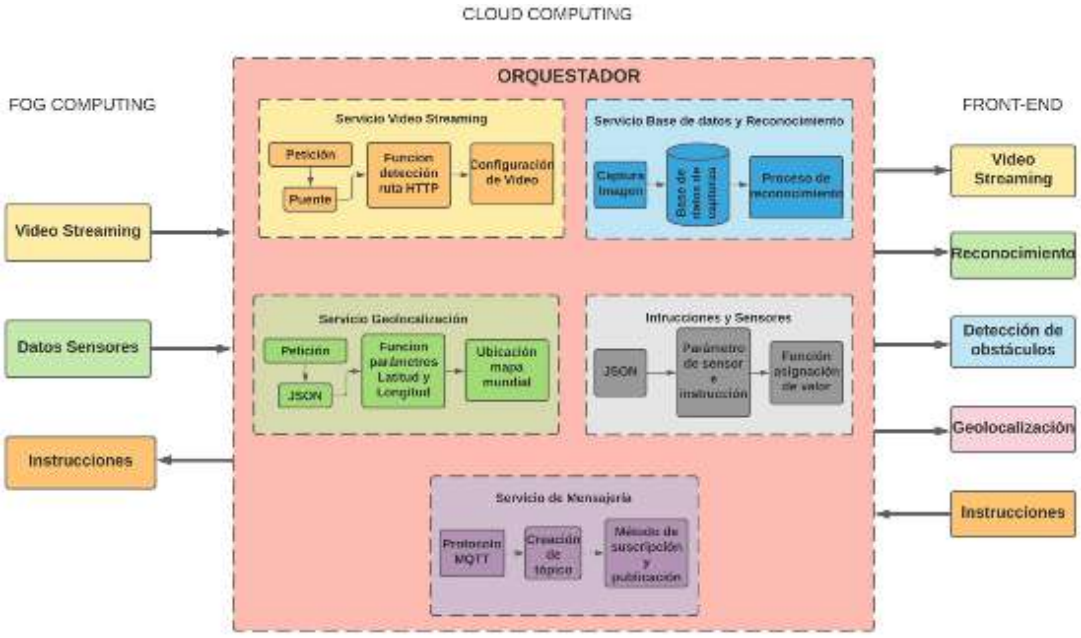
Sensores e Instrucciones. Los datos de sensores viene de igual manera con su respectivo parámetros dentro de un formato JSON, del cual se puede extraer la

información del mismo mediante el uso de funciones en JavaScript para que mediante la orquestación se lo pueda enviar al front-end, en cambio las instrucciones vienen desde el front-end a los cuales se les asigna un valor específico que se asigna a un mismo parámetro el cual en formato JSON se envía desde la capa Cloud Computing utilizando un servicio de mensajería para transmitirlo hasta la capa Fog Computing.

Servicios de mensajería. Principalmente es para la comunicación entre la capa de Cloud Computing con la capa Fog Computing, se planea utilizar un servicio con protocolos MQTT definiendo un tópico en la capa cloud para que los dispositivos utilizados en la capa fog puedan suscribirse a dicho tópico para publicar mensajes y a la vez funcionar en viceversa para recibir mensajes teniendo así comunicación bidireccional entre estas dos capas utilizando un servicio de mensajería con una calidad de servicio de nivel 1 de manera que se tenga una confirmación de que los mensajes fueron recibidos.

Figura 47

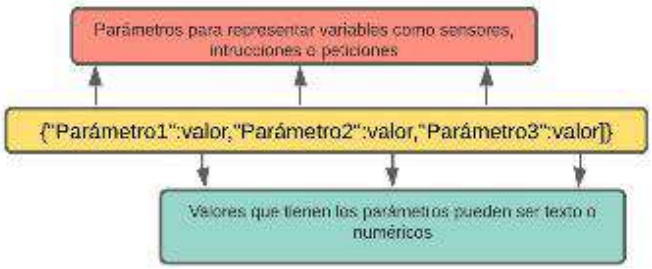
Subfunciones en la capa Cloud Computing



Nota: Representación en bloques de las funciones que se llevan a cabo por cada servicio en la capa Cloud Computing.

Figura 48

Diseño del JSON para envío de datos



Nota: Representación de cómo se estructura los JSON para poder realizar la transmisión de información entre las capas de la arquitectura Cloud IoT

Etapa 3. Principios de solución y variables

Edge Computing.

Tabla 18

Soluciones para el diseño de la capa Edge Computing

Sub-funciones	Soluciones
Instrucciones de desplazamiento y dirección del robot.	Recepción de instrucciones a través de identificadores de función.
Instrucciones para el movimiento de las articulaciones del brazo robótico.	Recepción de instrucciones a través de identificadores de función.
Comunicación entre los módulos controlador, módulo de comunicación, sensores.	Conexión física con cables y terminales según corresponda.
Software y hardware escalables, con el fin de que la arquitectura pueda crecer con la interconexión de más dispositivos.	Realizar una programación de software modular y dejar entradas y salidas disponibles para futuras conexiones.

Sub-funciones	Soluciones
Generación de video en tiempo real (Streaming)	Programación para la generación y envío de video continuo en tiempo real desde la cámara ubicada en el robot.
Lectura y empaquetamiento de datos de sensores para su posterior envío.	Uso de protocolos de comunicación.

Nota: Esta tabla da a conocer las soluciones a las problemáticas presentadas en el diseño de la capa Edge Computing.

Fog Computing.

Tabla 19

Soluciones para el diseño de la capa Fog Computing

Sub-funciones	Soluciones
Comunicación a larga distancia	Uso de módulos de comunicación con una alta eficiencia de transmisión y recepción de datos a distancia.
Protocolos de comunicación inalámbrica	Emplear el software adecuado basado en protocolos de comunicación para un correcto manejo de datos de forma inalámbrica.
Gateway's para establecer los enlaces de comunicación según se requiera.	Emplear el hardware que brinde las prestaciones adecuadas para proporcionar las puertas de enlace de los dispositivos y módulos utilizados.

Nota: Esta tabla da a conocer las soluciones a las problemáticas presentadas en el diseño de la capa Fog Computing.

Cloud Computing.

Tabla 20

Soluciones para el diseño de la capa Cloud Computing

Sub-funciones	Soluciones
Formato envío de datos de sensores e instrucciones	Formato de texto JSON
Herramienta de interconexión de servicio y orquestación	Node-RED y NODEJS
Servicio Video Streaming	Servidor local Rasberry Pi 4
Servicio Base de datos	Servicio Amazon S3
Servicio Reconocimiento de imágenes	Servicio ASW Rekognition
Servicio geolocalización	Servicio Web Map
Servicio de mensajería	Broker MQTT Mosquitto
Recursos adicionales	Internet, computador con Windows 10

Nota: Esta tabla da a conocer las soluciones a las problemáticas presentadas en el diseño de la capa Cloud Computing.

Etapa 4. Módulos

Módulo 1. Capa Edge Computing

- Control del Robot móvil.
- Control del brazo robótico.
- Procesamiento de datos de sensores y actuadores.

Módulo 2. Capa Fog Computing

- Módulos de comunicación.
- Servicio de Tele-operación (Sender-Receiver Bidireccional).

- Servicio de mensajería MQQT en el Gateway.

Módulo 3. Capa Cloud Computing

- Servicio video streaming
- Servicio de reconocimiento de imágenes y base de datos.
- Servicio de geolocalización.
- Sensores e Instrucciones.
- Servicios de mensajería.

Etapa 5. Esquema de los módulos

Módulo 1. Capa Edge Computing

***Control del Robot móvil*.** El control de desplazamiento y dirección del robot, se lo realizará a través de la recepción de identificadores de función, que en este caso serán caracteres, para que posteriormente el controlador discrimine el carácter y gestione la función que realizarán los actuadores del robot para brindar el desplazamiento y el cambio de dirección. Las funciones se relacionarán con los caracteres como se puede observar en la tabla 21.

Tabla 21

Función a ejecutar y carácter correspondiente.

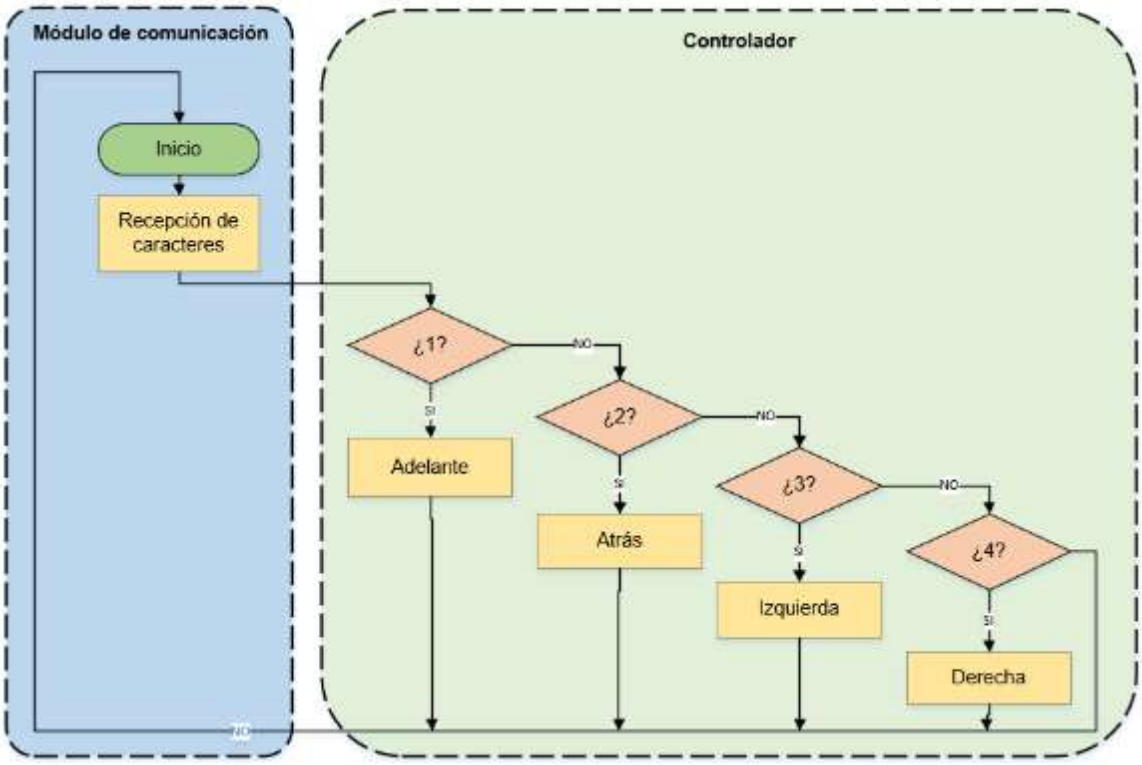
Función	Caracter
Adelante	1
Atrás	2
Izquierda	3
Derecha	4

Nota: Función que ejecutará el robot con su respectivo caracter discriminante.

Teniendo los datos de funciones con su respectivo caracter, el proceso de desplazamiento y dirección que deberá ser implementado en software se muestra en la figura 49.

Figura 49

Instrucciones de desplazamiento y dirección



Nota: Representación del proceso de desplazamiento y dirección en base a caracteres recibidos.

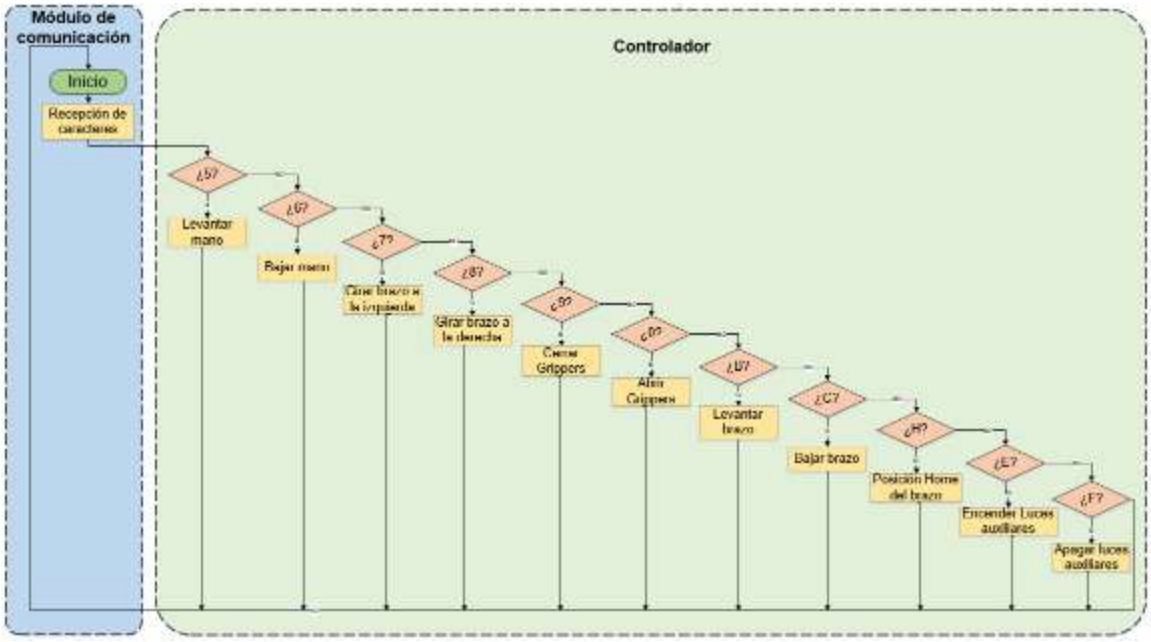
Control del brazo robótico. Para el control del brazo robótico, se utiliza la misma lógica mostrada anteriormente, ya que el controlador recibirá el carácter, y lo discriminará de acuerdo a la función que debe realizar, las funciones con el carácter correspondiente se muestran en la tabla 22.

El proceso de control del brazo robótico inicia con la recepción de instrucciones desde el módulo de comunicación, posteriormente el software del controlador es el encargado de enviar la instrucción que corresponda al actuador, que en este caso serán

los servomotores que estarán acoplados a la parte mecánica del brazo robótico, con el fin de manipular objetos, el proceso de manipulación se detalla en la figura 50.

Figura 50

Instrucciones para la manipulación del brazo robótico



Nota: Representación del proceso de manipulación del brazo robótico en base a caracteres recibidos.

Tabla 22

Función a ejecutar y carácter correspondiente.

Función	Caracter
Levantar mano	5
Bajar mano	6
Girar brazo a la izquierda	7
Girar brazo a la derecha	8
Cerrar Grippers	9
Abrir Grippers	0

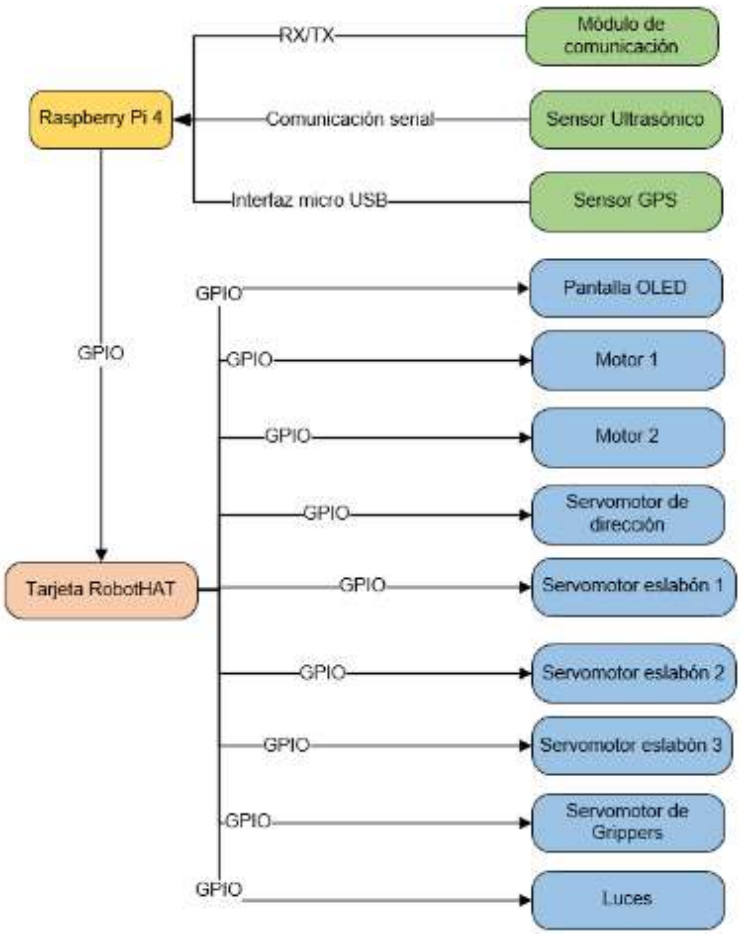
Función	Caracter
Levantar brazo	B
Bajar brazo	C
Posición Home del brazo	H
Encendido de luces auxiliares	E
Apagado de luces auxiliares	F

Nota: Función que ejecutara el robot con su respectivo caracter discriminante

Procesamiento de datos de sensores y actuadores.

Figura 51

Detalle de transmisión de datos para su posterior procesamiento.



Nota: Representación de la dirección y características del procesamiento de datos.

El sensor ultrasónico para medición de distancia y el GPS, para determinar la posición son los sensores incorporados en el robot, donde cada uno maneja independientemente su protocolo de comunicación para la transmisión de datos. Los otros dispositivos incorporados son los actuadores, que tienen sus señales de activación de acuerdo a las necesidades del usuario, todas estas características de transmisión y recepción para el procesamiento de datos se detalla en la figura 51. Por último, tenemos interconectado en esta capa el módulo de comunicación, que es el encargado de receptar y transmitir los datos de forma inalámbrica, su análisis se realiza a profundidad en la capa siguiente.

Módulo 2. Capa Fog Computing

Módulos de comunicación. En base a los requerimientos de comunicación, se hizo un análisis y búsqueda de los módulos de radiofrecuencia disponibles en el mercado, encontrando una gran variedad, con diferentes características y calidad. Para la selección del módulo se tomó en cuenta la calidad, por esta razón se buscó una marca conocida que cuente con certificaciones y normas de calidad, pero también que cumpla con la función de comunicación a largas distancias, seleccionando la tarjeta ESP32 Wireless Stick Lite fabricada por Heltec Automation.

Figura 52

ESP32 Heltec Wireless Stick Lite

Nota: Figura tomada de (HELTEC, 2021)

Tabla 23

Características de la tarjeta

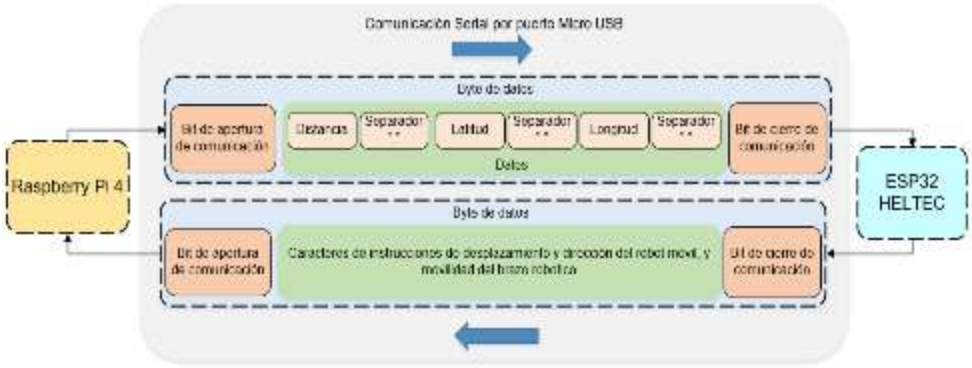
Nº	Características
1	Microprocesador: ESP32-PICO (MCU de 32 bits de doble núcleo + núcleo ULP + FLASH interno de 4 MB), con chip de nodo LoRa SX1276.
2	Interfaz micro USB con regulador de voltaje completo, protección ESD, protección contra cortocircuitos, blindaje RF y otras medidas de protección
3	Interfaz de batería SH1.25-2 a bordo, sistema de gestión de batería de litio integrado (gestión de carga y descarga, protección de sobrecarga, detección de energía de batería, conmutación automática de energía de batería / USB).
4	WiFi integrado, LoRa, tres conexiones de red Bluetooth, Wi-Fi integrado, antena 3D de metal de 2,4 GHz dedicada a Bluetooth, interfaz IPEX (UFL).
5	CP2102 USB integrado a chip de puerto serie, conveniente para la descarga de programas, la impresión de información de depuración.
6	IDE de desarrollo de Arduino.
7	Admite la versión Arduino de la rutina del protocolo ESP32 + LoRaWAN proporcionada por Heltec. Este es un protocolo LoRaWAN estándar que puede comunicarse con cualquier puerta de enlace / estación base que ejecute el protocolo LoRaWAN (requiere activación del número de serie, solo el desarrollo de la empresa).
8	Con un buen diseño de circuito de RF y un buen diseño de bajo consumo (corriente de suspensión ≤ 30uA), es conveniente para los proveedores de aplicaciones de IoT verificar rápidamente las soluciones e implementar aplicaciones.

Nota: Información tomada de (HELTEC, 2021)

Como se puede observar en la tabla 23, las características de esta tarjeta son idóneas y cumple con todos los requerimientos necesarios para formar parte de esta arquitectura. Esta capa nace en la comunicación entre el controlador de la capa anterior y esta tarjeta, es por eso que se debe emplear un protocolo de comunicación para que las dos tarjetas en mención puedan interactuar. Como se enuncia en las especificaciones esta tarjeta tiene un puerto micro USB para comunicación serial, por lo que se usará este protocolo de comunicación para el intercambio de datos entre el controlador y el módulo de comunicación de forma bidireccional. Se realiza el empaquetamiento de datos, para establecer la comunicación antes mencionada y se realiza una transmisión y recepción ordenada como se muestra en la figura 53 lo que facilitará el desarrollo del software.

Figura 53

Arquitectura de intercambio de datos entre el controlador y el módulo de comunicación



Nota: Se muestra a detalle, la transmisión y recepción de datos entre el controlador que es la Raspberry Pi 4 y el controlador que es la tarjeta ESP32.

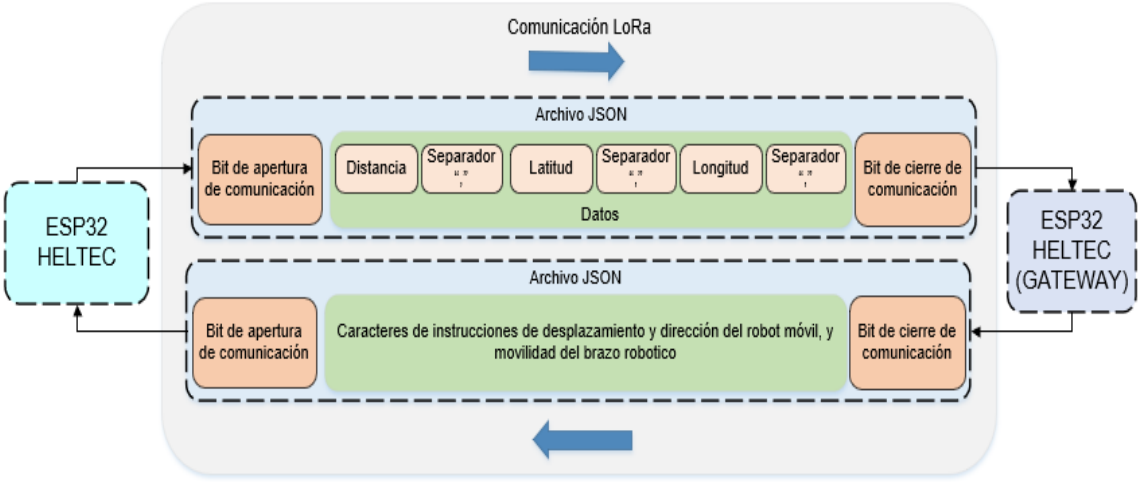
Otro de los módulos de comunicación que se ocupará en esta arquitectura es un router para tener acceso a una red local y poder comunicarse por WI-FI hacia el internet, con el fin de proporcionar a la capa de Edge Computing una conexión directa al internet desde su controlador, para fines de programación y algunos servicios requeridos para la implementación de esta arquitectura como lo es el Streaming.

Servicio de Tele-operación (Sender-Receiver Bidireccional). Otra de las características fundamentales por la cual se seleccionó la tarjeta ESP32 Heltec Wireless Stick Lite, es por su capacidad de comunicación por radiofrecuencia a unos cuantos centenares de metros, gracias a que viene implementada con el protocolo de comunicación LoRa, el cual a tenido un auge en la última década, con grandes prestaciones, especialmente en aplicaciones relacionadas con el IoT, es por eso que se implementara la comunicación inalámbrica a grandes distancias, con estos módulos.

Uno de los módulos estará instalado en el robot móvil, y receptará los datos del controlador como se detalla en la figura 54, los mismos que serán procesados para convertirlos en datos de formato JSON, este último será enviado de forma inalámbrica de acuerdo a las configuraciones que se establecerán en estos módulos, como frecuencia, ancho de banda, entre otros. Tendremos otro módulo que funcionará como Gateway, el cual será el encargado de recibir la información para posteriormente subirlo a la capa de Cloud a través de MQTT.

Figura 54

Arquitectura de intercambio de datos entre las tarjetas ESP32



Nota: Se muestra a detalle, la transmisión y recepción de datos entre la ESP32 del robot y la ESP32 que funcionará como GATEWAY.

Por otra parte, también este último módulo se encargará de recibir la información desde el Cloud y enviarla hacia el módulo que se encuentra en el robot, de esta manera se establecerá una comunicación bidireccional de forma inalámbrica. Estas dos tarjetas estarán comunicándose continuamente de la forma que se muestra en la figura 54, con el menor tiempo de latencia, para tener una comunicación optima.

Es importante destacar que estos módulos vienen con antenas de tipo resorte para ayudar a mejorar la eficiencia de la comunicación por radiofrecuencia. Por esta razón se debe tomar en cuenta la ubicación para tener la mayor línea de vista posible y la posición para evitar efectos de reflexión que podrían hacer perder la señal de comunicación.

Suscripción y publicación al servicio de mensajería MQTT en el Gateway y conexión WiFi.

Figura 55

Diagramas de flujo de conexión WiFi y comunicación con protocolo MQTT

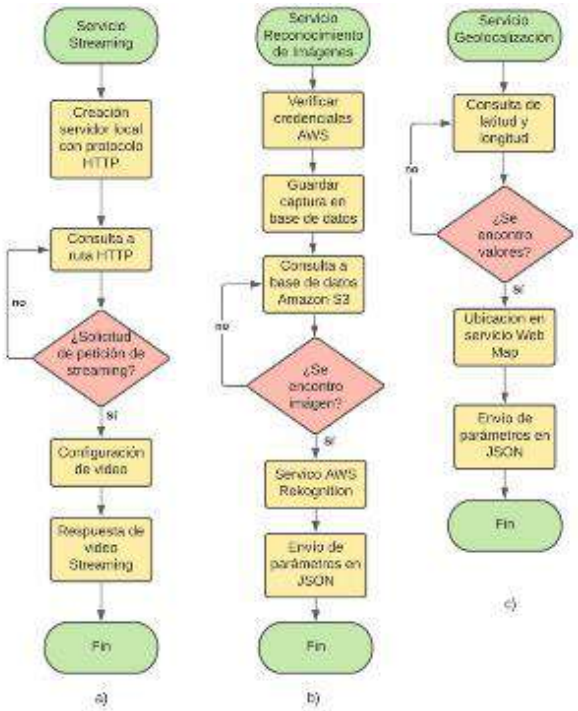
Nota: Se muestra la lógica de funcionamiento de la comunicación MQTT que se utilizará en la capa Fog Computing.

Primero debe conectarse el ESP32 Heltec Wireless Stick Lite con la red WiFi para luego realizar una conexión al servidor de mensajería el cual se detalla en la capa Cloud Computing, a pesar de ello lo que se realiza aquí prácticamente es declarar los identificadores para la conexión MQTT, luego suscribirnos al tópico del servidor para finalmente tener una secuencia en donde se envíe información a la capa Cloud Computing y luego se lea y se recibe información de la misma hasta que se detenga la aplicación, el diagrama de flujo donde se explica la lógica de programación que se realizará en el hardware se observa en la figura 55.

Módulo 3. Cloud Computing

Figura 56

Diagramas de flujo de los servicios streaming, reconocimiento de imágenes y geolocalización.



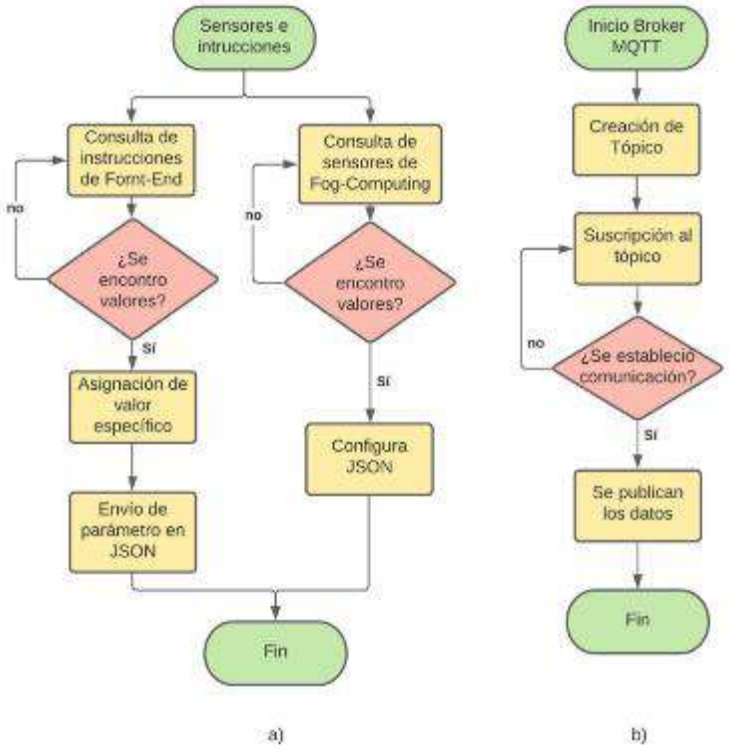
Nota: a) Lógica de funcionamiento del servicio de video streaming, b) Lógica de funcionamiento del servicio de reconocimiento de imágenes, c) Lógica de funcionamiento del servicio de geolocalización.

Servicio video streaming. Como ya se ha mencionado la cámara de video está en el robot, y se utiliza la rasberry pi para la creación de un servidor local utilizando protocolo HTTP y utilizando tecnología de acceso WiFi y puentes con ayuda de software independientes se puede llevar dicho servicio hacia la capa Cloud Computing, donde la lógica de programación para su uso se muestra en la figura 56.a.

Servicio de reconocimiento de imágenes y base de datos. Utilizando Amazon S3 para guardar las capturas del streaming y AWS Rekognition para el análisis y obtención de objetos reconocidos de cada imagen, el funcionamiento del servicio se lo observa en la figura 56.b.

Figura 57

Diagramas de flujo de los servicios de mensajería y el proceso para sensores e instrucciones



Nota: d) Lógica de funcionamiento de los sensores e instrucciones, e) Lógica de funcionamiento del servicio de mensajería.

Servicio de geolocalización. El servicio de geolocalización es de libre acceso y solo requiere de los datos tanto de latitud como de longitud para dar como respuesta la ubicación del objeto, su lógica de funcionamiento se observa en la figura 56.c.

Sensores e Instrucciones. En la figura 57.a se muestra cómo son tratados los datos provenientes de Fog-Computing y del Front-End.

Servicios de mensajería. Utilizando el bróker MQTT se da a conocer en la figura 57.b los pasos que sigue el servicio para empezar la transmisión de mensajes.

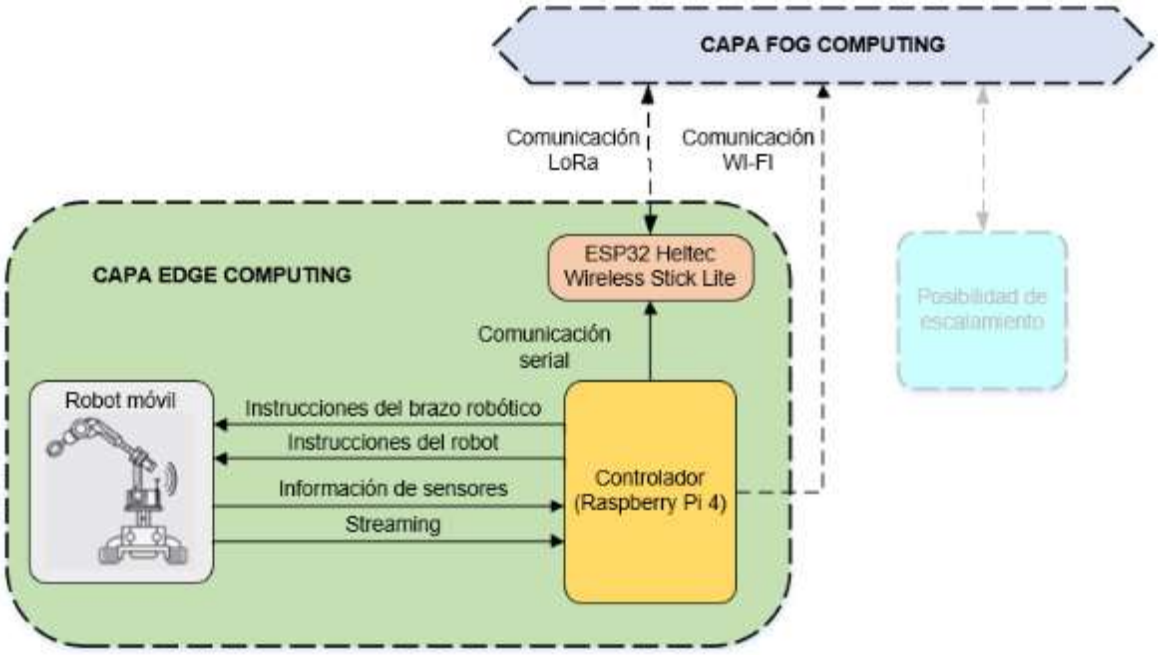
Etapa 6. Esquema definitivo

Módulo 1. Capa Edge Computing

Una vez detallados todos los requerimientos, características y estructura que tendrá esta capa, se muestran en la figura 58 el modelo de la arquitectura de esta capa.

Figura 58

Representación del funcionamiento de la capa Edge computing

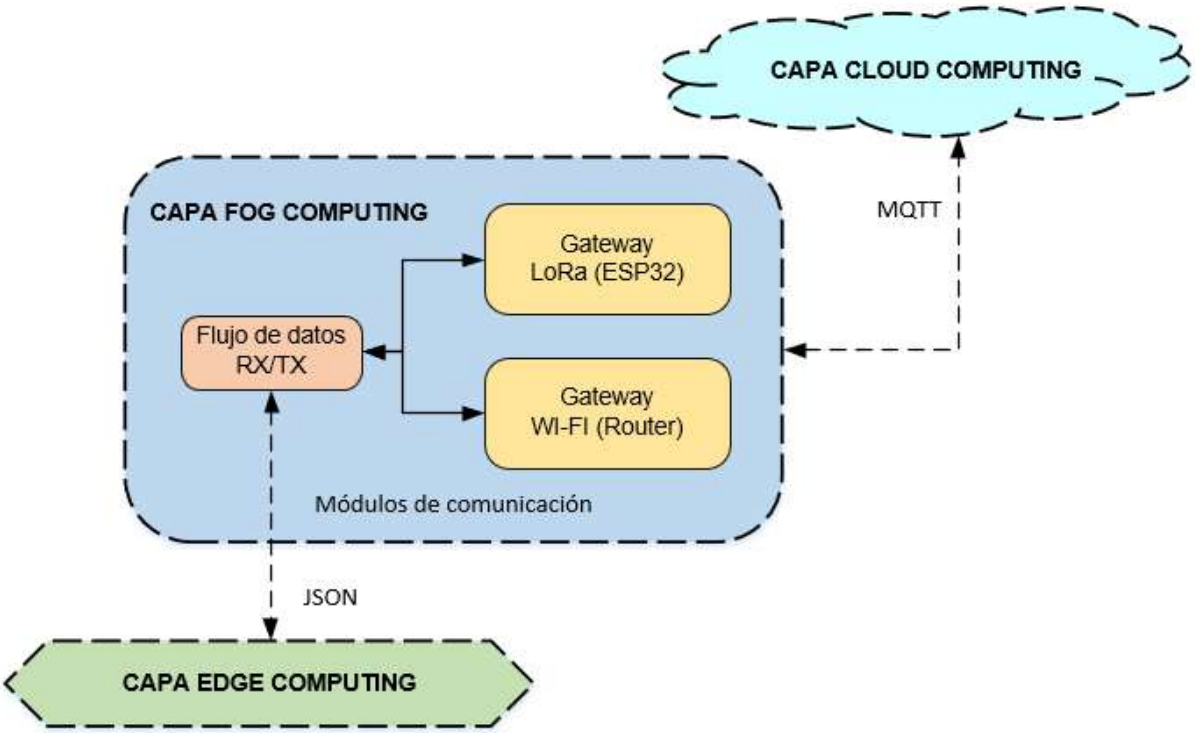


Nota: Se detalla la transferencia de datos, entre los módulos que componen esta capa.

Módulo 2. Capa Fog Computing

Figura 59

Representación del funcionamiento de la capa Fog computing



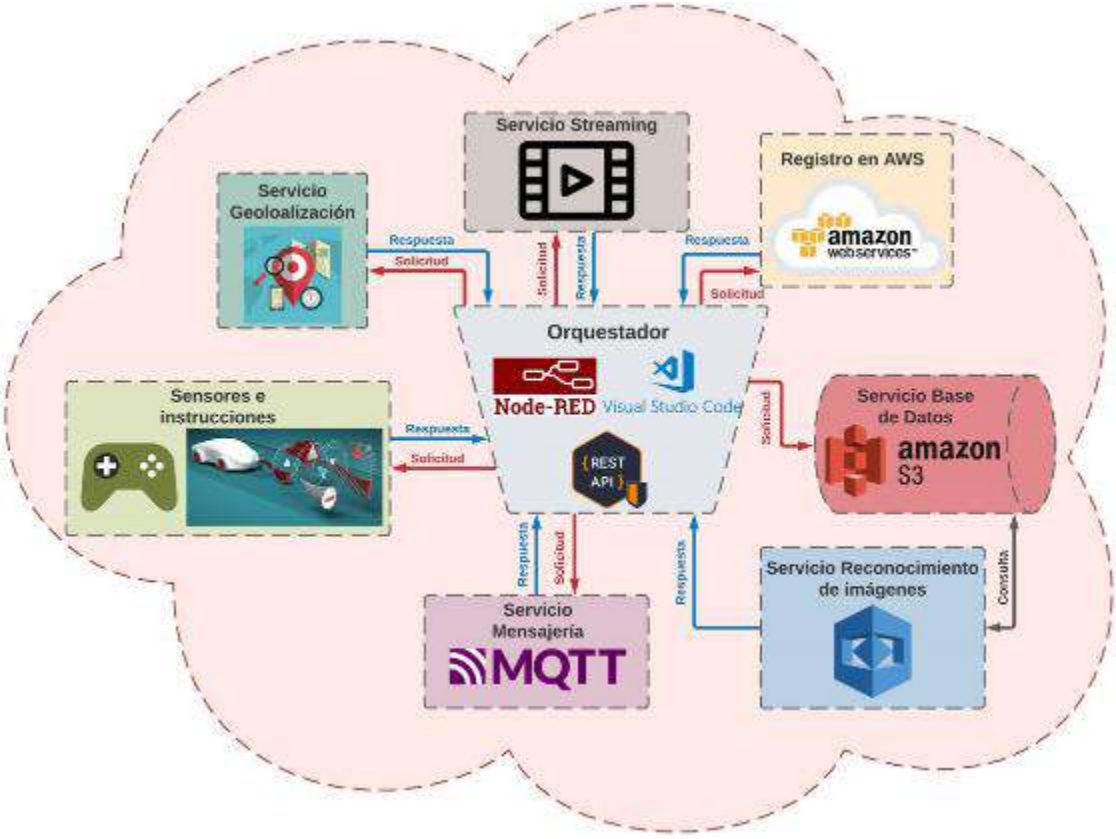
Nota: Se detalla la transferencia de datos, entre los módulos que componen esta capa.

Módulo 3. Cloud Computing

El orquestador en este caso permite que los servicios funcionen de forma paralela conforme a las peticiones del cliente REST aquí se incorporan todos los servicios con la finalidad de interpretar las respuestas de cada uno y llevarlas hacia el Front-End. El diagrama de la figura 60 corresponde a la capa Cloud Computing en donde se observa cómo se incorporó las herramientas de software de orquestación y los diferentes servicios utilizados junto con sus respectivos proveedores con la finalidad de tener una mejor comprensión de la arquitectura en esta capa.

Figura 60

Representación del funcionamiento de la capa Cloud computing



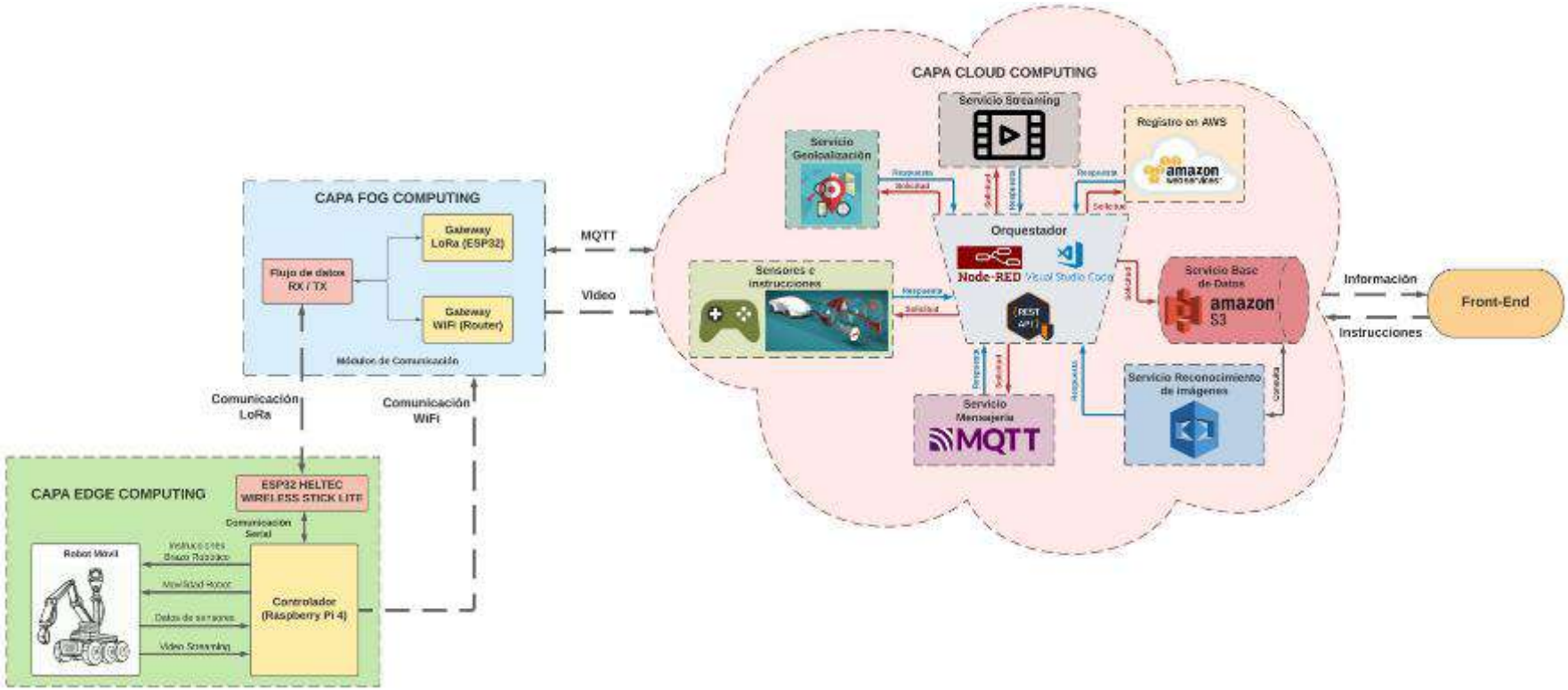
Nota: Se indica cómo se realizan las solicitudes a cada servicio de forma paralela.

Etapa 7. Integración de Arquitectura Completa

A continuación, en la figura 61 se da a conocer el diseño de arquitectura Cloud IoT comprendida por las 3 capas Edge Computing, Fog Computing y Cloud Computing a fin de tener una visualización completa de cómo se encuentra integrada luego de analizar a detalle lo que realiza y contiene cada una de las capas para un mejor entendimiento. La capa del Front-End representa la interfaz de visualización para el usuario y de igual manera se tiene un apartado para mostrar su diseño.

Figura 61

Representación del diseño de la arquitectura Cloud IoT



Nota: Se representa la comunicación entre las capas Edge Computing, Fog Computing y Cloud Computing.

Diseño de capa Front-End (HMI)

Etapa 1: Requerimientos

Los requerimientos para el diseño del HMI que se encuentra en la capa front – end son acorde a los campos de monitoreo y control con los cuales el usuario puede interactuar con el robot, el diseño es en base a lo propuesto en los objetivos del presente trabajo por lo que en las tablas 24 y 25 se detalla los requerimientos y las características a considerar dentro del HMI. El HMI se realiza en servidores de creación de dashboard por ejemplo el que se encuentra asociado con Node-RED.

Tabla 24

Requerimientos del usuario para el Front-End

Nº	Requerimiento
	HMI
1	Campo de textos
2	Campo de control
3	Campo de monitoreo
	Entorno programación
1	Solicitudes y respuestas en tiempo real.
2	Obtener información de video streaming, reconocimiento de imágenes, sensores y geolocalización.
3	Enviar las instrucciones para el control del robot.
4	Asignar valores específicos para cada parámetro de instrucción del robot.
5	Ser claro, conciso y estructurado.
6	Acoplarse al formato de envío y recepción de información.

Nota: En esta tabla se presentan los campos que constituyen al Front-End

Tabla 25

Características del Front-End

Nº	Características
	HMI
1	Letras legibles, establecer ubicación y matices de colores según la norma ANSI/ISA 101.
2	Botones para controlar el direccionamiento del robot y la movilidad del brazo robótico.
3	Transmisión de video streaming, visión de la captura para el reconocimiento de imágenes, detector de obstáculos y geolocalización del robot.
	Entorno programación
1	Se presentan las respuestas de los servicios en tiempo real presentando la latencia originada por el tiempo de procesamiento de información, además del tiempo de envío y transmisión de los datos al pasar por la capa Fog Computing.
2	El Front-End utiliza nodos específicos para que las respuestas de cada servicio sean representadas de forma individual pero visualizadas al mismo tiempo correspondiente al diseño del HMI.
3	Se utiliza un nodo para cada instrucción, las cuales se envían de forma paralela a guardarse en un parámetro, cada orden se ejecuta de forma instantánea una tras otra en el menor tiempo posible.
4	A cada instrucción se le asigna un valor específico para que pueda ser interpretado luego en la capa de Fog Computing.

Nº	Características
5	Utilizando los nodos de dashboard se puede organizar por campos, previamente descritos en el HMI, evitando la existencia de confusión en las solicitudes y respuestas tenidas en el Front-End.
6	El formato de envío de información es en JSON para que toda la arquitectura se maneje de igual manera.

Nota: En esta tabla se presentan las características correspondientes a cada requerimiento del Front-End.

Etapa 2. Funciones y Estructuras

Las funciones que constituyen el Front-End son en base a los requerimientos previos en base al usuario, pues es el medio donde se el usuario tiene la disponibilidad de interactuar con la máquina y no se realiza ningún procesamiento de datos solamente es para visualizar y enviar instrucciones por lo que si se lo interpretara en un diagrama las entradas se representan por los campos de texto y monitoreo mientras que las salidas están representadas por el campo de control como se visualiza en la figura 62.

Figura 62

Funciones del HMI

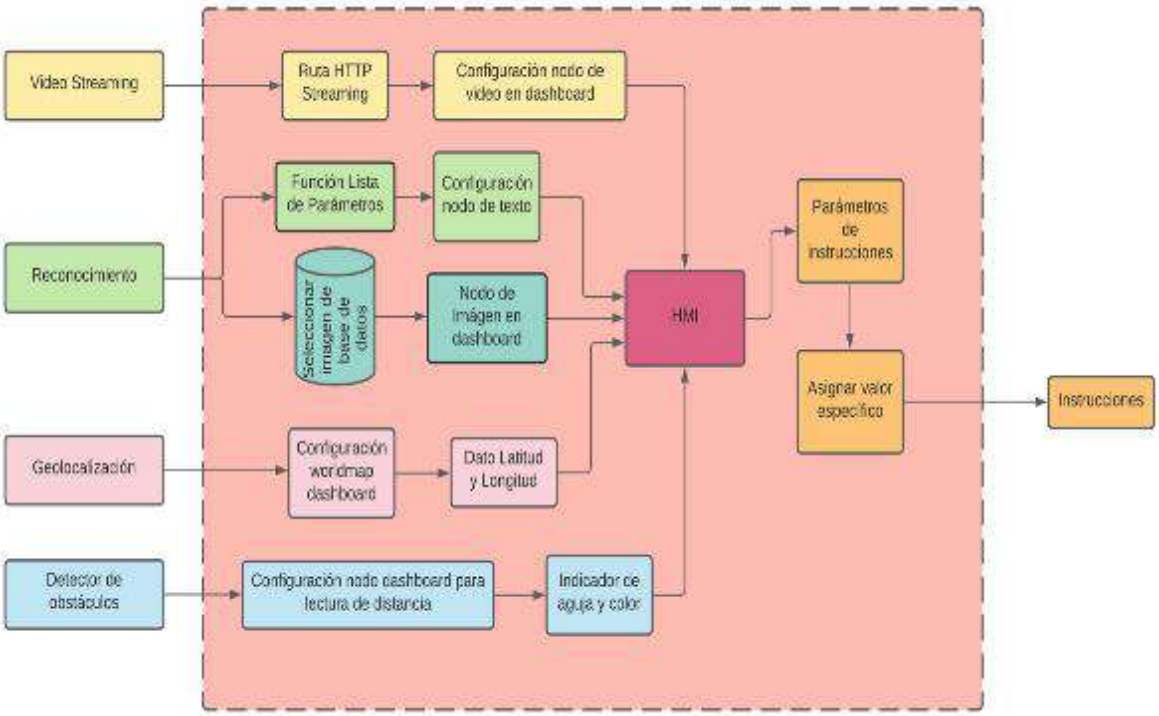
Nota: Representación de las funciones de un HMI a manera de entradas y salidas

Tanto el formato de los datos de entrada como de salida se encuentran en formato JSON a excepción del video streaming que se lo realiza mediante un llamado a la ruta HTTP del

servidor para visualizarse en el HMI, cada servicio utiliza un nodo para poder obtener una mejor organización en la herramienta utilizada, para conocer cómo son las funciones para la programación del dashboard a partir de uno de nodos se puede observar la figura 63.

Figura 63

Diagrama funcionamiento Front-End



Nota: Representación de las funciones de los nodos propio de un dashboard para poder realizar el HMI

Entradas.

- Video streaming: Se visualiza utilizando un nodo del dashboard llamando a la ruta HTTP del servidor streaming.
- Reconocimiento: Se muestra el listado de los objetos, personas y ambientes en una captura del streaming.
- Geolocalización: Se representa latitud y longitud para ubicar al robot en un servidor de mapa mundial y visualizarlos en el HMI.

- Detector de obstáculos: Con capacidad máxima de 3 metros representado con un identificador de aguja y con colores, cuando el objeto está a menos de 1 metro se pone en rojo, entre 1 y 2 metros en amarillo y mayor a 2 metros el color es verde.

Salidas.

- Instrucciones: Son los botones para controlar la dirección del robot móvil y la movilidad del brazo robótico.

Etapa 3: Principios de solución y variables

Tabla 26

Soluciones para el diseño del HMI

Sub-funciones	Soluciones
Equipo de visualización	PC o smartphone
Normas HMI	ANSI/ISA 101 - Determina colores Blanco/Cian y posición de los campos
Visualizar video streaming	Nodo con ruta HTTP
Comunicación	Datos formato JSON
Geolocalización	Datos de latitud y longitud
Detector de obstáculos	Dato de distancia del objeto
Reconocimiento de Imágenes	Datos de los parámetros de reconocimiento desde AWS Rekognition.
Tecnología de Acceso	WiFi

Nota: Esta tabla da a conocer las soluciones a las problemáticas presentadas en el diseño del HMI.

Se procede a determinar las formas en las que se pueden desarrollar el HMI de tal manera que pueda cumplir con lo especificado en los requerimientos y cumpla con las funciones que se le fueron asignadas para lo que tenemos la tabla 26 en donde se

especifica las alternativas de solución a todas las subfunciones que se pueden encontrar involucradas.

Etapa 4. Módulos

Módulo 1. Configuración Front-End

- Lógica de nodos para realizar el HMI utilizando un dashboard de una herramienta de interconexión de flujos.

Módulo 2. Campo de texto

- Reconocimiento de Imágenes con AWS Rekognition.

Módulo 3. Campo de control

- Botones direccionamiento de robot móvil.
- Botones de movilidad del brazo robótico.
- Botones extra para luces y posición Home del robot.

Módulo 4. Campo de Monitoreo

- Video Streaming.
- Detector de Obstáculos.
- Geolocalización.

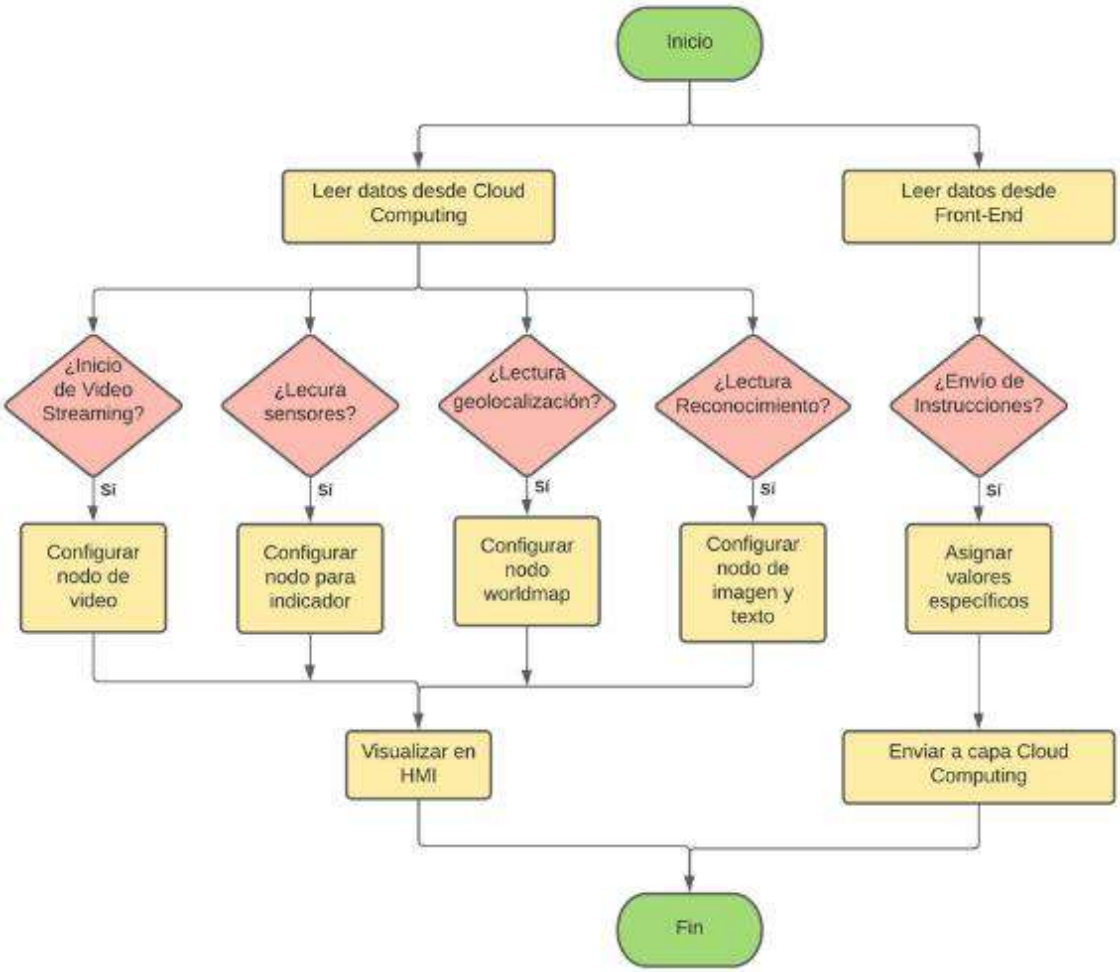
Etapa 5. Esquema de los módulos

Módulo 1. Configuración Front-End

El diagrama de flujo de la figura 64 representa el funcionamiento y la lógica para la programación de los nodos en donde se coordine tanto la entrada como salida de datos del Front-End.

Figura 64

Diseño de la lógica de programación en el Front-End



Nota: Representación del diagrama de flujo para comprender el funcionamiento del Front End.

Módulo 2. Campo de texto

Reconocimiento de Imágenes con AWS Rekognition. Se presenta el diseño del HMI para la sección de reconocimiento de imágenes utilizando los servicios de AWS Rekognition, en donde se pretende presentar al uruaio un campo donde se visualizase la captura de imagen tomada del video streaming seguido de un listado en un campo de texto sobre todos los objetos reconocidos en dicha captura.

Figura 65

Diseño del módulo 1 – Sección de reconocimiento de Imágenes con AWS Rekognition.

Nota: Representación de la sección de reconocimiento de imágenes en el HMI

El botón de captura permite mostrar una imagen extraída del video streaming para luego mostrar el listado de todos los objetos detectados con el porcentaje de confiabilidad de cada uno.

Módulo 3. Campo de control

Botones de direccionamiento del robot. Se presenta en la figura 66 la disposición de los botones y los nombres que llevarán cada uno en el HMI para el direccionamiento del robot.

Figura 66

Diseño del módulo 2 – Botones de direccionamiento del robot

Nota: Representación de la disposición de botones de direccionamiento.

Los botones hacen referencia a:

- Adelante. El robot avanza hacia adelante.
- Atrás. El robot retrocede.
- Izquierda. Las llantas del robot giran hacia la izquierda.
- Derecha. Las llantas del robot rotan hacia la dirección derecha.

Botones de movilidad del brazo robótico. Se presenta en la figura 67 la disposición de los botones y los nombres que llevarán cada uno en el HMI para la movilidad del brazo robótico.

Figura 67

Diseño del módulo 2 – Botones de movilidad del brazo robótico

Nota: Representación de la disposición de botones de movilidad.

Los botones hacen referencia a:

- Giro Izq. El eje principal del brazo del robot gira hacia la izquierda junto con la cámara.
- Giro Der. El eje principal del robot gira hacia la derecha junto con la cámara.
- Levantar brazo. El brazo se mueve hacia arriba sin la cámara.
- Bajar brazo. El brazo se mueve hacia abajo sin la cámara.
- Levantar mano. La mano se mueve hacia arriba con las pinzas.
- Bajar mano. La mano se mueve hacia abajo con las pinzas.
- Abrir. Abre las pinzas.

- Cerrar. Cierra las pinzas.

Botones extra. Se presenta en la figura 68 la disposición de los botones y los nombres que llevarán cada uno en el HMI para el control de las luces del robot y la posición Home del robot.

Figura 68

Diseño del módulo 2 – Botones extra del robot

Nota: Representación de la disposición de botones para encendido de luces y para colocar al robot en una posición inicial llamada Home.

Los botones hacen referencia a:

- Luz ON. Las luces frontales del robot se encienden.
- Luz OFF. Las luces frontales del robot se apagan.
- HOME. Coloca al robot en una posición inicial de funcionamiento.

Módulo 4. Campo de monitoreo

Video Streaming. Área en donde se coloca el video en tiempo real captado por la cámara del robot, está localizado entre los botones de control del robot.

Figura 69

Diseño del módulo 3 – Video Streaming

Nota: Representación del servidor de video streaming en el HMI

Detector de obstáculos. Indicador con una aguja y colores para la detección de objetos cercanos al robot.

Figura 70

Diseño del módulo 3 – Detector de obstáculos

Nota: Representación del detector de obstáculos en el HMI

Geolocalización. Mapa mundial donde se representa el punto exacto de ubicación de robot.

Figura 71

Diseño del módulo 3 – Geolocalización

Nota: Servidor de mapa mundial representado en el HMI.

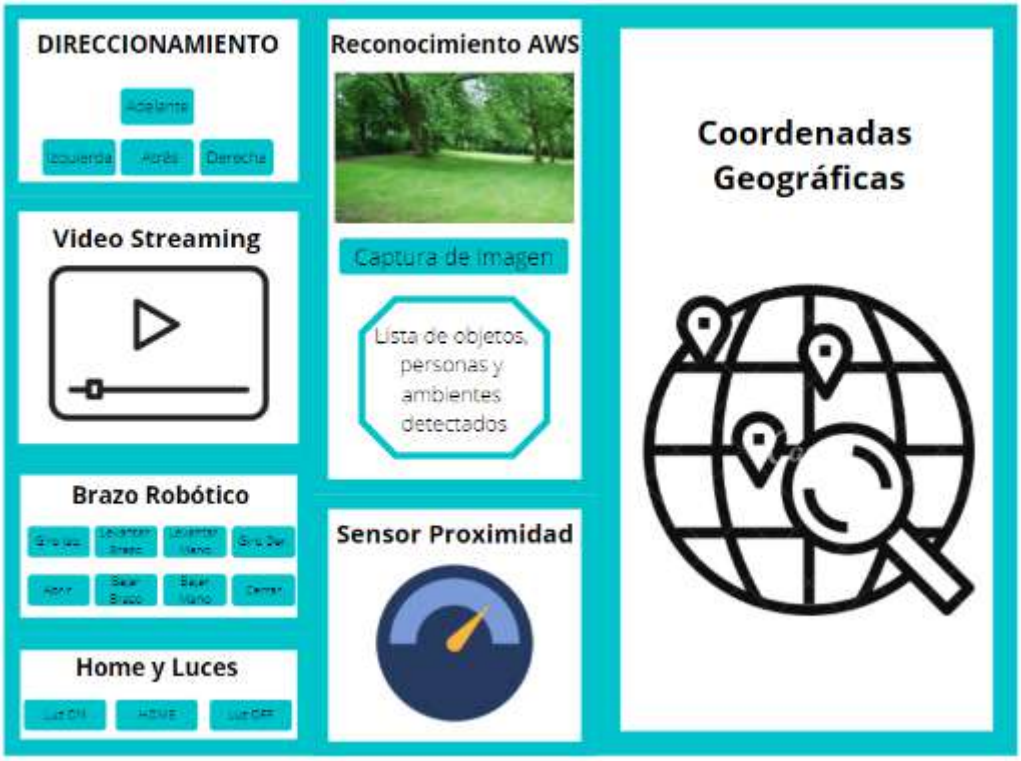
Etapa 6. Esquema definitivo

Se muestra a continuación en la figura 72 el HMI utilizando los colores blanco/cian como lo pide la norma ANSI/ISA 101 y la disposición conformado por cada uno de los módulos previamente descrito como es el campo de control, texto y monitoreo, la organización es de tal forma que el campo de control con los botones de instrucciones

para el robot se encuentre junto al video streaming para una mejor interpretación en la parte derecha, seguido de la parte de reconocimiento y los sensores del robot colocados en la parte izquierda como campos de texto y monitoreo.

Figura 72

Esquema definitivo del HMI



Nota: Diseño del HMI listo para ser implementado en un dashboard.

Capítulo 4: Implementación

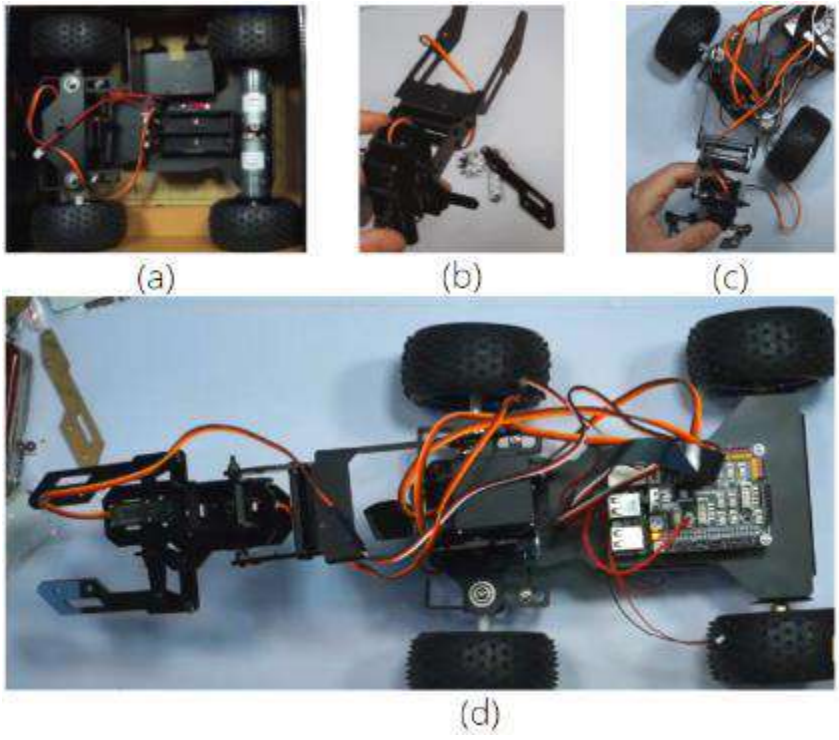
Ensamble del robot móvil

Una vez establecidos todos los parámetros de diseño del robot móvil, se procede con la implementación que se basará en principios de orden y calidad, para un correcto funcionamiento en la etapa de pruebas y posteriores. El ensamblaje del robot fue separado acorde a los módulos de diseño, donde cada cual tiene sus criterios de implementación específicos, para que posteriormente en conjunto pueda funcionar correctamente, satisfaciendo las necesidades dentro de la arquitectura.

Sistema mecánico

Figura 73

Ensamblaje de la estructura mecánica del robot móvil



Nota: a) Ensamblaje del Chasis. b) Ensamblaje del brazo robótico. c) Unión de chasis y brazo robótico. d) Ensamblaje mecánico completo.

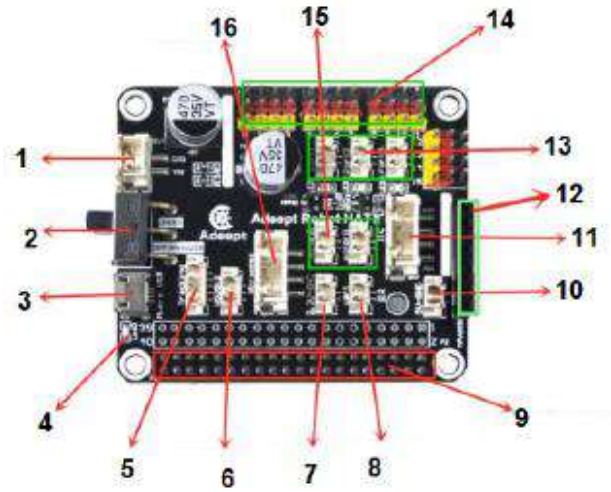
Los elementos del Kit PiCar Pro que componen el sistema mecánico y que fueron adaptados a esta aplicación en la etapa de diseño, se ensamblan para tener la estructura física del robot móvil. Cabe recalcar que algunos elementos electrónicos, como motores y servomotores también se han implementado en esta parte conjuntamente con el sistema mecánico, por la facilidad del montaje al estar las piezas mecánicas aun separadas como se puede observar en la figura 73.

Sistema electrónico

El sistema electrónico con el que cuenta el Kit PiCar Pro tiene una variedad de elementos, por lo que se seleccionó únicamente los elementos que se involucrarán para este aplicativo detallado en la etapa de diseño, cada uno de los elementos serán conectados en la tarjeta robotHAT, que es la parte central de las conexiones, que posteriormente ayudará a interconectar todos los elementos con el controlador. Esta tarjeta cuenta con varios pines de entrada y salida específicos para cada elemento como se muestra en la tabla 27 y figura 74, los mismos que deben ser conectados correctamente donde corresponda para evitar averías.

Figura 74

Elementos y pines para la conexión de los componentes detallados en la tabla 27.



Nota: Se detallan los componentes y pines de entradas y salida de la tarjeta robotHAT.

Tabla 27

Componentes de la tarjeta robotHat

Nº	Nombre	Descripción
1	Vin	Conexión para fuente de alimentación externa.
2	Switch	Encendido y apagado de la tarjeta robotHAT.
3	Micro USB	Conexión a PC o suministro de energía.
4	Alimentación LED	Indicador de encendido de la tarjeta.
5	Seguidor	Módulo de seguimiento.
6	WS2812	Pines de conexión del módulo WS2812.
7	3.3v-GND	Fuente para interfaz de 3.3V.
8	Uart	Puerto de comunicación Uart.
9	GPIO 40-PIN	Pines de conexión con la tarjeta Raspberry Pi 4.
10	5V-GND	Fuente para interfaz de 5V.
11	IIC	Módulo de conexión de la pantalla OLED.
12	MPU6050	Módulo de conexión del sensor MPU6050.
13	Puertos	Tres puertos para conexión de luces LED.
14	Servo port	Puertos para la conexión de los servomotores.
15	Motor	Conexión para motor 1 y motor 2.
16	Ultrasónico	Módulo de conexión de sensor ultrasónico

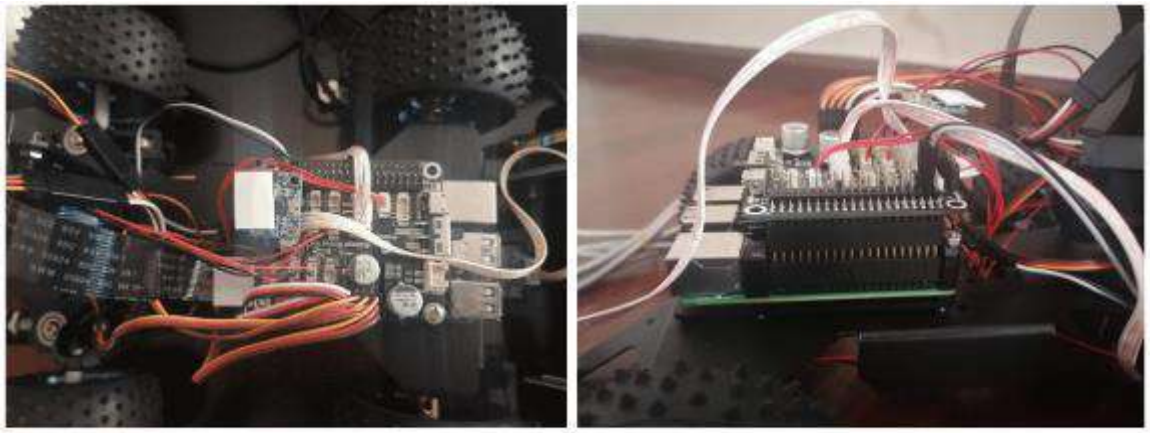
Nota: Información tomada de (HELTEC, 2021)

Definidos los elementos electrónicos a usarse e identificados los pines de conexión en la tarjeta para cada uno, se procede a la conexión física en el socket que corresponde a cada elemento, finalizado este proceso vamos a observar las conexiones como se muestra en la figura 75.a. Posteriormente vamos a conectar la tarjeta robotHAT con la Raspberry Pi, para ello se coloca esta última por debajo, haciendo calzar los 40

pines de conexión I/O entre las dos tarjetas como se muestra en la figura 75.b, terminando así las conexiones electrónicas de los elementos del Kit.

Figura 75

Conexiones de los elementos electrónicos.

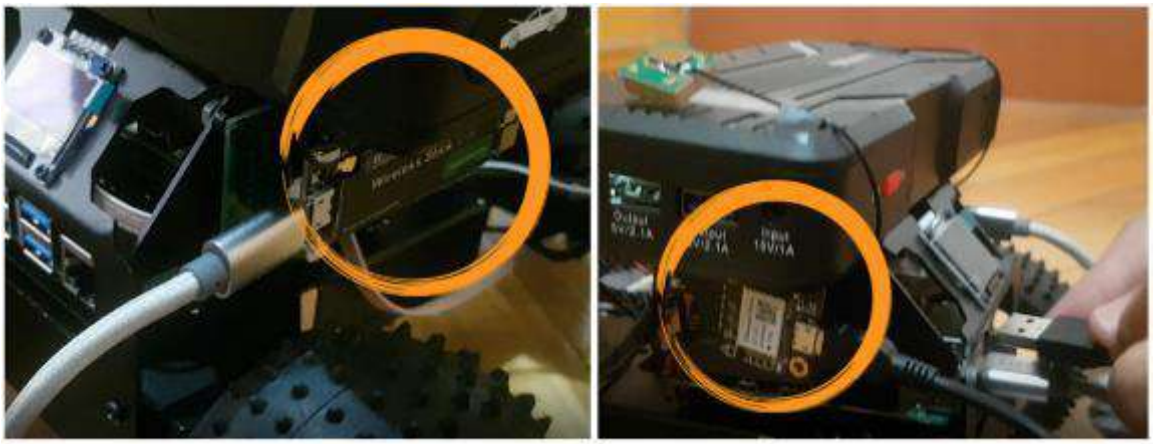


(a) (b)

Nota: a) Conexión de sensores y actuadores en la tarjeta robotHAT. b) Conexión de la tarjeta robotHAT con la Raspberry Pi.

Figura 76

Conexiones del sensor GPS y módulo de comunicación



(a) (b)

Nota: Adaptación mecánica y conexión eléctrica del a) módulo de comunicación y b) sensor GPS.

Algunos componentes no se encontraban dentro del Kit PiCar Por, por lo que fueron adaptados tanto a la parte eléctrica como mecánica como se observa en la figura 76, este es el caso del sensor GPS, y el módulo de comunicación de los cuales ya se detallaron las especificaciones en la etapa de diseño, una de ellas es que cuentan con la conexión micro USB, la misma que será usada para la comunicación y alimentación.

Fuente de energía

Determinados los valores de consumo de los elementos electrónicos que componen el robot en la etapa de diseño, se busca en el mercado una batería que se adapte a esas características y que a la vez su tamaño pueda adaptarse físicamente a la estructura del robot móvil como se muestra en la figura 77.

Tabla 28

Especificaciones técnicas de la batería Beatit B7 Pro

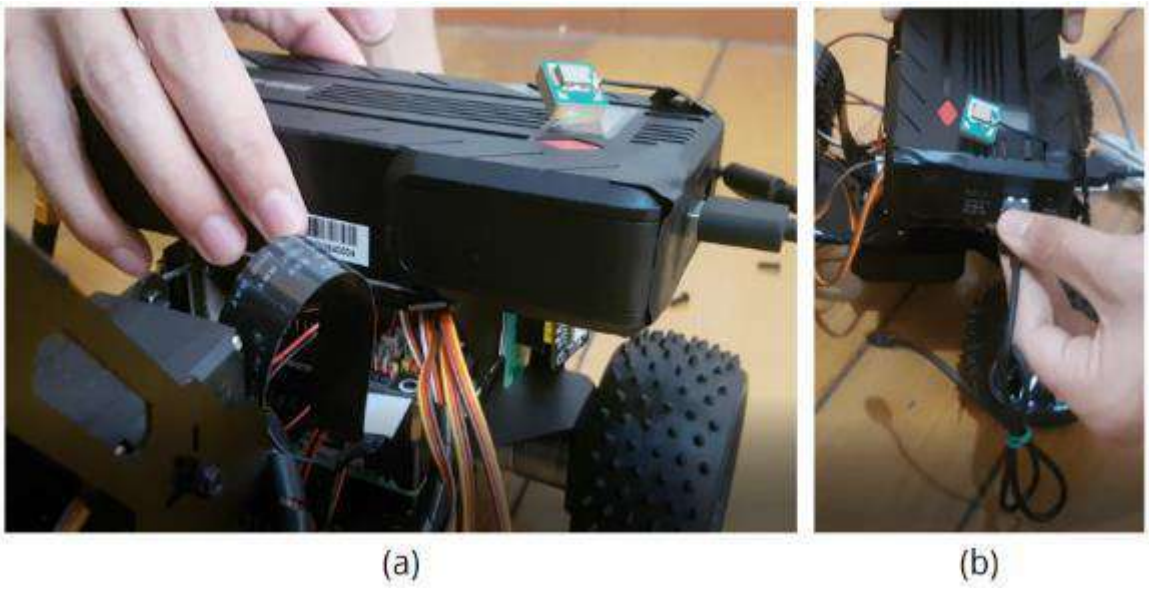
Especificaciones	**Detalle**
Tamaño	6.8 x 1.3 x 3.3 pulgadas
Amperaje	1200 A
Voltaje	12 V
Potencia	61.05 Vatios
Componentes	1 x arrancador B7, 1 x pinzas de batería inteligentes de 12 V, 1 x cargador de coche, 1 x adaptador de corriente, 1 manual de usuario, 1 maletín de transporte
Características especiales	Salida USB dual (5V / 2.1A), admite la última tecnología QDSP (Quick Discharge Start Power), banco de energía de 16500mAh
Calificación de seguridad	Certificación UL / CE / FCC / RoHs

Nota: Información tomada de (HELTEC, 2021)

La batería seleccionada fue la Beatit B7 Pro cuyas especificaciones técnicas se detallan en la tabla 28.

Figura 77

Montaje y conexión de batería



Nota: a) Ubicación y fijación de batería, b) conexión mediante cable USB.

El consumo en corriente de los elementos electrónicos analizado en la etapa de diseño es de 4.135 A, por lo que el tiempo estimado de duración de la batería se podría estimar con el dato del banco de energía con el que cuenta la batería, detallado en las especificaciones. Posteriormente este tiempo de duración será corroborado en la etapa de pruebas. $Duración\ bateria = \frac{16500mAh}{4135mA} = \mathbf{3.9h}$ (estimado).

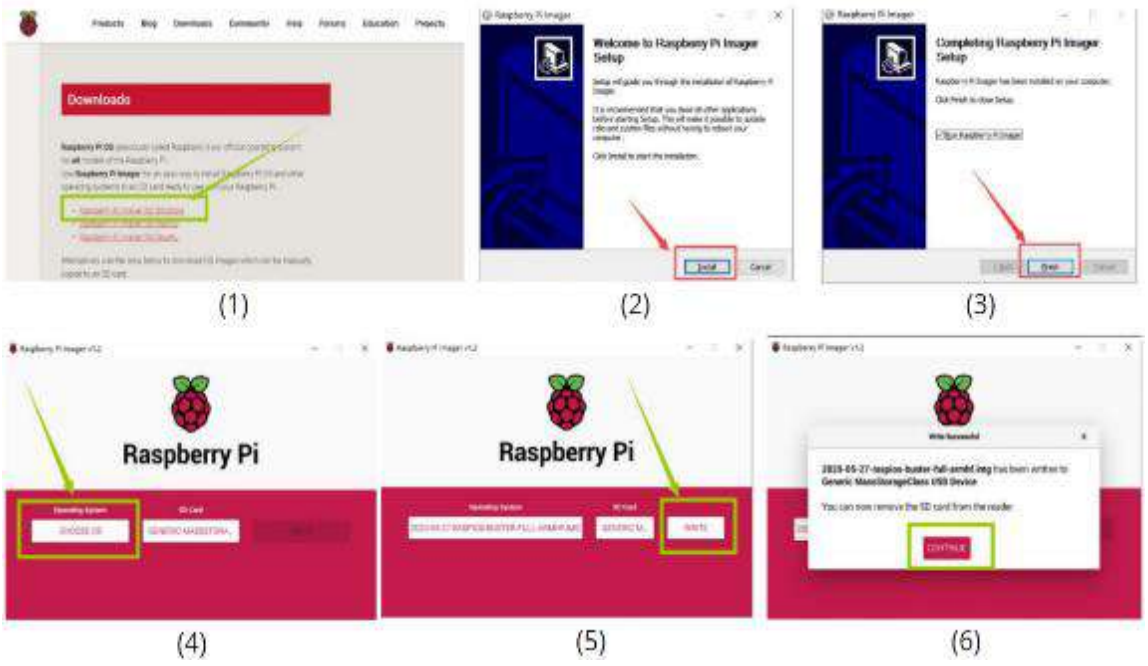
Controlador

El hardware del controlador será la tarjeta Raspberry Pi, de esta manera para establecer el software, primero se deben realizar las configuraciones detalladas a continuación que son necesarias para poder tener acceso a la tarjeta. Primero nos dirigimos a la página oficial de Raspberry, donde en la sección de descargas se buscará el sistema operativo Raspbian que tendrá adaptada una imagen para Windows, una vez

localizado procedemos a su descarga y posterior instalación. Después de la instalación se abrirá la interfaz donde primero seleccionaremos el sistema operativo, que después de seleccionarlo se descargará automáticamente en el computador. Terminada la descarga ubicamos el botón para seleccionar la tarjeta SD, para luego proceder a escribir el sistema en la misma, como se muestra en la figura 78.

Figura 78

Proceso de escritura del sistema operativo Raspbian en la micro SD

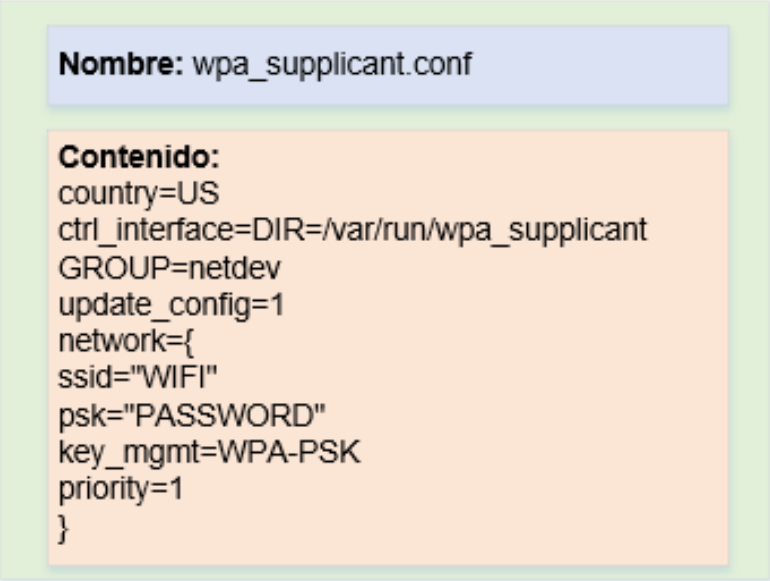


Nota: El proceso está establecido en orden del 1 al 6.

Para habilitar la comunicación, sin extraer aún la tarjeta SD del computador, dentro de la carpeta de instalación del sistema operativo, vamos a crear un archivo .txt llamado ssh que lo dejaremos en blanco, posteriormente creamos otro archivo llamado wpa_supplicant.conf donde se ingresaran los parámetros de conexión WIFI como usuario y contraseña como se muestra en la figura 79. Por último, extraemos la tarjeta del controlador y la colocamos en la llanura correspondiente de la Raspberry Pi.

Figura 79

Configuración de parámetros para conexión de Raspberry a WIFI



Nota: Los parámetros “WIFI” y “PASSWORD” son el usuario y contraseña de la red local a la cual va estar conectada la Raspberry mediante WIFI.

Con la tarjeta SD colocada, se encenderá la Raspberry, la cual automáticamente se conectará a la red configurada, de esta manera se le asignará una dirección IP de la red. Para encontrar esta dirección se necesita instalar una aplicación móvil que pueda hacer un barrido de las IP de la red con la que se identificara la que aparece con el nombre de Raspberry. Posteriormente vamos abrir un enlace remoto a través de VNC, pero antes debemos abrir la comunicación a través de programas que gestionen comunicación SSH, como Putty o MobaXterm. Estos programas necesitan tres parámetros para establecer la conexión, como la dirección IP identificada anteriormente, el “login name” y “password” que por defecto son pi y Raspberry respectivamente. Abierta la conexión se puede usar el programa VNC, para abrir la comunicación remota.

Teniendo el acceso al escritorio del sistema operativo de la Raspberry, usaremos el programa Python 3 para realizar la programación del software necesario que estará

contenido en el controlador. La lógica de control se estableció en la etapa de diseño, donde se pudo identificar los lazos de control, que para el caso de los actuadores manejaban lazos de control abiertos, mientras que para los sensores únicamente se hacía la lectura y organización de datos. Teniendo en cuenta el requerimiento que el software debe ser escalable, se realizan los programas por separado para cada actuador y bajo la lógica de funciones, con el fin de que cuando se requieran modificaciones o un crecimiento de la estructura únicamente se llamen a las funciones, sin alterar o dañar la programación. A continuación, se presentan las estructuras de los programas para cada uno de los actuadores.

Figura 80

Programa para el control de motores

Nombre: Move.py

Librerías:
```
import time
import RPi.GPIO as GPIO
```

Declaración GPIO:
```
Motor_A_EN    = 4
Motor_B_EN    = 17
Motor_A_Pin1  = 26
Motor_A_Pin2  = 21
Motor_B_Pin1  = 27
Motor_B_Pin2  = 18
```

Funciones:
```
def motorStop():#Parar motores
def setup():#Inicializar motores
def motor_left(status, direction, speed):#Control motor A
def motor_right(status, direction, speed):#Control motor B
def move(speed, direction, turn, radius=0.6):#Control motor A y B
def destroy():#Reinicio de programa
```

Main:
```
if __name__ == '__main__':
      try:
            speed_set = 60
            setup()
            move(speed_set, 'forward', 'no', 0.8)
            time.sleep(1.3)
            motorStop()
            destroy()
      except KeyboardInterrupt:
            destroy()
```

Nota: Programa para generar rotación positiva o negativa de los motores.

Figura 81

Programa para el movimiento de los servomotores

RPIservo.py

Nombre:

Librerías:

```
import time
import RPi.GPIO as GPIO
import sys
import Adafruit_PCA9685
import threading
import random
```

Declaración GPIO:

```
Adafruit_PCA9685.PCA9685()
```

Funciones:

```
def moveInit(self):#Inicializar servos
def initConfig(self, ID, initInput, moveTo):#Inicializar configuración
def moveServoInit(self, ID):#Servos en posición inicial
def posUpdate(self):#Cargar posiciones de servos
def speedUpdate(self, IDinput, speedInput):#Cargar velocidad de servos
def moveAuto(self):#Movimiento de servos
def moveCert(self):#Selección de servo a mover
def pwmGenOut(self, angleInput):#Configuración de pwm
def setAutoTime(self, autoSpeedSet):#Sincronización
def setDelay(self, delaySet):#Retardos
def autoSpeed(self, ID, angleInput):#Velocidad de retorno
def singleServo(self, ID, direcInput, speedSet):#Movimiento general
def moveAngle(self, ID, angleInput):#Movimiento servo dirección
def stopWiggle(self):#Parar servo
```

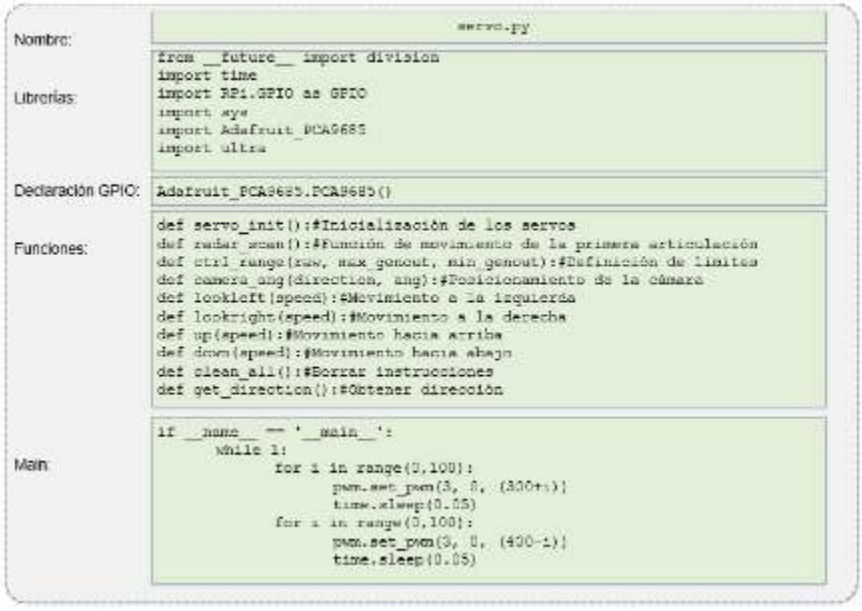
Main:

```
if __name__ == '__main__':
        sc = ServoCtrl()
        sc.start()
        while 1:
                sc.moveServoInit([0])
                time.sleep(delaytime)
        pass
```

Nota: Programa para mover los servomotores de dirección y del brazo robótico.

Figura 82

Programa que complementa las funciones del programa RPIservo



servo.py

Nombre:

Librerías:

```
from __future__ import division
import time
import RPi.GPIO as GPIO
import sys
import Adafruit_PCA9685
import ultra
```

Declaración GPIO:

```
Adafruit_PCA9685.PCA9685()
```

Funciones:

```
def servo_init():#Inicialización de los servos
def radar_scan():#Función de movimiento de la primera articulación
def ctrl_range(raw, max_genout, min_genout):#Definición de límites
def camera_ang(direction, ang):#Posicionamiento de la cámara
def lookleft(speed):#Movimiento a la izquierda
def lookright(speed):#Movimiento a la derecha
def up(speed):#Movimiento hacia arriba
def down(speed):#Movimiento hacia abajo
def clean_all():#Borrar instrucciones
def get_direction():#Obtener dirección
```

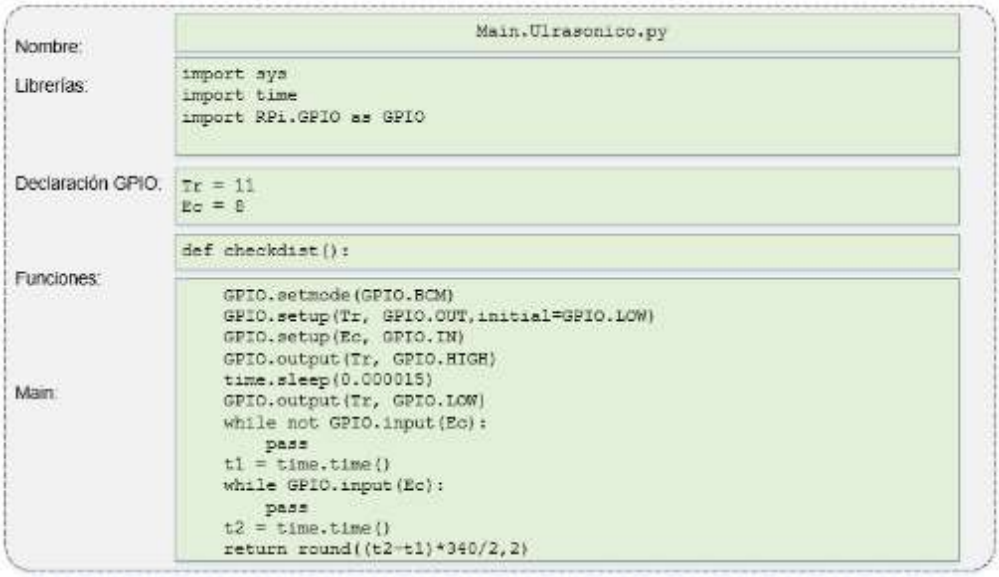
Main

```
if __name__ == '__main__':
        while 1:
                for i in range(0,100):
                        pwm.set_pwm(3, 0, (300+i))
                        time.sleep(0.05)
                for i in range(0,100):
                        pwm.set_pwm(3, 0, (400-i))
                        time.sleep(0.05)
```

Nota: Complemento del programa anterior, con el propósito de escalabilidad.

Figura 83

Programa para el sensor ultrasónico



Nota: Mediante este programa se hace la lectura de la distancia a través del sensor.

Figura 84

Programa para el sensor GPS

Nota: Mediante este programa se hace la lectura de ubicación del robot móvil.

Figura 85

Programa para control de luces auxiliares

Nombre: switch.py

Librerías:
```
import RPi.GPIO as GPIO
import time
```

Declaración GPIO:
```
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(5, GPIO.OUT)
GPIO.setup(6, GPIO.OUT)
GPIO.setup(13, GPIO.OUT)
```

Funciones:
```
def switchSetup():#Inicialización de GPIO
def switch(port, status):#Instrucciones ON OFF
def set_all_switch_off():#Apagar todos los puertos
```

Main:
```
if __name__ == "__main__":
    switchSetup()
    while 1:
        switch(1,1)
        switch(2,1)
        switch(3,1)
        print("Light on...")
        time.sleep(1)
        set_all_switch_off()
        print("Light off...")
        time.sleep(1)
```

Nota: Mediante este programa se encienden o apagan las luces de los puertos GPIO.

Figura 86

Programa para generación de acciones automáticas

Nombre: functions.py

Librerías:
```
import time
import RPi.GPIO as GPIO
import threading
from mpu6050 import mpu6050
import Adafruit_PCA9685
import os
import json
import ultra
import Kalman_filter
import move
import RPIservo
```

Declaración GPIO:
```
Adafruit_PCA9685.PCA9685()
```

Funciones:
```
def num_import_int(initial):#Llamado a las funciones requeridas
def pwmGenOut(angleInput):#Obtener los parametros de pwm para los servos
def setup():#Inicialización
def pause(self):#Pausar el movimiento
def automatic(self):#Movimiento automático
def automaticProcessing(self):#Movimiento automático
```

Main:
```
if __name__ == '__main__':
        try:
                fuc=Functions()

        except:
                move.move(0, 'no', 'no', 0)
```

Nota: Adaptación de funciones automáticas para funciones de robótica colaborativa.

Figura 87

Programa para monitoreo de la Raspberry

Nombre:	Info.py
Librerías:	`import psutil`
Declaración GPIO:	Ninguna

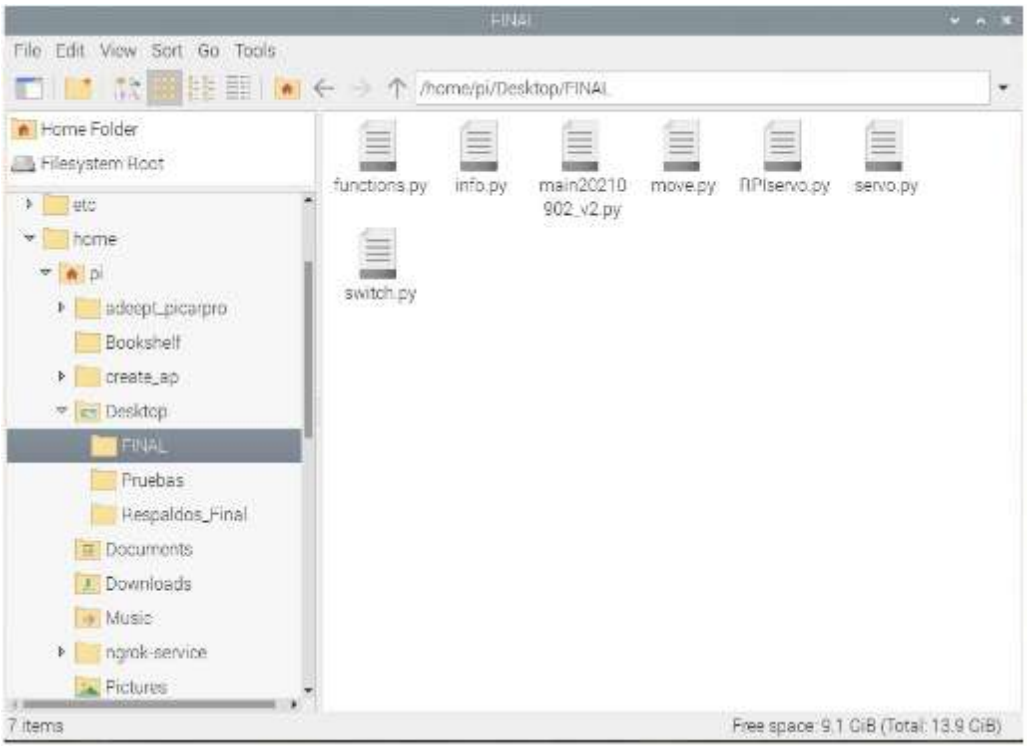
Funciones:	`def get_cpu_tempfunc():#Temperatura de la Raspberry`
	`def get_gpu_tempfunc():#Temperatura del chip`
	`def get_cpu_use():#Usabilidad de la Raspberry`
	`def get_ram_info():#Informacion de la RAM de la Raspberry`
	`def get_swap_info():#Información de los parámetros físico`

Main:	Se llama las funciones según las necesidades

Nota: Mediante este programa se monitorea la Raspberry en tiempo real.

Figura 88

Archivos de programas dentro de la Raspberry



Nota: Contenido de la carpeta donde se encuentran los programas necesarios para el controlador.

Terminados los programas, se los almacena en una sola carpeta dentro de la Raspberry como se observa en la figura 88 para de esta manera ir llamando las funciones según se requiera. Como se puede observar, toda la programación se la realizó de forma modular, para tener la posibilidad de escalamiento, de esta manera se ira llamando a las funciones según los requerimientos del robot, es por eso que se debe tener un programa principal denominado “Main” que será el encargado de enlazar todas las funciones. Este programa se analiza a detalle en la capa de Edge Computing que es a la que corresponde.

Implementación de la arquitectura Cloud IoT

Las tres capas serán implementadas de acuerdo a los criterios establecidos en la etapa de diseño, para de esta manera lograr tener el funcionamiento más óptimo de la estructura. Se establece una implementación ordenada y robusta tanto en el software y hardware implementado para no tener inconvenientes posteriores al funcionamiento del sistema.

Capa de Edge Computing

La capa de Edge Computing es la encargada del control del robot móvil, así como del envío de datos, donde los parámetros de recepción como de transmisión ya fueron establecidos en la etapa de diseño. Primeramente, vamos a establecer las funciones de recepción que son las encargadas de generar el desplazamiento y dirección del robot, así como también del brazo robótico. Las funciones ya fueron establecidas anteriormente en los programas del controlador, por lo que únicamente vamos a crear un programa “Main” el cual irá invocando las funciones según el carácter de instrucción recibido. El detalle de estos parámetros se da a conocer en la tabla 29.

Tabla 29

Invocación de funciones según el carácter recibido desde el programa "Main"

Caracter	Función invocada	Instrucción
1	`move.move(velocidad, 'forward', 'no', 0.8)`	Adelante
2	`move.move(velocidad, 'backward', 'no', 0.8)`	Atrás
3	`scGear.moveAngle(0,60)`	Izquierda
4	`scGear.moveAngle(0,-60)`	Derecha
5	`H_sc.singleServo(3, 1, 3)`	Levantar mano
6	`H_sc.singleServo(3, -1, 3)`	Bajar mano
7	`P_sc.singleServo(1, 1, 3)`	Giro izquierda
8	`P_sc.singleServo(1, -1, 3)`	Giro derecha
9	`G_sc.singleServo(4, -1, 3)`	Cerrar Gripper
0	`G_sc.singleServo(4, 1, 3)`	Abrir Gripper
B	`T_sc.singleServo(2, 1, 3)`	Levanta brazo
C	`T_sc.singleServo(2, -1, 3)`	Baja brazo
E	`switch.switch(2,1),switch.switch(3,1)`	Luz ON
F	`switch.switch(2,1),switch.switch(3,1)`	Luz OFF

H	`P_sc.moveServoInit([1])`
	`T_sc.moveServoInit([2])`
	`H_sc.moveServoInit([3])`
	`G_sc.moveServoInit([4])`
		Posición
		Home

Nota: Los atributos scGear, H_sc, P_sc, G_sc, T_sc invocan la función RPIservo.ServoCtrl().

Los datos de transmisión primeramente son almacenados en la Raspberry después de ser recibidos desde los sensores, estos son leídos mediante funciones de lectura y almacenados en variables, para posteriormente empaquetarlos convertirlos en

archivo JSON y enviarlos a través del puerto de comunicación serial abierto, estas funciones e instrucciones se detalla en la tabla 30.

Tabla 30

Invocación de funciones y variables para la transmisión de datos

Función invocada	Instrucción
`str(checkdist())`	Lectura del dato del sensor ultrasónico.
`gps_read()`	Lectura del dato del sensor GPS.
`distancia`	Variable que almacena el dato del sensor ultrasonico
`(lat,lon)`	Variable que almacena el dato del sensor GPS.
`datotx={"D":distancia,"A":lat,"L":lon,}`	Empaquetamiento de datos.
`jsonString=json.dumps(datotx)+"]"`	Dato convertido a JSON
`esp32 = serial.Serial ("/dev/ttyUSB0",9600,timeout=0.2)`	Comunicación serial con la ESP32
`esp32.write(jsonString.encode('ascii'))`	Envío del dato tipo JSON

Nota: Los atributos scGear, H_sc, P_sc, G_sc, T_sc invocan la función RPIservo.ServoCtrl().

El programa “Main” es el encargado de orquestar las funciones antes detalladas, donde la transmisión de datos de los sensores es constante, y la recepción se realiza cuando el puerto de Rx detecta que algún dato ha sido enviado desde las capas superiores, lo lee y almacena para posteriormente ejecutar la instrucción correspondiente.

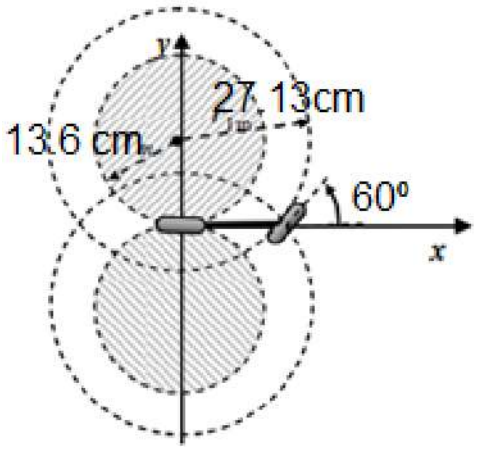
Uno de los parámetros que entró en análisis en esta parte de la implementación fue el ángulo de giro de las ruedas de dirección, ya que el robot debe estar sujeto a cambios de dirección repentinos por la función de exploración que debe ejecutar, es por eso que se analizaron varios ángulos de giro, donde por circunstancias mecánicas se estableció 60º como máximo ángulo de giro. A continuación, se analizan los valores de los radios de giro que se obtienen al emplear este ángulo, ya que este dato será útil para tener en cuenta en la manipulación al momento de esquivar obstáculos.

$$r_m = \frac{L}{\tan(\theta_m)}, \quad r_{1m} = \frac{L}{\sin(\theta_m)}.$$

$$r_m = \frac{23.5cm}{\tan(60°)} = 13.6\ cm, \quad r_{1m} = \frac{23.5cm}{\sin(60°)} = 27.13\ cm$$

Figura 89

Valores de los radios de giro generados con 60º de dirección



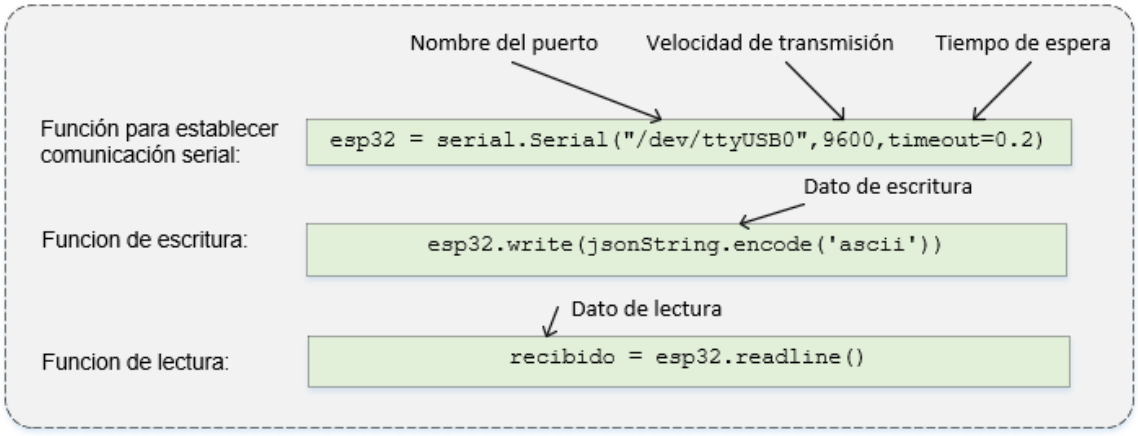
Nota: Figura tomada y adaptada de (Teodovich Sosa & Carelli, 2008).

Capa de Fog Computing

En esta capa primero se establece la comunicación entre la Raspberry que es el controlador de la capa anterior, con el módulo de comunicación que en este caso es la ESP32. Desde el lado la Raspberry, se establecen los parámetros de comunicación en software como se detalla en la figura 90.

Figura 90

Parámetros de comunicación serial establecidos en software en la Raspberry Pi

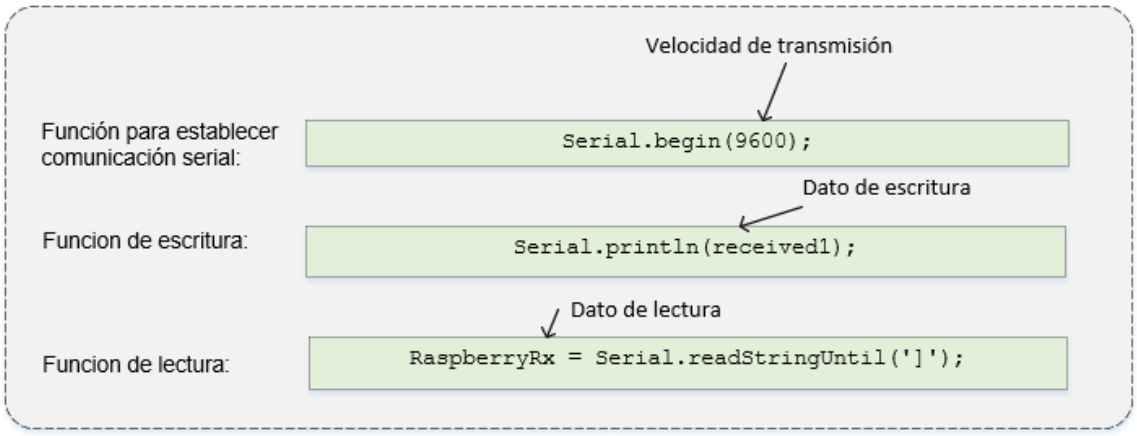


Nota: Las funciones están dentro el programa “Main” de la Raspberry Pi

En el módulo de comunicación ubicado en el robot que es la tarjeta ESP32, también se establecen los parámetros de comunicación como se muestra en la figura 91 para que puedan intercambiar datos entre las dos tarjetas. Para la programación del software en la ESP32 se utiliza el IDE de Arduino.

Figura 91

Parámetros de comunicación serial establecidos en software en la ESP32

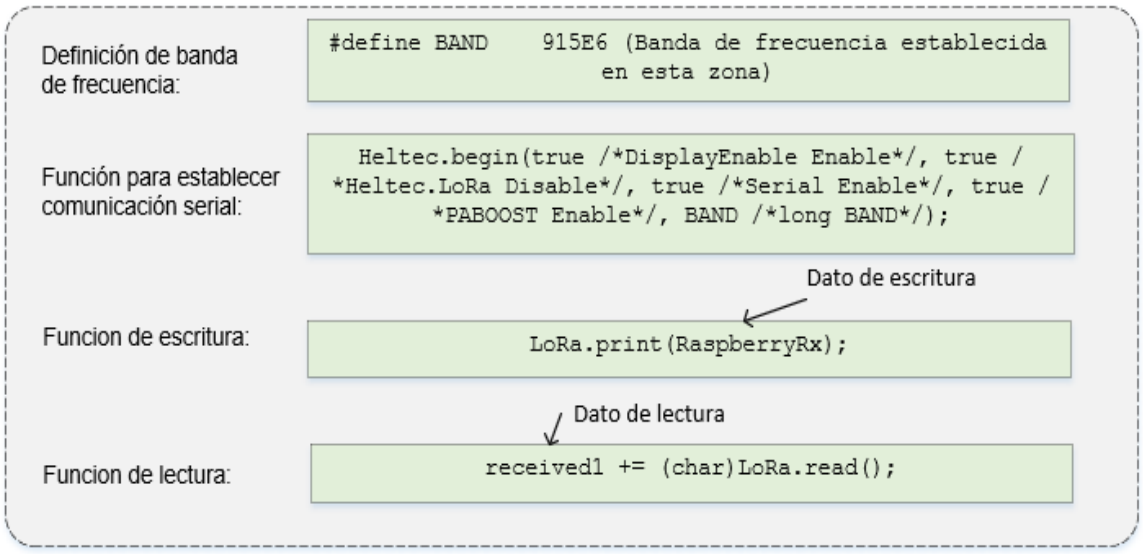


Nota: Las funciones están dentro el programa “Cliente” de la ESP32

Establecida la comunicación bidireccional entre la Raspberry Pi y la ESP32, se procede a implementar la comunicación LoRa entre las dos tarjetas ESP32, las cuales brindarán las prestaciones de comunicación inalámbrica a largas distancias. Primero se establecen las funciones necesarias en software para la comunicación LoRa dentro del programa de la ESP32 ubicada en el robot móvil como se detalla en la figura 92, que funcionará como “Cliente”.

Figura 92

Parámetros de comunicación Lora establecidos en software en la ESP32 “Cliente”



Nota: Las funciones están dentro el programa “Cliente” de la ESP32

De igual forma se implementan las funciones de comunicación necesarias del lado de la tarjeta ESP32 que estará funcionando como Gateway, de esta manera se podrá establecer la comunicación inalámbrica a largas distancias. A la vez este módulo estará conectado al internet a través de WI-FI mediante la conexión a una red local. Teniendo acceso a la red se procede a establecer la comunicación con la capa de Cloud a través de MQTT, todas estas funciones de conexión y configuración de comunicación se muestran en la figura 93.

Figura 93

Funciones de conexión para establecer comunicación entre los componentes de la capa

Nota: Las funciones están dentro del programa “Gateway” de la ESP32 a distancia.

Realizadas todas las conexiones necesarias para establecer la comunicación entre los módulos que componen esta capa, queda activada la transferencia de datos de forma bidireccional entre las capas de Edge y Cloud. Cabe recalcar que la conexión de la Raspberry con el Gateway WIFI para el envío de datos de video Streaming ya se realizó anticipadamente en la etapa de implementación del controlador por la necesidad de conectar y acceder a la Raspberry Pi mediante WIFI para implementar el software, de esta manera toda la arquitectura de esta capa ha sido implementada acorde a la arquitectura detallada en la etapa de diseño.

Capa de Cloud Computing

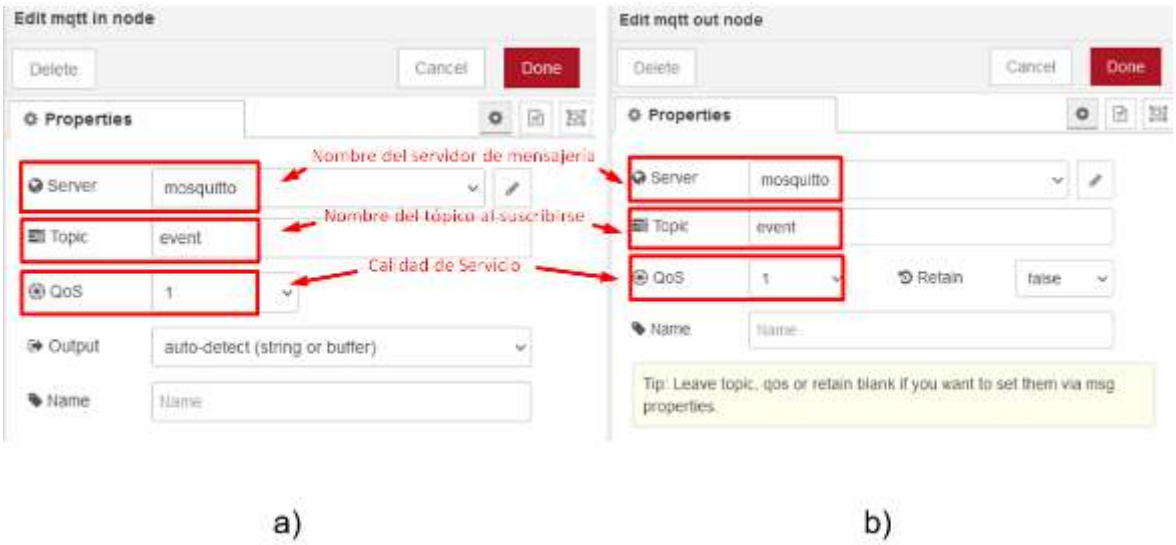
Las herramientas de software utilizadas para realizar la orquestación de los microservicios fueron dos, principalmente donde se encuentra gran cantidad de lo

desarrollado en el presente trabajo fue en Node-RED, pero también se utilizó Visual Studio Code para realizar una parte de la orquestación en NodeJS. A continuación, se muestra por cada servicio como se realizó su implementación y uso en la capa Cloud Computing.

Servicios de mensajería. El servidor de mensajería utilizado para envío y recepción de datos entre software / hardware y software / software fue MQTT Mosquito, el cual se puede instalar en cualquier computador para ser utilizado como un servidor local, la manera de funcionamiento de este servicio es Publisher / Subscriber, en donde se debe crear un tópico del lugar en donde se planea utilizar, además de conocer el host que es la IP del computador donde se encuentre levantado el servicio y el puerto de comunicación que usualmente es el 1883. Para evitar confusiones el tópico utilizado para enviar información de sensores e instrucciones fue 'event' y para la parte del reconocimiento de imágenes fue 'Reconocimiento'.

Figura 94

Ruta del servicio de video streaming



Nota: Se muestra como acceder al servicio de video streaming creado en una red local.

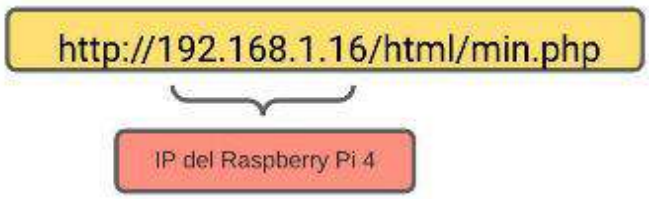
La ventaja de utilizar Node-RED para este servicio es la disponibilidad de nodos para MQTT donde se puede configurar un nodo de entrada para lectura de información que se lo ve en la figura 94 así como un nodo de salida para el envío de información.

Como se puede observar ambos nodos permiten configurar el nombre del servidor que en este caso se colocó Mosquitto para una mejor identificación, luego tenemos el apartado del tópico, el nodo de entrada permite configurar el formato de los datos de entrada a diferencia del nodo de salida que no lo hace, y ambos permiten configurar la calidad de servicio que en este caso es de nivel 1. Si se desea conocer más información sobre la lógica de este servicio el diagrama de flujo se encuentra expuesto en el apartado de diseño. Posteriormente cuando se utilice este servicio se mostrará el diagrama de nodos en Node-RED.

Servicio video streaming. El servicio de video streaming fue creado desde la Rasberry Pi 4, es decir un servidor local para que luego con el uso de puentes de la ruta HTTP para que pueda utilizarse en la capa cloud computing.

Figura 95

Ruta del servicio de video streaming



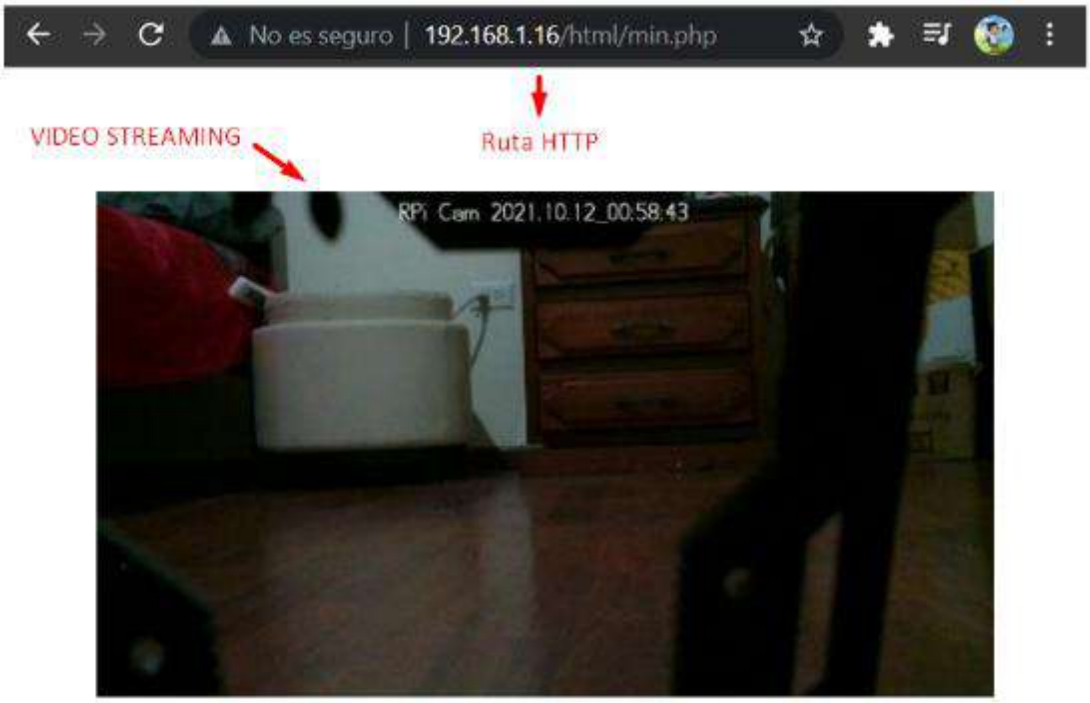
Nota: Se muestra como acceder al servicio de video streaming creado en una red local.

Gracias a la plataforma GtHub podemos encontrar información de varios investigadores para poder servir de guía en la realización de los proyectos, como fue en nuestro caso gracias a (T3ALLIANCE, 2019) se nos proporcionó un software independiente para levantar nuestro servidor de video streaming desde un servidor local donde nosotros creamos nuestra ruta HTTP, pero utilizando la IP de nuestra Rasberry Pi

4 desde la red en la que nos encontremos conectados como se ve en la figura 95 para acceder a una página web donde se transmite el video streaming teniendo así la ruta como en la figura 96.

Figura 96

Acceso a página web del video streaming



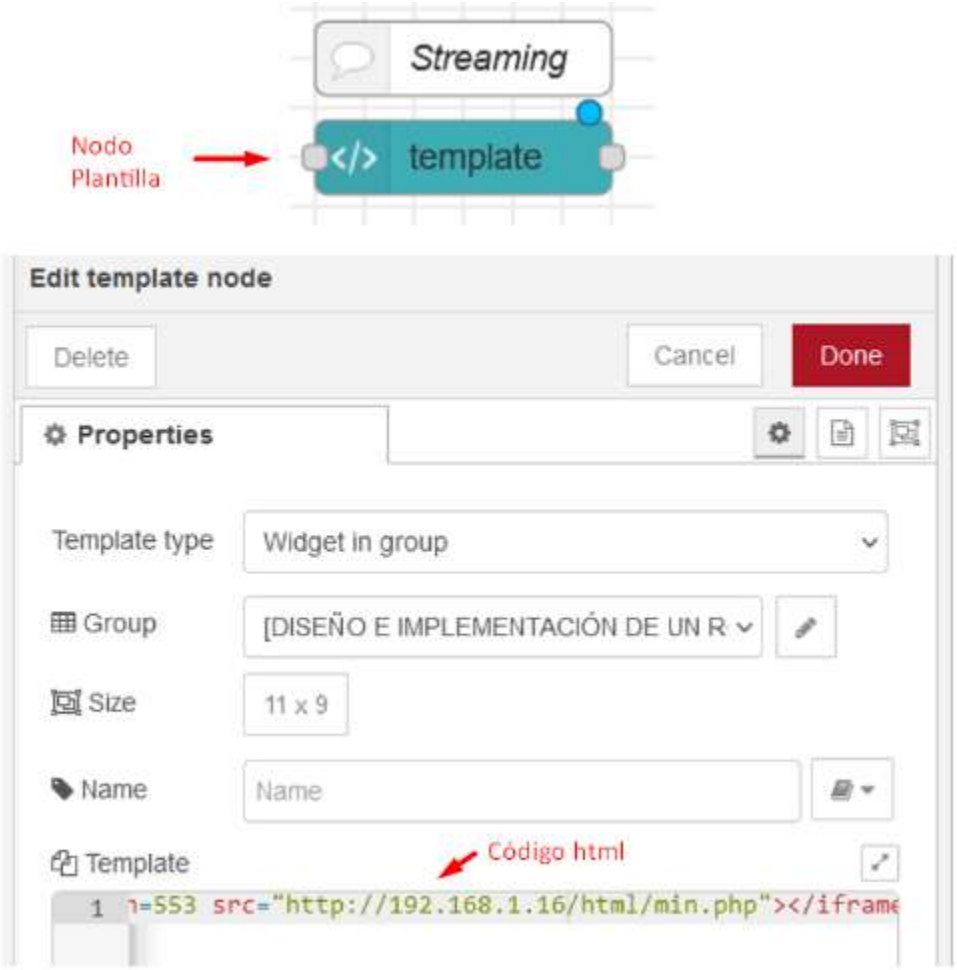
Nota: Así es como se representa el servicio de video streaming en una página web.

Luego en el software Node-RED colocamos un nodo de función para poder llamar a la ruta HTTP del servicio de video streaming, aunque también se lo puede realizar directamente en un nodo de plantilla para que se muestre la transmisión en el dashboard o la capa del Front-End, la configuración se la realiza en código html como se indica en la figura 97 donde se configura tanto las dimensiones del video, así como la ruta.

En el apartado del Front-End se explica como configurar este nodo para mostrarlo en el HMI ya que se puede utilizar un mismo nodo llamado template o plantilla con las mismas configuraciones para visualizar el video.

Figura 97

Configuración servicio video streaming en Node-RED

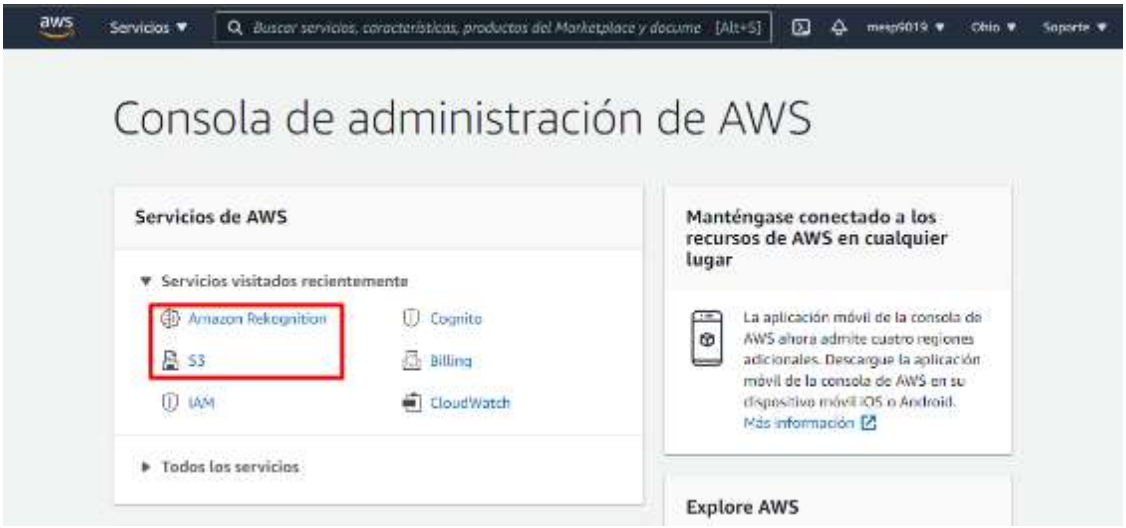


Nota: El código html a configurar es el siguiente: *<iframe scrolling=no marginwidth=0 marginheight=0 frameborder=0 height=439 width=553 src="http://192.168.1.16/html/min.php"></iframe>*

Servicio de reconocimiento de imágenes y base de datos. Para configurar este servicio se requirió de la herramienta Node-RED y de NodeJS, en primer lugar, se debe crear una cuenta en Amazon Web Services AWS donde se establezca que servicios se van a utilizar en la consola de administrador, para nuestro caso es la base de datos Amazon S3 y AWS Rekognition como se observa en la figura 98.

Figura 98

Consola de administrador de AWS al crear la cuenta en Amazon

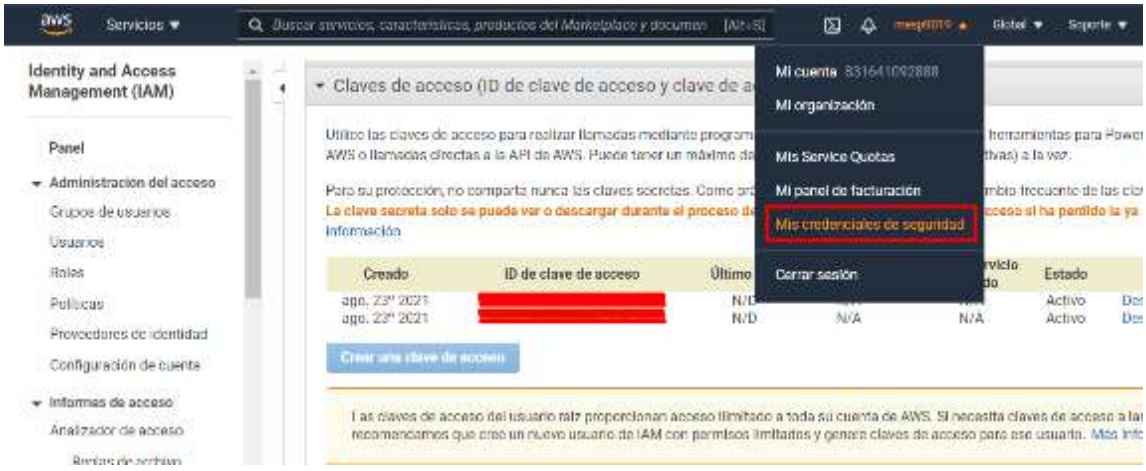


Nota: Se muestra la consola de administrador donde se tiene acceso a todos los servicios de Amazon.

Al momento de utilizar AWS debemos generar unas credenciales de seguridad utilizadas como identificación para utilizar servicios de Amazon.

Figura 99

Credenciales de seguridad en Amazon Web Services



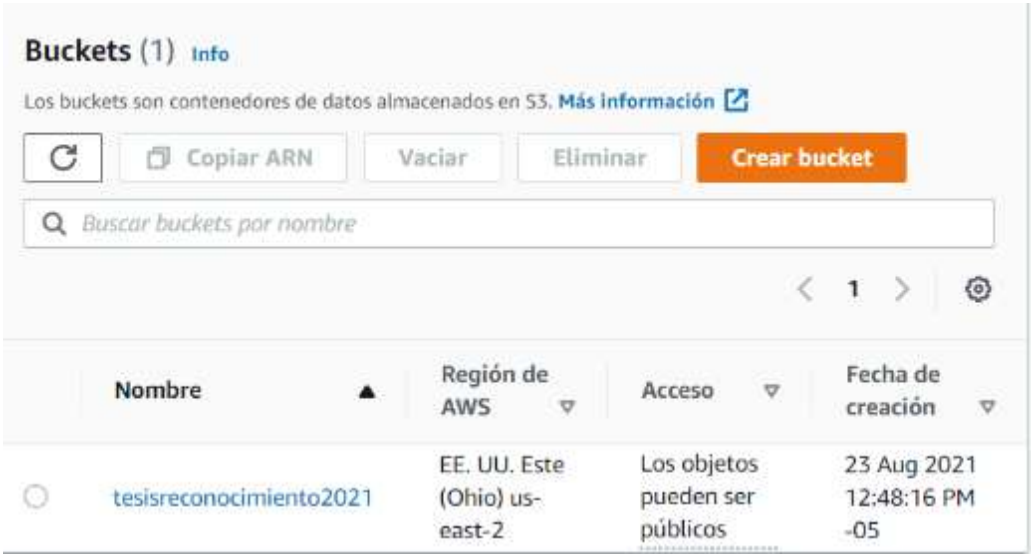
Nota: Creación y consulta de credenciales de seguridad para utilizar AWS.

AWS al momento de crear un rol nos da como identificador un accessKeyId y un secretAccessKey los cuales son visibles por un corto periodo de tiempo y el usuario debe copiarlos y guardarlos en un lugar seguro, además otro parámetro de identificación a utilizar es la región que configura al crear nuestra cuenta.

Posterior se crea la base de datos en el servicio de Amazon S3, accedemos y nos dirigimos a crear buckets en donde debemos colocar un nombre de la base de datos, la región y permitir que el servicio sea de libre acceso para el caso de este trabajo, algo a tener en cuenta es que la versión gratuita permite almacenar hasta 5 Gb de información por mes, obteniendo así la base de datos de la figura 100.

Figura 100

Credenciales de una base de datos en Amazon S3

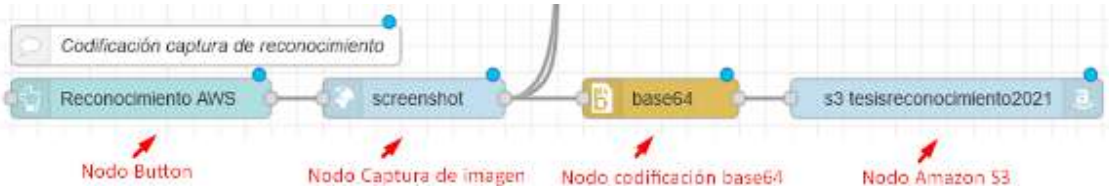


Nota: Aparecen las fechas de creación de las bases de datos.

En el caso de Amazon Rekognition no se debe crear nada pues con las credenciales ya se tiene acceso al servicio. Ahora en Node-RED utilizamos un nodo para realizar la captura de imagen del video streaming, luego esta imagen se codifica en base64 para posteriormente ser guardada en Amazon S3 utilizando otro nodo, todo este evento se activa mediante un botón como se observa en la figura 101.

Figura 101

Captura de imagen y guardado en Amazon S3



Nota: Nodos utilizados en Node-RED para capturar la imagen y guardarla.

Figura 102

Código en NodeJS para el reconocimiento de imágenes

Introducir credenciales AWS

```
//Detalles de acceso a AWS
function orqInd() {
AWS.config.update({
accessKeyId: '**********',
secretAccessKey: '**********',
region: 'us-east-2'});
```

Acceso a imágen en Amazon S3

```
//Entrada de parámetros
var params = {
Image: {
S3Object: {
Bucket: "tesisreconocimiento2021",
Name: "reconocimiento.jpg"
}
}};
```

Acceso servicio AWS Rekognition

```
//Llamada a la clase AWS Rekognition
const rekognition = new AWS.Rekognition();
var detectedTXT;
//Detección de Labels
rekognition.detectLabels(params, function(err, data) {
if (err) console.log(err, err.stack); // Ocurrió un error
else//console.log(data)
var response = data.Labels[1].Name
var longitud = data.Labels.length
var listainfo
var listainfo2 = ""
console.log(longitud)
let datosrekog = [data.Labels.length-1]
//Extraemos unicamente los datos necesarios del reconocimiento
for (var i =0; i < data.Labels.length; i++){
listainfo = data.Labels[i].Name + " ==> "
+data.Labels[i].Confidence.toFixed(2)
datosrekog[i]=listainfo
}
console.log(datosrekog)
global.reconocimiento = datosrekog.toString()
;// Respuesta exitosa
});
```

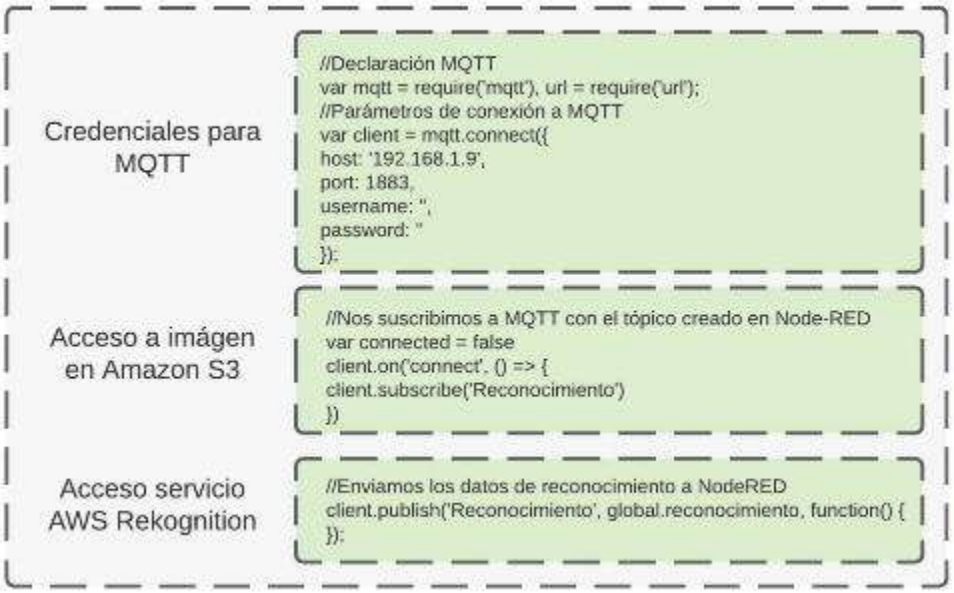
Nota: Código de acceso a base de dato Amazon S3 y Amazon Rekognition

Cuando la imagen se encuentra guardada en la base de datos procedemos a crear un programa en NodeJS donde insertamos las credenciales de AWS, luego accedemos a Amazon S3 y extraemos la imagen en formato .jpg, así accedemos de igual manera al servicio de Amazon Rekognition donde se sube la imagen y luego extraemos los parámetros de reconocimiento en formato JSON, cabe decir que estos servicios funcionan como sistemas RESTful, el código para este paso se lo aprecia a continuación:

De esa manera tenemos el JSON del reconocimiento y mediante el servicio MQTT enviamos desde NodeJS a Node-RED, donde se necesita suscribir al tópico 'Reconocimiento' que está configurado en el nodo de entrada MQTT en Node-RED y luego publicar el JSON como se observa en la figura 103.

Figura 103

Código en NodeJS para acceder al servicio de mensajería y comunicar NodeJS con Node-RED

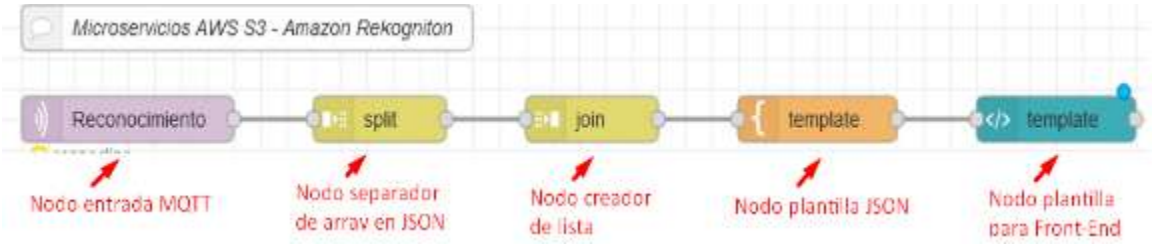


Nota: Se envía el JSON del reconocimiento de imágenes desde NodeJS a Node-RED

Al llegar la información en Node-RED se acomodan los parámetros a manera de listado para ser enviados hacia el Front-End, como se observa en la figura 104.

Figura 104

Recepción de parámetros de reconocimiento enviados a Node-RED



Nota: Recepción de parámetros de reconocimiento en un nodo MQTT de entrada en Node-RED.

Servicio de geolocalización. A través del servicio de mensajería se puede obtener los datos de los sensores, en donde en formato JSON se puede verificar los parámetros de latitud con la letra ‘A’ y la longitud con la letra ‘L’ observando el formato en la figura 105.

Figura 105

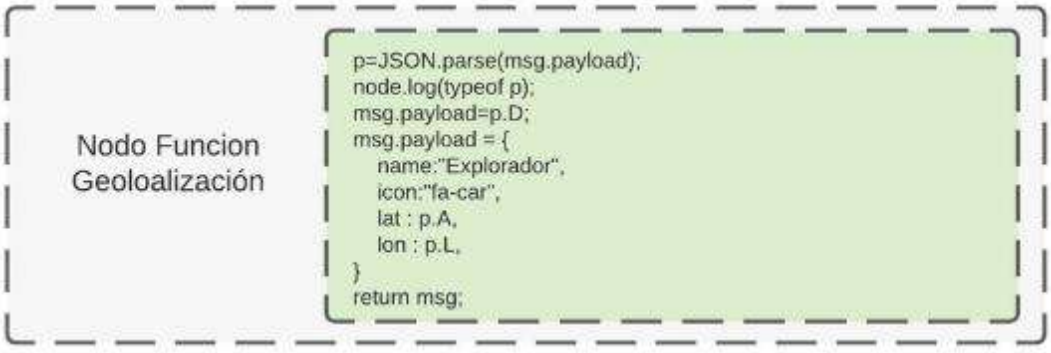
Formato JSON con datos de Latitud y Longitud

Nota: Así es como viajan los datos desde la capa Edge Computing hasta Cloud Computing

Con el nodo MQTT de entrada en Node-RED se recibe el JSON y se utiliza un nodo función en donde se pueda separar los parámetros requeridos para el servicio de geolocalización, el código utilizado está expuesto en la figura 106 donde también se aprecia que se modificó el icono de representación del objeto por un auto denominado “fa-car”.

Figura 106

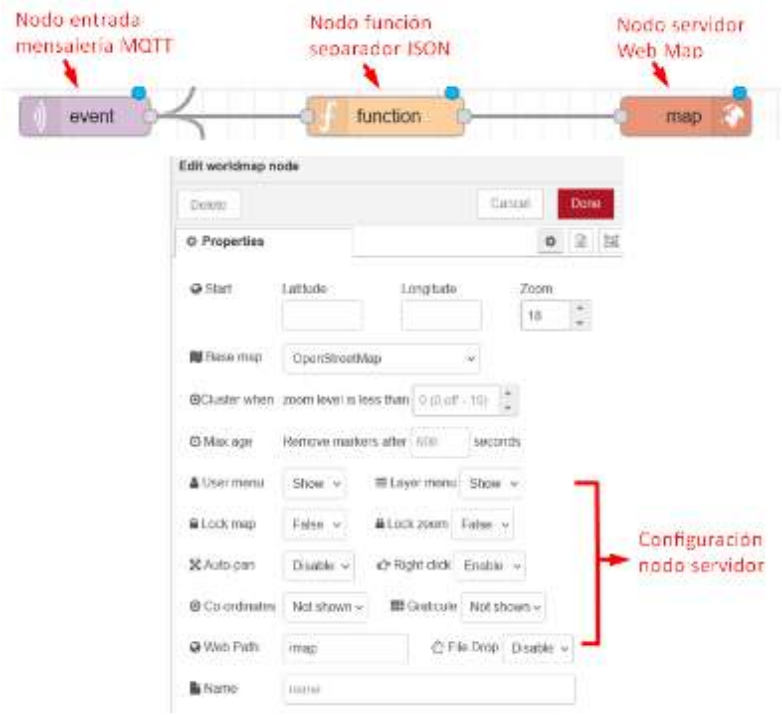
Código en el nodo función para separar datos de geolocalización



Nota: Se presenta la configuración del nodo donde se separa los valores de latitud y longitud del JSON y además se configura el ícono de un carro para mostrar la ubicación.

Figura 107

Nodos para configurar servicio de geolocalización en Node-RED



Nota: Se presenta la estructura de los nodos utilizados, así como la configuración del nodo para conexión con el servicio Web Map

Como ya se mencionó en la etapa de diseño Node-RED tiene a disposición un nodo que trabaja con un servicio de Web Map representado en la figura 107 el cual requiere la información de latitud y longitud para representarla en el mapa y además tiene a disposición otro nodo para poder representarlo en el Front-End.

Sensores e Instrucciones. Al igual que el caso anterior para el sensor de proximidad se tiene el JSON que viene desde la capa Edge Computing representado como la figura 108 representando a la distancia con la letra 'D'.

Figura 108

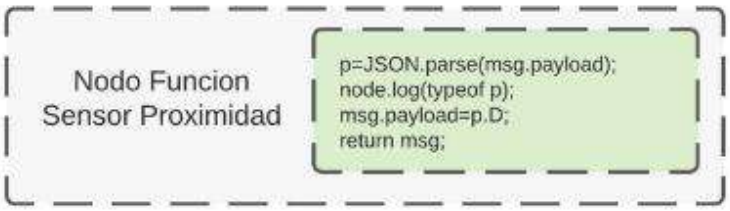
Formato JSON con el dato de distancia

Nota: Diferenciación del parámetro de distancia en el JSON

Aplicamos de igual manera una función con el código de la figura 109 para extraer el dato y luego enviarlo al Front-End para que el usuario lo visualice.

Figura 109

Nodo función para separar el dato de distancia



Nota: Se presenta la configuración del nodo función para extraer el dato de distancia.

Para las instrucciones el proceso difiere pues ahora el dato llega desde el Front-End por lo que se debe identificar cada valor y guardarlo con una variable específica como se ve en la tabla 31.

Tabla 31

Asignación de valores específicos a las instrucciones en la capa Cloud Cmputing

Instrucción proveniente del Front-End	Valor asignado en capa Cloud Computing
Adelante	1
Atrás	2
Izquierda	3
Derecha	4
Levantar mano	5
Bajar mano	6
Girar brazo a la izquierda	7
Girar brazo a la derecha	8
Cerrar Pinzas	9
Abrir Pinzas	0
Levantar brazo	B
Bajar brazo	C
Posición Home del Robot	H
Encendido de luz	E
Apagado de luz	F

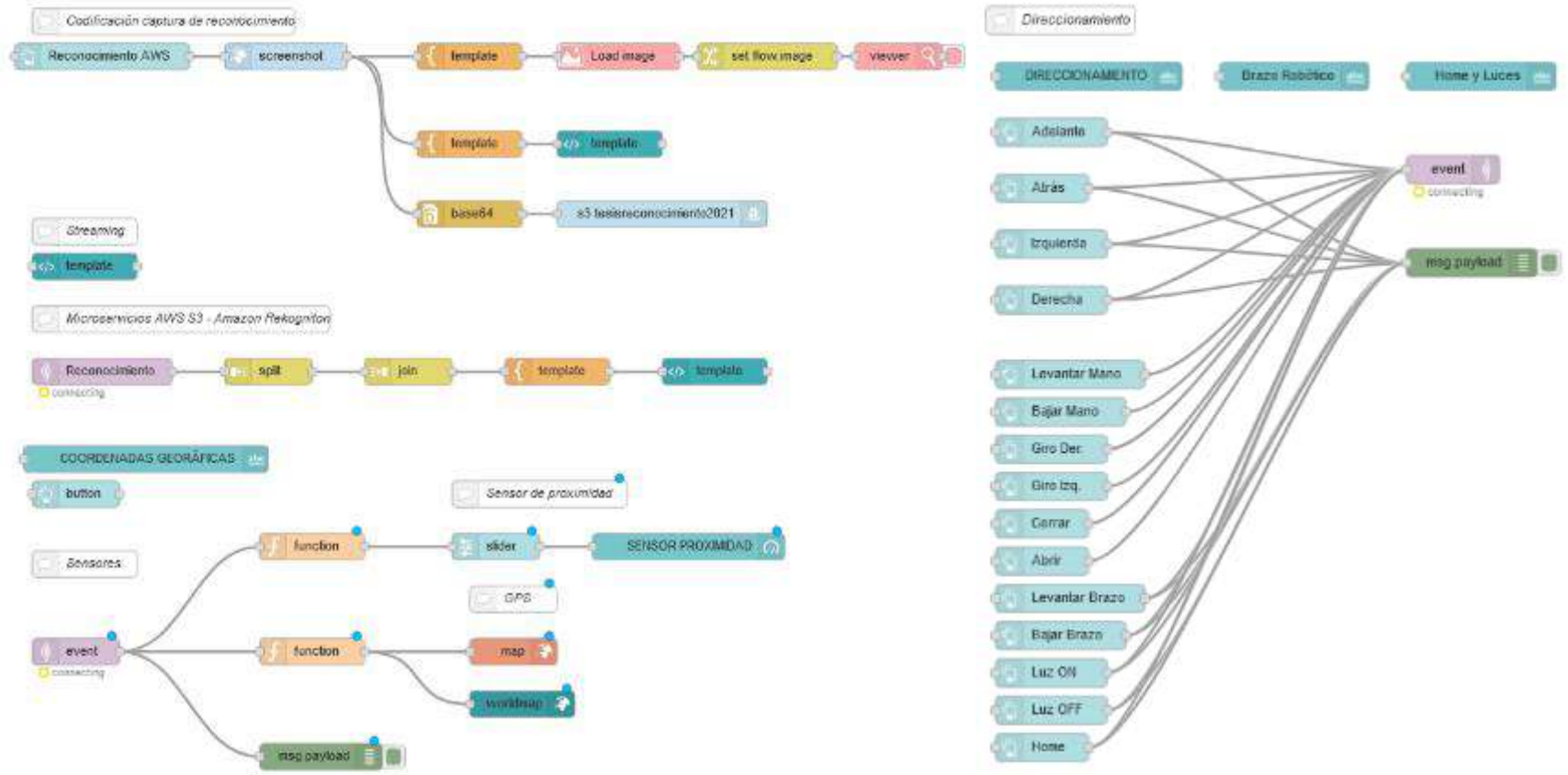
Nota: La configuración del nodo permite asignar el valor correspondiente, pero al provenir del Front-End esta configuración se la verá en la implementación del Front-End.

Integración

Finalmente, en la figura 110 se puede mostrar el resultado obtenido de la conexión de nodos en la herramienta de Node-RED que el principal orquestador.

Figura 110

Orquestación en la herramienta Node-RED



Nota: Se presenta la estructura de funcionamiento en paralelo para la capa Cloud Computing

Construcción de Front-End

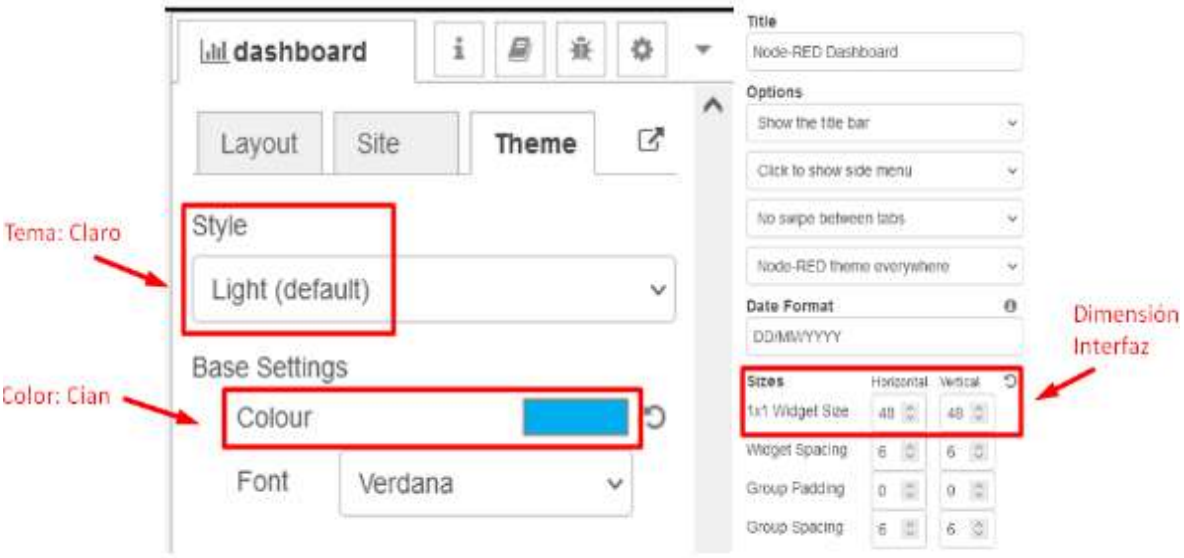
Como ya se definió el diseño de la interfaz se procede a explicar el procedimiento para poder implementar en un dashboard como librería adicional en Node-RED. En primera instancia se mostrará la configuración de los nodos para la interfaz y posterior se mostrará como se colocaron los distintos campos en la interfaz y el producto final.

Configuración Front-End

En el caso de esta librería Dashboard en Node-RED se puede modificar los colores a manera de tema predispuestos, donde se escogió que el tema sea claro y que los colores a utilizarse para la interfaz sean el blanco y cian, por otro lado, también se configura las opciones de visualización del título y el tamaño de la plantilla que en este caso se la hace por cuadrículas escogiendo que las dimensiones sean 48x48 cuadrículas como se observa en la figura 111.

Figura 111

Configuración de colores y dimensiones de la interfaz



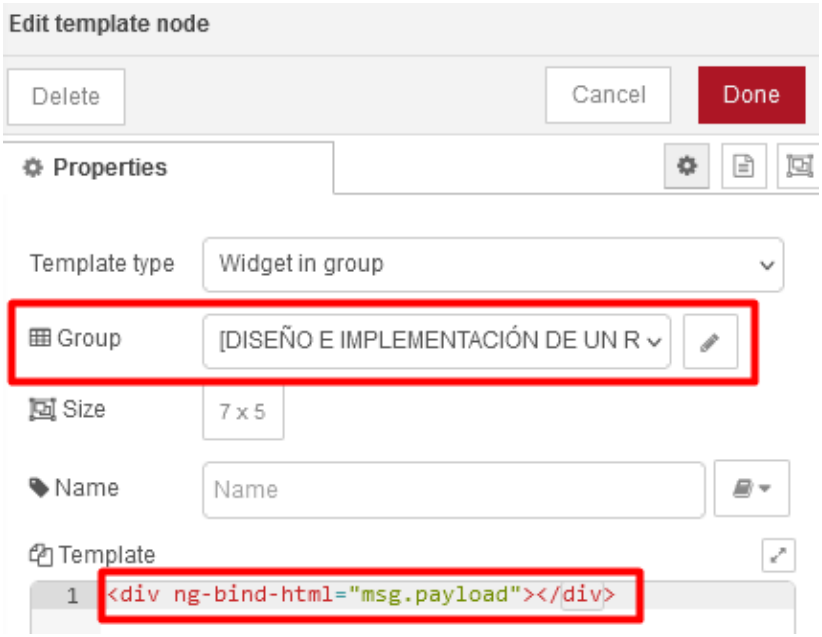
Nota: Se presenta los cuadros de edición de tema y sitio del dashboard en Node-RED.

Campo de texto

Reconocimiento de Imágenes con AWS Rekognition. El JSON que se obtiene del servicio de AWS Rekognition dividido en forma de lista desde la capa Cloud Computing para lo que se utiliza un nodo ‘Plantilla’ para colocar la lista en un sector de la interfaz y otro nodo ‘Plantilla’ para colocar a la captura de igual manera en la interfaz.

Figura 112

Configuración de nodo para mostrar la captura de video y los parámetros de reconocimiento.



Nota: En la sección grupo se debe apuntar a una parte específica del dashboard y en el template solo colocar que se visualice el payload que será el JSON del reconocimiento de imágenes.

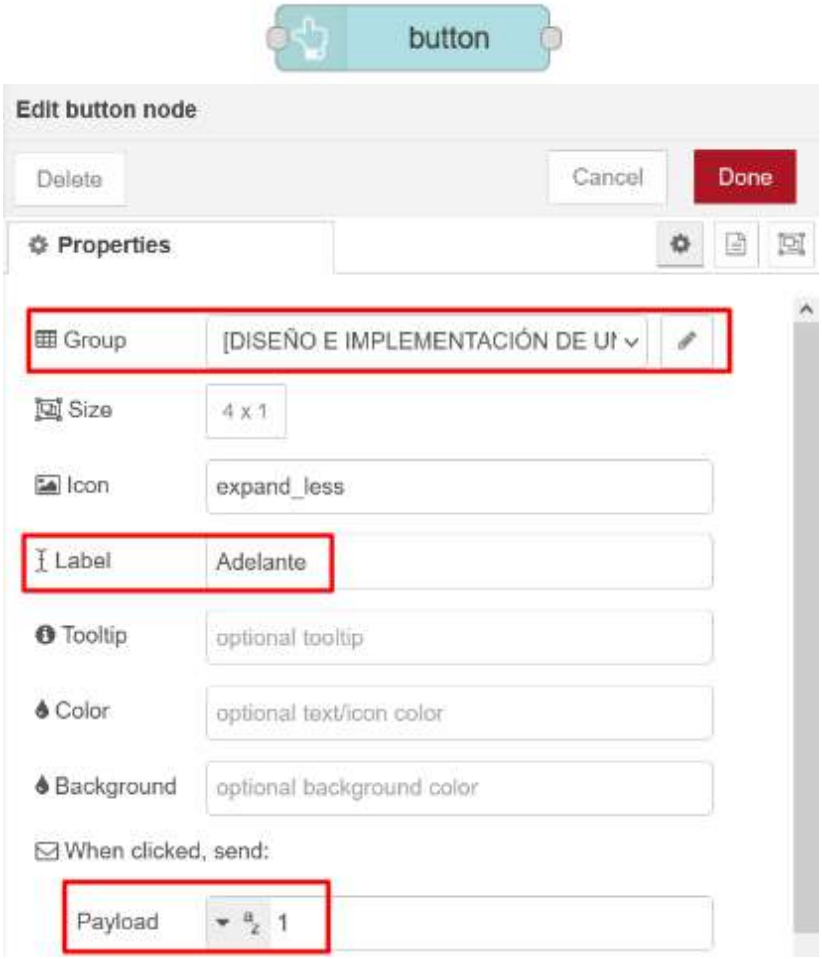
Campo de control

Los botones se den asignar a un valor correspondiente y tanto el nombre de la instrucción como su carácter se los puede ver en la tabla 31, para esto utilizamos el nodo ‘button’ de la librería dashboard que no solo permite representar un botón en la interfaz,

sino que también permite asignarle un valor específico para identificarlo y luego utilizarlo en la capa Cloud Computing y enviarlo a la capa Fog Computing.

Figura 113

Configuración de nodo 'button' para instrucciones del robot



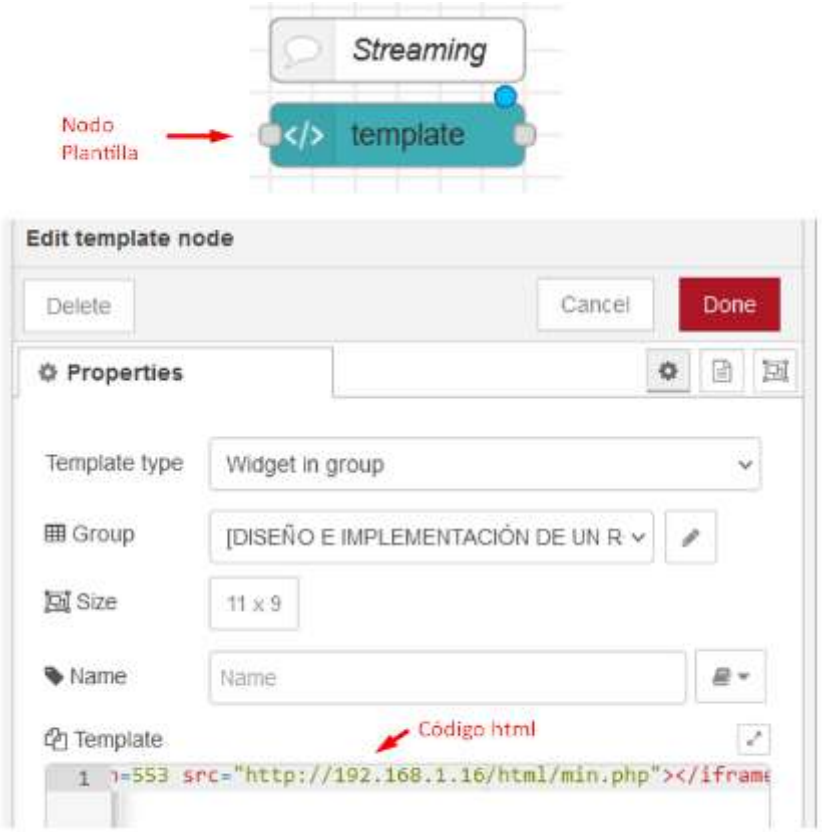
Nota: Se configura el grupo asignado a la interfaz creada, el label que es el nombre con el cual aparecerá el botón, y en payload se asigna el valor específico.

Campo de monitoreo

Video Streaming. Como ya se ha venido indicando para el video streaming se hace un llamado a la ruta HTTP, por lo que para la interfaz se lo hace de la misma manera para visualizar el video.

Figura 114

Configuración servicio video streaming para el Front-End



Nota: La sección grupo es donde se asigna a la interfaz creada y en template se configura las dimensiones y la ruta HTTP del video streaming.

Detector de obstáculos. Con la información tratada en JSON que ya se vio en la implementación de la capa Cloud Computing, llevamos el parámetro de distancia a un nodo ‘slider’ y luego a un nodo ‘gauge’. En el nodo ‘Slider’ se configura el rango de distancia que sería de 0 a 3 metros con pasos configurados en centímetros, mientras que en el nodo ‘Gauge’ que es un indicador para la interfaz se modifican los colores, rojo cuando el objeto este a menos de 1 metro, amarillo cuando el objeto este entre 1 metro a 2 metros de distancia y el color verde cuando el objeto este a más de 2 metros. En la figura 115 se puede ver los nodos y la configuración de los mismos.

Figura 115

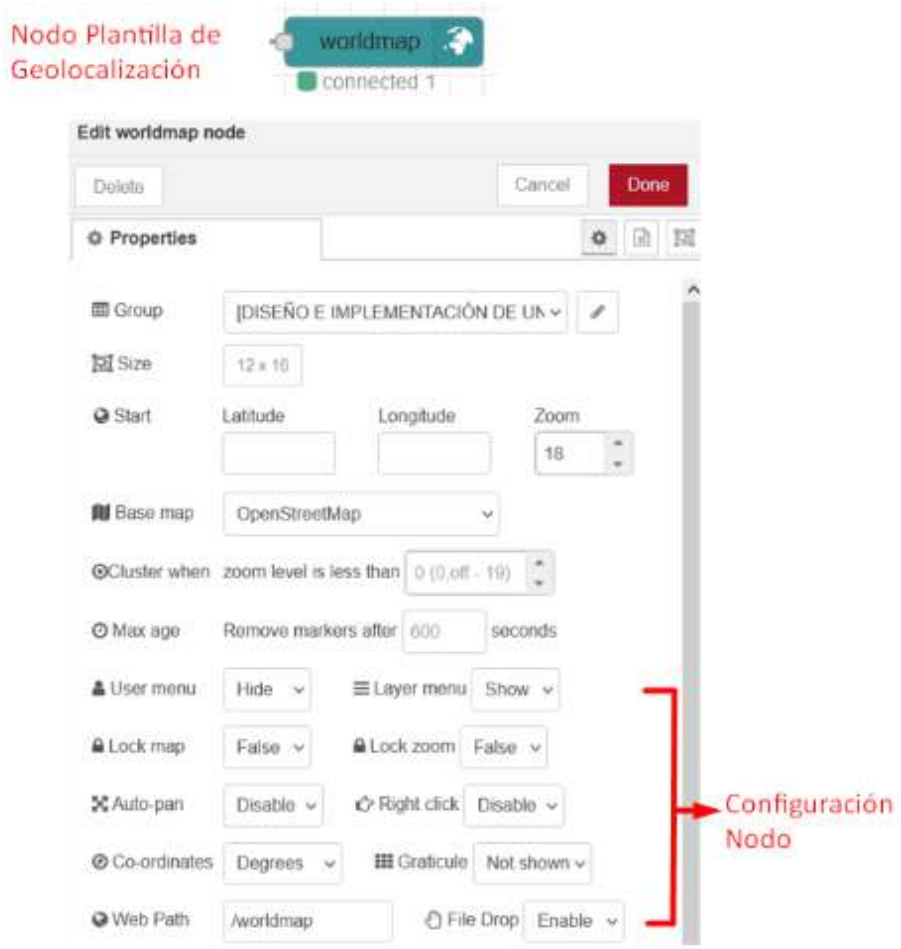
Configuración nodo de sensor de proximidad en Front-End

Nota: Se indica la configuración del nodo Slider y el nodo Gauge.

Geolocalización. Así como se tiene el parámetro de distancia pasa de igual manera con los parámetros de latitud y longitud que llegan dentro del formato JSON, estos son llevados al nodo WorldMap el cual trabaja con el servicio Web Map gracias a que Node-RED permite la instalación de una librería, aquí se tienen diversas configuraciones como el zoom, mostrar menú de uso, tipos de mapas, formato de las coordenadas, movilidad del mapa entre otras que son a conveniencia del usuario sin embargo nosotros solo configuramos el formato de coordenadas en grados para que no exista confusión. Es importante considerar que el dato recibido tanto de latitud y longitud tenga por lo mínimo cuatro cifras decimales esto con el objeto de que la precisión en la localización sea la más cercana y el rango del radio de error disminuya pues dependiendo de los sensores esto puede variar de cinco hasta los quince metros esto ya es a consideración del fabricante.

Figura 116

Nodo de geolocalización para el Front-End



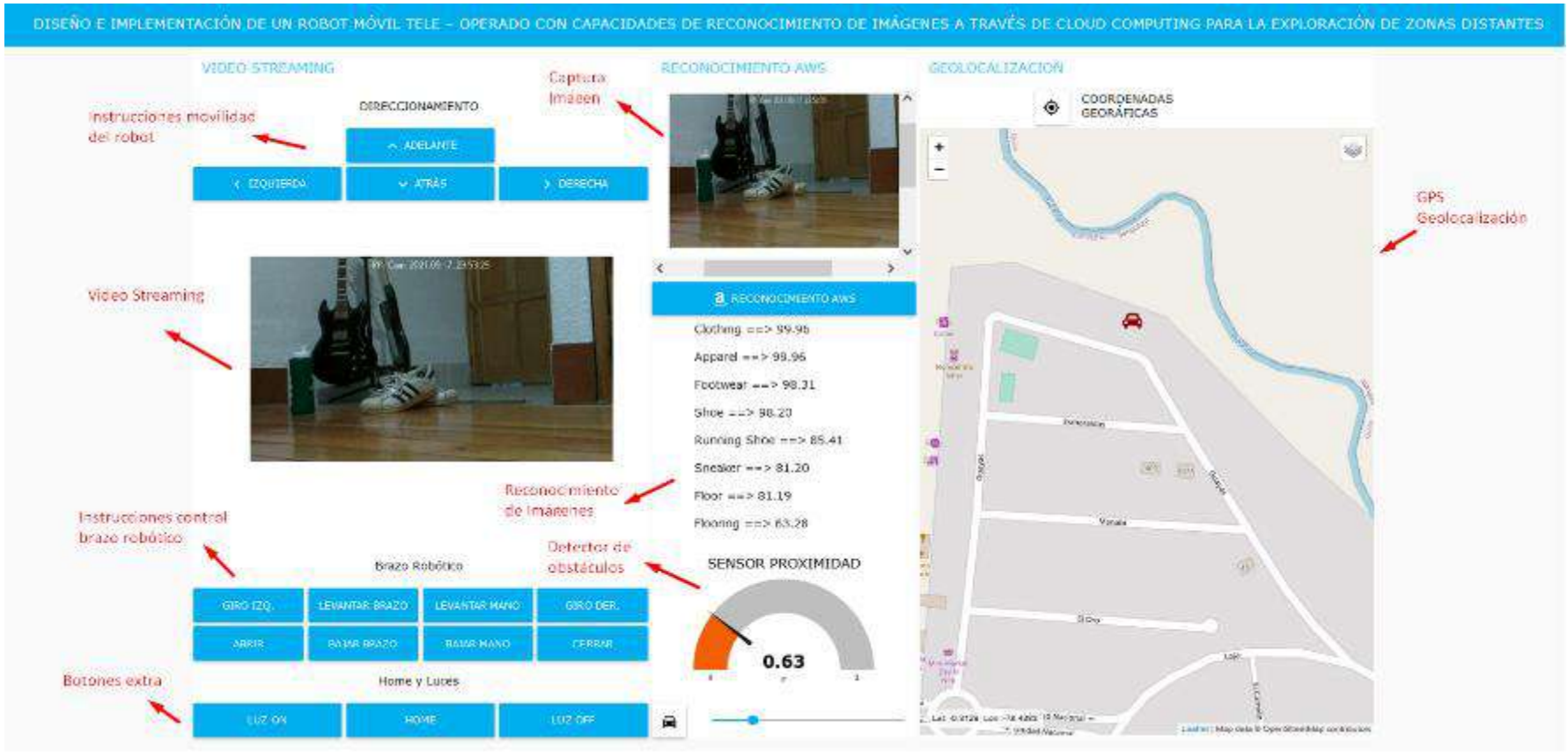
Nota: Se indica la configuración del nodo de geolocalización

Presentación final del Front-End

Como ya se estableció en la sección de diseño del Front-End con cada uno de los pasos realizados, la distribución de los campos se estableció de la misma manera para la implementación teniendo como resultado la interfaz expuesta en la figura 117 que será la utilizada por el usuario en el presente trabajo.

Figura 117

Implementación de la interfaz en el Front-End



Nota: Se presenta la HMI presentada al usuario para poder interactuar con el proyecto realizado.

Capítulo 5: Pruebas

Pruebas de funcionamiento

Pruebas de tele-operación

En las pruebas de tele-operación se optó por adecuar un registro en donde se envíen las instrucciones desde la interfaz para el traslado del robot y control del brazo robótico midiendo los tiempos de respuesta con la cantidad de éxitos y fallas en 10 interacciones por cada instrucción. De la tabla 32 hasta la tabla 46 se presentan los datos recolectados durante la prueba.

Como la transmisión de instrucciones del HMI hacia el robot se las realiza a través del uso de la tecnología inalámbrica LoRa implementada en los Arduinos Heltec Wireless Stick Lite con un chip LoRa SX1276, se estableció un rango de distancia para estas pruebas de 0 a 30 metros.

Tabla 32

Resultados para la instrucción "Adelante" del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	1980 ms	✓	
2	2240 ms	✓	
3	2120 ms	✓	
4	---		X
5	2210 ms	✓	
6	1380 ms	✓	
7	980 ms	✓	
8	2880 ms	✓	
9	1480 ms	✓	
10	---		X

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten movilizar al robot hacia adelante.

Tabla 33

Resultados para la instrucción "Atrás" del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	1620 ms	✓	
2	1650 ms	✓	
3	1850 ms	✓	
4	2110 ms	✓	
5	1550 ms	✓	
6	1320 ms	✓	
7	1830 ms	✓	
8	1750 ms	✓	
9	---		X
10	1270 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten movilizar al robot hacia atrás.

Tabla 34

Resultados para la instrucción "Izquierda" del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	2070 ms	✓	
2	970 ms	✓	
3	1660 ms	✓	
4	1820 ms	✓	
5	2120 ms	✓	

Interacción	Tiempo de respuesta	Éxito	Falla
6	1740 ms	✓	
7	2130 ms	✓	
8	1350 ms	✓	
9	2300 ms	✓	
10	1810 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten girar la dirección del robot hacia la izquierda.

Tabla 35

Resultados para la instrucción "Derecha" del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	910 ms	✓	
2	1730 ms	✓	
3	2100 ms	✓	
4	1650 ms	✓	
5	1020 ms	✓	
6	1940 ms	✓	
7	---		X
8	2030 ms	✓	
9	2400 ms	✓	
10	830 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten girar la dirección del robot hacia la derecha.

Tabla 36

Resultados para la instrucción “Giro Izquierda” del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	2100 ms	✓	
2	1260 ms	✓	
3	2040 ms	✓	
4	1780 ms	✓	
5	2070 ms	✓	
6	2130 ms	✓	
7	980 ms	✓	
8	1390 ms	✓	
9	1650 ms	✓	
10	2020 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten girar el brazo robótico hacia la izquierda.

Tabla 37

Resultados para la instrucción “Giro Derecha” del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	1510 ms	✓	
2	1430 ms	✓	
3	1400 ms	✓	
4	1930 ms	✓	
5	1490 ms	✓	

Interacción	Tiempo de respuesta	Éxito	Falla
6	1640 ms	✓	
7	1820 ms	✓	
8	1220 ms	✓	
9	2010 ms	✓	
10	830 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten girar el brazo robótico hacia la derecha.

Tabla 38

Resultados para la instrucción "Levantar Brazo" del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	1260 ms	✓	
2	1920 ms	✓	
3	1790 ms	✓	
4	---		X
5	1541 ms	✓	
6	2200 ms	✓	
7	1790 ms	✓	
8	1690 ms	✓	
9	1890 ms	✓	
10	1930 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten levantar el brazo robótico.

Tabla 39

Resultados para la instrucción “Bajar Brazo” del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	1580 ms	✓	
2	---		X
3	2120 ms	✓	
4	1440 ms	✓	
5	1970 ms	✓	
6	1930 ms	✓	
7	---		X
8	1910 ms	✓	
9	1240 ms	✓	
10	2180 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten bajar el brazo robótico.

Tabla 40.

Resultados para la instrucción “Levantar Mano” del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	1800 ms	✓	
2	1450 ms	✓	
3	2180 ms	✓	
4	1710 ms	✓	
5	1470 ms	✓	

Interacción	Tiempo de respuesta	Éxito	Falla
6	2050 ms	✓	
7	1810 ms	✓	
8	1830 ms	✓	
9	1990 ms	✓	
10	1580 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten levantar la mano del robot.

Tabla 41

Resultados para la instrucción "Bajar Mano" del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	2010 ms	✓	
2	2210 ms	✓	
3	1730 ms	✓	
4	1350 ms	✓	
5	---		X
6	2100 ms	✓	
7	1810 ms	✓	
8	810 ms	✓	
9	1010 ms	✓	
10	1830 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten bajar la mano del robot.

Tabla 42

Resultados para la instrucción “Abrir” del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	680 ms	✓	
2	960 ms	✓	
3	1520 ms	✓	
4	1720 ms	✓	
5	2260 ms	✓	
6	---		X
7	1860 ms	✓	
8	2100 ms	✓	
9	2230 ms	✓	
10	1520 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten movilizar abrir las pinzas del robot.

Tabla 43

Resultados para la instrucción “Cerrar” del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	1540 ms	✓	
2	2140 ms	✓	
3	1160 ms	✓	
4	820 ms	✓	
5	1640 ms	✓	

Interacción	Tiempo de respuesta	Éxito	Falla
6	1630 ms	✓	
7	980 ms	✓	
8	830 ms	✓	
9	850 ms	✓	
10	910 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten cerrar las pinzas del robot.

Tabla 44

Resultados para la instrucción "Luz ON" del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	1360 ms	✓	
2	2100 ms	✓	
3	2140 ms	✓	
4	1710 ms	✓	
5	2080 ms	✓	
6	1140 ms	✓	
7	---		X
8	1570 ms	✓	
9	1940 ms	✓	
10	2220 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten encender las luces del robot.

Tabla 45

Resultados para la instrucción “Luz OFF” del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	920 ms	✓	
2	1520 ms	✓	
3	970 ms	✓	
4	1740 ms	✓	
5	1240 ms	✓	
6	1020 ms	✓	
7	---		X
8	1130 ms	✓	
9	1810 ms	✓	
10	1910 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten apagar las luces del robot.

Tabla 46

Resultados para la instrucción “Home” del robot

Interacción	Tiempo de respuesta	Éxito	Falla
1	2300 ms	✓	
2	2170 ms	✓	
3	1870 ms	✓	
4	1270 ms	✓	
5	1920 ms	✓	

Interacción	Tiempo de respuesta	Éxito	Falla
6	---		X
7	1850 ms	✓	
8	1130 ms	✓	
9	1040 ms	✓	
10	1790 ms	✓	

Nota: La presente tabla da a conocer todos los resultados obtenidos con 10 interacciones que permiten posicionar al robot en una posición estable.

Analizando los resultados se pudo ver a detalle del que el tiempo de respuesta varía, es decir el tiempo que tarda desde que se envió la instrucción hasta ser ejecutada por el robot y que en que existe ocasiones en las que las instrucciones no son ejecutadas, a su vez se establece la cantidad de instrucciones que tuvieron éxito y las que fallaron. A manera de resumen se presenta el promedio de los tiempos de respuesta y el porcentaje de éxito y fracaso de cada instrucción en la tabla 47.

Tabla 47

Resultado total de las iteraciones en las pruebas de tele-operación

Instrucción	Promedio Tiempo de Respuesta	% Éxito	% Falla
Adelante	1908.75 ms	80	20
Atrás	1661.11 ms	90	10
Izquierda	1797 ms	100	0
Derecha	1623.33 ms	90	10
Giro Izquierda	1742 ms	100	0
Giro Derecha	1528 ms	100	0
Levantar Brazo	1779 ms	90	10

Instrucción	Promedio Tiempo de Respuesta	% Éxito	% Falla
Bajar Brazo	1796.25 ms	80	20
Levantar Mano	1787 ms	100	0
Bajar Mano	1651.11 ms	90	10
Abrir	1650 ms	90	10
Cerrar	1250 ms	100	0
Luz ON	1806.66 ms	90	10
Luz OFF	1362.22 ms	90	10
Home	1704.44 ms	90	10

Nota: Se muestran los resultados promedio de cada una de las interacciones de cada instrucción probada.

Con los resultados de la tabla 47 se determinó que la mayoría de instrucciones tienen un porcentaje de éxito al ejecutarse del 90% al 100% lo cual es positivo pues al utilizar la tecnología inalámbrica LoRa debido a las distancias, movimientos y factores externos del medio siempre existe una influencia en la pérdida de datos al momento de transmitirlos, por otra parte el promedio de tiempo de respuesta no supera los 2 segundos, lo cual sugiere que el programa en los Arduino Heltec Wireless Stick Lite fue implementado correctamente para transmisión y recepción de datos vía LoRa.

Pruebas del sistema de reconocimiento de imágenes

En el reconocimiento de imágenes al estar utilizado el servicio de AWS Rekognition se toma una captura del video streaming y posterior se genera como resultado una lista de los objetos, ambientes y personas reconocidas.

Para evaluar la capacidad de reconocimiento se mostraron imágenes referidas a un dormitorio, calle, área de juegos y jardín, que se evaluaron según la cantidad de cosas que se reconocieron y la confiabilidad que presentaron.

Prueba de reconocimiento 1.

Figura 118

Reconocimiento dormitorio

Nota: Captura de video streaming realizada en un dormitorio

La figura 118 al momento de ser analizada por AWS Rekognition se obtuvieron los resultados de la tabla 48, en donde se muestran gran variedad de etiquetas de objetos reconocidos, a pesar de ello lo requerido principalmente en el trabajo presentado es la detección de personas y ambientes. Los resultados fueron satisfactorios pues se tiene un 99.9% de confiabilidad que hay una persona y 74.1% de confiabilidad.

En la columna de Acertado de la tabla 48 se puede apreciar que en su mayoría hubo coincidencia con los objetos existentes en la habitación, pero si existieron algunos errores como es la detección de un sofá, plywood que no se consideraron críticos pues tienen un parecido a la imagen, un error un poco mayor podría haber sido la confusión del cuarto con una sala, aunque esa etiqueta es la final y tiene el menor porcentaje de

confiabilidad por lo cual es despreciable, lo que dio como resultado de esta prueba un reconocimiento del 86.6%.

Tabla 48

Resultados de reconocimiento de un dormitorio

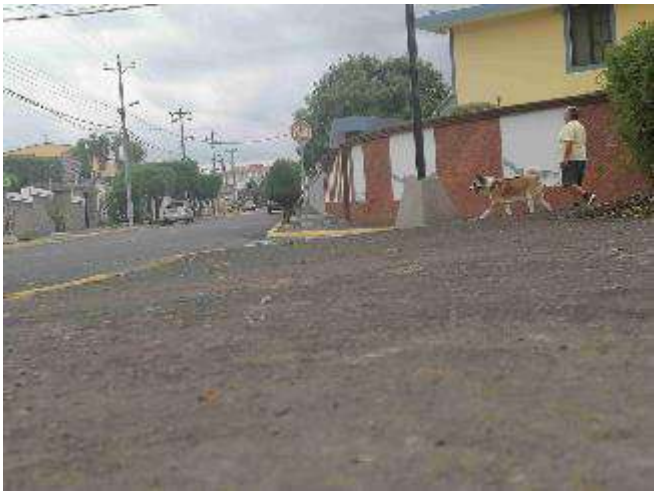
Eiquetas	Confiabilidad	Acertado SI / NO
Flooring / Piso	99.9 %	SI
Person / Persona	99.9 %	SI
Human / Humano	99.9 %	SI
Wood / Madera	99.9 %	SI
Floor / Suelo	99.85 %	SI
Hardwood / Madera dura	99.7 %	SI
Furniture / Muebles	93.4 %	SI
Couch / Sofá	87.9 %	NO
Bedroom / Cuarto	74.1 %	SI
Indoors / Interior	74.1 %	SI
Room / Habitación	74.1 %	SI
Bed / Cama	70 %	SI
Plywood	61.4%	NO
Dorm Room / Dormitorio	59 %	SI
Living Room / Sala	57 %	NO

Nota: La presente tabla da a conocer todos los resultados obtenidos al pasar la figura 118 por AWS Rekognition

Prueba de reconocimiento 2.

Figura 119

Reconocimiento calle



Nota: Captura de video streaming realizada en una calle

AWS Rekognition generó los resultados de la tabla 49 al importar la captura de la figura 119, en donde se muestran las etiquetas de objetos reconocidos, y como el caso anterior lo principal es que se detectó una persona con confiabilidad del 99.1% y una carretera con confiabilidad del 99.4%.

Tabla 49

Resultados de reconocimiento de una calle

Eiquetas	Confiabilidad	Acertado SI / NO
Reload / Carretera	99.4 %	SI
Person / Persona	99.1 %	SI
Human / Humano	99.1 %	SI
Tarmac / pista o vía	97.6 %	SI
Asphalt / Asfalto	97.6 %	SI
Dog / Perro	95.5 %	SI

Eiquetas	Confiabilidad	Acertado SI / NO
Mammal / Mamífero	95.5 %	SI
Animal / Animal	95.5 %	SI
Canine / Canino	95.5 %	SI
Pet / Mascota	95.5 %	SI
Intersection / Intersección	89 %	SI
Outdoor / Exterior	74.2 %	SI
Urban / Urbano	73.3 %	SI
Vehicule / Vehículo	69.6 %	SI
Automobile / Automóvil	69.6 %	SI

Nota: La presente tabla da a conocer todos los resultados obtenidos al pasar la figura 119 por AWS Rekognition

En la columna de Acertado de la tabla 49 se verificó que todos los elementos fueron coincidentes deduciendo que se tuvo un reconocimiento del 100%.

Prueba de reconocimiento 3.

Figura 120

Reconocimiento área de juegos

Nota: Captura de video streaming realizada en un área de juegos

Al momento de indagar con la figura 120 por AWS Rekognition se obtienen los resultados de la tabla 50, apareciendo pocas etiquetas de objetos reconocidos, como es debido se establece interés en que no se detectó personas pues no las había y pero si un área de juegos con confiabilidad del 71.5%.

Tabla 50.

Resultados de reconocimiento de un área de juegos

Eiquetas	Confiabilidad	Acertado SI / NO
Swing / Columpio	78.1 %	SI
Toy / Juguete	78.1 %	SI
Grass / Césped	75.5 %	SI
Plant / Plantas	75.5 %	SI
Play Area / Área de juegos	71.5 %	SI
Playground / Campo de juegos	71.5 %	SI

Nota: La presente tabla da a conocer todos los resultados obtenidos al pasar la figura 120 por AWS Rekognition

En la columna de Acertado de la tabla 50 se aprecia que todos los elementos fueron coincidentes, no obstante, hubo algunos elementos que no se detectaron debido a la calidad de la imagen al ser capturada en contraluz. Se obtuvo en esta prueba un reconocimiento del 100%.

Prueba de reconocimiento 4. Los resultados al importar la captura de la figura 121 obtenidos de AWS Rekognition se contemplan en la tabla 51, en donde hay pocas etiquetas de objetos reconocidos, estableciendo interés en que no se detectó personas pues no las había y se identificó un jardín con confiabilidad del 81% teniendo un porcentaje de confiabilidad elevado.

Figura 121

Reconocimiento jardín

Nota: Captura de video streaming realizada en un jardín

Tabla 51

Resultados de reconocimiento de un jardín

Eiquetas	Confiabilidad	Acertado SI / NO
Yard / Patio	99 %	SI
Ourtdoors / Exterior	99 %	SI
Nature / Naturaleza	99 %	SI
Tree / Árbol	87.9 %	SI
Plant / Planta	87.8 %	SI
Garden / Jardín	81 %	SI
Grass / Césped	79.7 %	SI
Vegetation / Vegetación	75.3 %	SI
Ground / Tierra	73.4 %	SI

Nota: La presente tabla da a conocer todos los resultados obtenidos al pasar la figura 121 por AWS Rekognition

En la columna de Acertado de la tabla 51 se puede apreciar que todos los elementos fueron coincidentes deduciendo que se tuvo un reconocimiento del 100%.

Con las pruebas realizadas para el reconocimiento de imágenes se pudo notar que existe un alto grado de dependencia de la calidad de imagen al igual que el ángulo de captura pues si esta contraluz esto afectará a un más a la calidad y por ende dificulta el buen reconocimiento de los objetos. Otro punto a destacar es que en un lugar pueden existir muchos elementos y algunos tal vez por su tamaño o por el ángulo de posición no son reconocidos.

Pruebas del sistema de posicionamiento

Tabla 52

Resultados prueba de posicionamiento por GPS

Prueba	GPS GT-U7		Dispositivo conectado a Internet		Distancia de diferencia
	Latitud	Longitud	Latitud	Longitud	
1	-0.31213	-78.44008	-0.31210	-78.44009	2.05 m
2	-0.31210	-78.44012	-0.31219	-78.44012	3.29 m
3	-0.31298	-78.43975	-0.31300	-78.43973	2.97 m
4	-0.31736	-78.44076	-0.31738	-78.44076	2.62 m
5	-0.31509	-78.44139	-0.31505	-78.44137	3.37 m
6	-0.31527	-78.44023	-0.31526	-78.44019	4.12 m
7	-0.31677	-78.44156	-0.31680	-78.44159	3.68 m
8	-0.31477	-78.43860	-0.31477	-78.43863	3.21 m
9	-0.31688	-78.44045	-0.31685	-78.44048	3.61 m
10	-0.31792	-78.43901	-0.31789	-78.43903	3.42 m

Nota: La presente tabla muestra los resultados de geolocalización de un mismo punto capturado por el módulo GPS GT-U7 en comparativa con un dispositivo móvil.

Para evaluar el rendimiento de nuestro sistema de posicionamiento proporcionado por el módulo GPS GT-U7 se realizó una comparativa de los datos de latitud y longitud obtenidos del sensor con los de un dispositivo que se encuentre conectado a internet, la información fue implementada en el servidor de Google Maps donde se observó si existía algún tipo de variación por distancia en un punto exacto. Las pruebas fueron realizadas en 10 diferentes puntos de ubicación expuestos en la tabla 52.

Se observó que la mayor variación está en un radio de 4.12 metros y la menor de 2.05 metros para poder tener la geolocalización a nivel mundial lo cual resultó satisfactorio, pues en teléfonos móviles el radio de variación es de 10 metros o mayor dependiendo de la marca y operadora, el módulo GPS GT-U7 es de alta sensibilidad y tiene que conectarse a 3 satélites para identificar la posición con latitud y longitud, a pesar de ello se puede conectar a más satélites, en nuestro caso se encontró conectividad de hasta 7 satélites.

Pruebas de usabilidad

Dentro de las pruebas de usabilidad se seleccionó a un grupo de 50 usuarios para que interactúen con la aplicación, y de esta manera puedan llevar a cabo las tareas para las cuales fue diseñada. Posteriormente los usuarios realizaron la respectiva evaluación a través de una encuesta que fue diseñada bajo el SUS (System Usability Scale) o sistema de escalas de usabilidad con lo que se pretende determinar qué tan amigable es la aplicación para el usuario.

El sistema de escalas de usabilidad cuenta con 10 preguntas, las mismas que pueden ser puntuadas de 1 a 5, donde 1 se interpreta como total desacuerdo, mientras que 5 significa total acuerdo. Para obtener los resultados bajo este método, se debe considerar que las preguntas impares tendrán el valor asignado por el usuario menos 1, mientras que para las preguntas pares será de 5 menos el valor asignado por el usuario. Por último, se realiza la sumatoria de estos valores y al resultado final, se lo multiplica por

2.5, por lo que el test completo tendrá una evaluación sobre 100. El valor final, será el promedio de la evaluación de todos los usuarios.

En la tabla 53 se muestran los valores de la evaluación realizada por los 50 usuarios encuestados bajo el sistema SUS.

Tabla 53

Puntaje SUS obtenido de los 50 usuarios evaluados

Usuario	SUS	Usuario	SUS	Usuario	SUS	Usuario	SUS	Usuario	SUS
U1	100	**U11**	62.5	**U21**	87.5	**U31**	100	**U41**	100
U2	100	**U12**	82.5	**U22**	92.5	**U32**	92.5	**U42**	100
U3	87.5	**U13**	77.5	**U23**	92.5	**U33**	95	**U43**	100
U4	100	**U14**	62.5	**U24**	97.5	**U34**	100	**U44**	100
U5	42.5	**U15**	62.5	**U25**	95	**U35**	95	**U45**	100
U6	90	**U16**	95	**U26**	97.5	**U36**	100	**U46**	100
U7	97.5	**U17**	67.5	**U27**	100	**U37**	100	**U47**	20
U8	85	**U18**	87.5	**U28**	87.5	**U38**	100	**U48**	100
U9	87.5	**U19**	92.5	**U29**	87.5	**U39**	100	**U49**	100
U10	95	**U20**	92.5	**U30**	100	**U40**	85	**U50**	100

Nota: La presente tabla muestra los valores obtenidos mediante el sistema SUS, de cada uno de los 50 usuarios encuestados.

Teniendo los valores individuales obtenidos de la evaluación realizada por los usuarios que han manipulado el sistema de forma remota, se obtiene el promedio general que da un valor de 89.85/100, lo que es muy satisfactorio para el proyecto, ya que se traduce en que los usuarios consideran al sistema altamente útil, amigable y manipulable. Teniendo en cuenta que hay aspectos donde se pueden mejorar aún más, para lograr la excelencia.

Pruebas de interfaz

Para esta prueba vamos a considerar específicamente la pregunta número tres del sistema SUS, la cual hace mención a que tan fácil fue usar el sistema, refiriéndonos

específicamente a la interfaz. De esta manera vamos a extraer todos los valores de la pregunta tres, proporcionados por los 50 usuarios encuestados como se muestra en la tabla 54.

Tabla 54

Puntaje de la pregunta tres obtenido de los 50 usuarios evaluados

Usuario	P3	Usuario	P3	Usuario	P3	Usuario	P3	Usuario	P3
U1	5	**U11**	3	**U21**	5	**U31**	5	**U41**	5
U2	5	**U12**	5	**U22**	5	**U32**	5	**U42**	5
U3	5	**U13**	5	**U23**	5	**U33**	5	**U43**	5
U4	5	**U14**	3	**U24**	5	**U34**	5	**U44**	5
U5	2	**U15**	3	**U25**	5	**U35**	5	**U45**	5
U6	5	**U16**	5	**U26**	5	**U36**	5	**U46**	5
U7	5	**U17**	4	**U27**	5	**U37**	5	**U47**	1
U8	3	**U18**	5	**U28**	5	**U38**	5	**U48**	5
U9	3	**U19**	5	**U29**	5	**U39**	5	**U49**	5
U10	4	**U20**	5	**U30**	5	**U40**	5	**U50**	5

Nota: La presente tabla muestra los valores obtenidos de la pregunta tres, enfocada en la facilidad del uso de la interfaz.

Teniendo en cuenta que un puntaje de 5 significa total acuerdo, en este caso con respecto a la facilidad de uso de la interfaz. El promedio obtenido en la pregunta tres por parte los 50 usuarios es de 4.62, lo que nos muestra con optimismo que los usuarios consideran que la interfaz es de fácil uso.

Pruebas de eficiencia

Sabiendo que la eficiencia es la capacidad del sistema de cumplir adecuadamente con la función para la que fue diseñado, vamos a extraer las puntuaciones obtenidas en la pregunta número cinco, la cual menciona la buena integración de las funciones del sistema, lo que se relaciona muy bien, viéndolo desde el punto de vista de eficiencia del sistema.

Tabla 55

Puntaje de la pregunta cinco obtenido de los 50 usuarios evaluados

Usuario	P5	Usuario	P5	Usuario	P5	Usuario	P5	Usuario	P5
U1	5	**U11**	3	**U21**	5	**U31**	5	**U41**	5
U2	5	**U12**	5	**U22**	5	**U32**	5	**U42**	5
U3	4	**U13**	1	**U23**	4	**U33**	4	**U43**	5
U4	5	**U14**	4	**U24**	5	**U34**	5	**U44**	5
U5	4	**U15**	4	**U25**	5	**U35**	4	**U45**	5
U6	5	**U16**	5	**U26**	5	**U36**	5	**U46**	5
U7	5	**U17**	4	**U27**	5	**U37**	5	**U47**	1
U8	5	**U18**	5	**U28**	5	**U38**	5	**U48**	5
U9	5	**U19**	4	**U29**	4	**U39**	5	**U49**	5
U10	5	**U20**	5	**U30**	5	**U40**	4	**U50**	5

Nota: La presente tabla muestra los valores obtenidos de la pregunta cinco, enfocada en la eficiencia del sistema.

El promedio en esta pregunta es de 4.58 que corresponde a un valor muy aceptable, con lo que se puede deducir que el sistema es considerado eficiente en cuanto a la integración de todos los elementos que lo componen.

Pruebas de satisfacción

La satisfacción viene desde el punto de vista que el usuario se haya sentido bien al usar el sistema, para esto haremos referencia con respecto a la pregunta nueve del sistema SUS, la cual menciona la confianza que genera en el usuario la manipulación del sistema.

Al realizar la evaluación se obtuvo un promedio de 4.6, lo que gratificantemente muestra que el sistema fue satisfactorio para los usuarios demostrando que es fácil de utilizar y de gran adaptabilidad.

Tabla 56

Puntaje de la pregunta nueve obtenido de los 50 usuarios evaluados

Usuario	P9	Usuario	P9	Usuario	P9	Usuario	P9	Usuario	P9
U1	5	**U11**	3	**U21**	4	**U31**	5	**U41**	5
U2	5	**U12**	4	**U22**	5	**U32**	4	**U42**	5
U3	5	**U13**	5	**U23**	5	**U33**	5	**U43**	5
U4	5	**U14**	4	**U24**	5	**U34**	5	**U44**	5
U5	3	**U15**	2	**U25**	5	**U35**	5	**U45**	5
U6	5	**U16**	5	**U26**	5	**U36**	5	**U46**	5
U7	5	**U17**	3	**U27**	5	**U37**	5	**U47**	1
U8	4	**U18**	5	**U28**	5	**U38**	5	**U48**	5
U9	5	**U19**	5	**U29**	5	**U39**	5	**U49**	5
U10	5	**U20**	4	**U30**	5	**U40**	4	**U50**	5

Nota: La presente tabla muestra los valores obtenidos de la pregunta nueve, enfocada en la satisfacción del usuario al usar el sistema.

Pruebas de alcance

Pruebas en zonas urbanas

Se realizaron mediciones de cobertura de la comunicación a larga distancia registrando resultados de 3 pruebas, en un entorno que se encontraba rodeado de casa, edificios y circulación de vehículos, con el objetivo de obtener el promedio de RSSI que representa la fuerza de la señal a medida que aumenta la distancia así se pudo notar como se encuentra recibiendo la señal el Gateway desde el robot. El lugar donde se encontraba el Gateway estaba situado en el segundo piso de una casa.

Cabe aclarar que para promediar el valor de dBm primero debemos convertir a un valor escalar obtener el promedio y luego volver a pasarlo al valor en dBm. En la tabla 57 se observa los resultados obtenidos.

Tabla 57

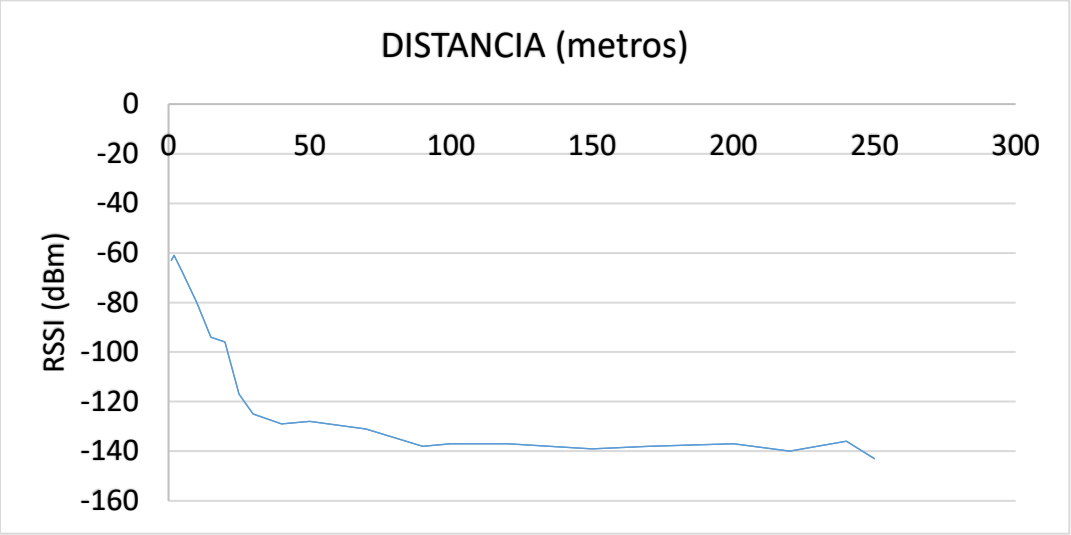
Resultado pruebas de distancia en zonas urbanas

Distancia	RSSI dBm			RSSI Promedio
	Prueba 1	Prueba 2	Prueba 3	
1 m	-60 dBm	-65 dBm	-67 dBm	-63 dBm
2 m	-60 dBm	- 63 dBm	-61 dBm	-61 dBm
5 m	-74 dBm	-71 dBm	-64 dBm	-68 dBm
10 m	-81 dBm	-81 dBm	-79 dBm	-80 dBm
15 m	-98 dBm	- 92 dBm	-93 dBm	-94 dBm
20 m	-100 dBm	-94 dBm	-95 dBm	-96 dBm
25 m	-117 dBm	-115 dBm	-120 dBm	-117 dBm
30 m	-125 dBm	-122 dBm	-130 dBm	-125 dBm
40 m	-129 dBm	-137 dBm	-121 dBm	-129 dBm
50 m	-127 dBm	-123 dBm	-134 dBm	-128 dBm
70 m	-129 dBm	-131 dBm	-131 dBm	-131 dBm
90 m	-143 dBm	-135 dBm	-135 dBm	-138 dBm
100 m	-139 dBm	-137 dBm	-136 dBm	-137 dBm
120 m	-136 dBm	-140 dBm	-134 dBm	-137 dBm
150 m	-140 dBm	-138 dBm	-140 dBm	-139 dBm
170 m	-136 dBm	-140 dBm	-138 dBm	-138 dBm
200 m	-137 dBm	-135 dBm	-138 dBm	-137 dBm
220 m	-138 dBm	-141 dBm	-141 dBm	-140 dBm
240 m	-137 dBm	-137 dBm	-135 dBm	-136 dBm
250 m	-144 dBm	-141 dBm	-143 dBm	-143 dBm

Nota: Se muestran los resultados de 3 pruebas de distancia realizadas en unas zonas urbanas.

Figura 122

Relación distancia y fuerza de señal en comunicación LoRa Arduino Heltec Stick Lite en zona urbana



Nota: Gráfica de los promedios de la fuerza de la señal de comunicación LoRa Arduino Heltec Stick Lite en zonas urbanas

Se analizó que mientras más cerca esté el emisor del receptor LoRa Arduino Heltec Stick Lite la fuerza de señal percibida es alta, a lo que se van alejando aumentando la distancia se observó que la fuerza de la señal va decayendo y llega a un punto en el cual no se consideró factible, como se puede ver en la figura 122 a los 250 metros de distancia lo máximo que se transmitió fueron los datos de geolocalización y sensores, cabe mencionar que desde los 180 metros estos datos comenzaron a tener una mayor latencia de actualización, en cuanto a las instrucciones estas comenzaron a decaer desde los 70 metros y a los 100 metros se tenía muchas interrupciones, no se pudo explotar toda la capacidad del dispositivo LoRa que en el datasheet nos menciona que puede llegar de 500 metros a 700 metros, esto se debe a que el lugar de pruebas se encontraba con la presencia de vehículos y casas que obstaculizan la señal, además de que la antena en el robot aparte de estar en constante movimiento se encontraba muy cerca del suelo

lo cual afecta a la transmisión pues lo que se recomienda es que las antenas sean colocadas en lugares altos donde se tenga buena recepción de datos evitando la obstaculización.

Pruebas en zonas despejadas

Las pruebas realizadas en zonas despejadas de edificios muestran ser mejor en cuanto a la fuerza de señal recibida y transmitida a larga distancia, aunque la diferencia en distancia percibida no es superior a los 15 metros. Los resultados obtenidos se presentan en la tabla 58.

Tabla 58

Resultado pruebas de distancia en zonas despejadas

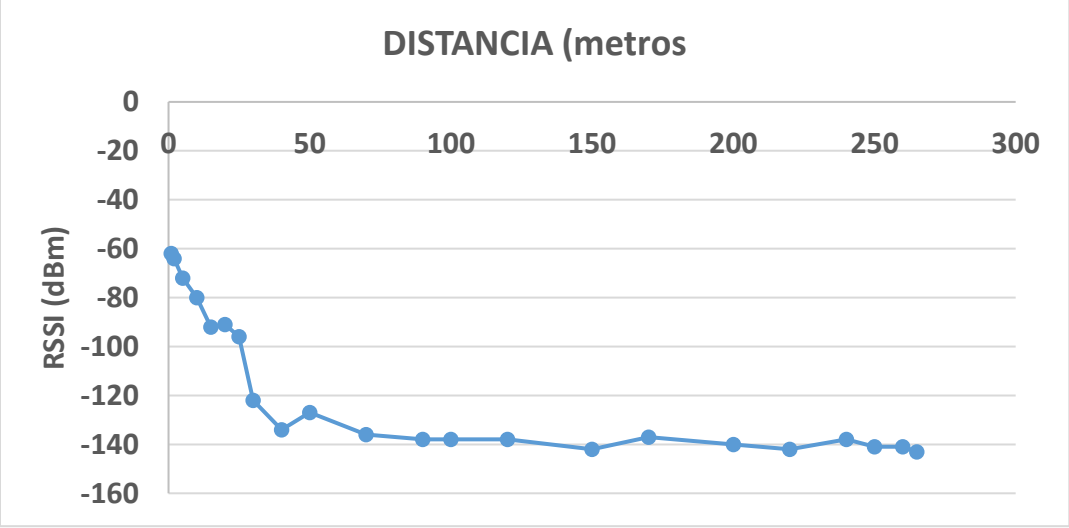
Distancia	RSSI dBm			RSSI Promedio
	Prueba 1	Prueba 2	Prueba 3	
1 m	-61 dBm	-63 dBm	-63 dBm	-62 dBm
2 m	-63 dBm	- 63 dBm	-65 dBm	-64 dBm
5 m	-70 dBm	-71 dBm	-74 dBm	-72 dBm
10 m	-79 dBm	-79 dBm	-81 dBm	-80 dBm
15 m	-90 dBm	- 95 dBm	-92 dBm	-92 dBm
20 m	-89 dBm	-90 dBm	-95 dBm	-91 dBm
25 m	-92 dBm	-98 dBm	-97 dBm	-96 dBm
30 m	-117 dBm	-128 dBm	-121 dBm	-122 dBm
40 m	-130 dBm	-137 dBm	-135 dBm	-134 dBm
50 m	-128 dBm	-125 dBm	-129 dBm	-127 dBm
70 m	-135 dBm	-133 dBm	-139 dBm	-136 dBm
90 m	-139 dBm	-141 dBm	-135 dBm	-138 dBm
100 m	-135 dBm	-137 dBm	-129 dBm	-138 dBm
120 m	-138 dBm	-135 dBm	-141 dBm	-138 dBm

Distancia	RSSI dBm			RSSI Promedio
	Prueba 1	Prueba 2	Prueba 3	
150 m	-143 dBm	-141 dBm	-143 dBm	-142 dBm
170 m	-133 dBm	-139 dBm	-139 dBm	-137 dBm
200 m	-141 dBm	-144 dBm	-135 dBm	-140 dBm
220 m	-144 dBm	-141 dBm	-140 dBm	-142 dBm
240 m	-141 dBm	-135 dBm	-138 dBm	-138 dBm
250 m	-144 dBm	-140 dBm	-139 dBm	-141 dBm
260 m	-144 dBm	-140 dBm	-142 dBm	-141 dBm
265 m	-144 dBm	-141 dBm	-144 dBm	-143 dBm

Nota: Se muestran los resultados de 3 pruebas de distancia realizadas en zonas despejadas.

Figura 123

Relación distancia y fuerza de señal en comunicación LoRa Arduino Heltec Stick Lite en zona despejada



Nota: Gráfica de los promedios de la fuerza de la señal de comunicación LoRa Arduino Heltec Stick Lite en zonas despejadas

Se tuvo resultados similares con el anterior caso de pruebas la única diferencia fue que se alcanzó una distancia de 265 metros antes de perder la señal de conexión entre los dispositivos Arduino LoRa Heltec Wireless Stick Lite, se aprecian los promedios de la fuerza de señal en la figura 123. Siempre desde un mismo punto de posición con tan solo movilizar el robot unos centímetros el RSSI variable de entre 5 a 10 dBm, en este caso los obstáculos para la transmisión de datos pueden verse obstaculizados por la presencia de vegetación.

Pruebas de integración

Como podemos observar en la figura 61 tenemos dos enlaces principales que concatenan el flujo de datos de la arquitectura. El primer enlace es el “main” desarrollado en Python, que se ejecuta en la Raspberry Pi, con el fin de enlazar los datos de transmisión y recepción entre el software y el hardware de la arquitectura, es decir realiza la lectura de los datos de los sensores, para posteriormente empaquetarlos y enviarlos por los medios de transmisión de nuestra arquitectura hasta enlazarla al software y mostrarlos en la HMI. Así mismo receptar las instrucciones enviadas desde la HMI, para de esta manera manipular los actuadores del Robot.

El segundo enlace es el programa realizado en Node Red con el objetivo de integrar el software de la arquitectura como son, microservicios, bases de datos, codificación y decodificación de datos de transmisión y recepción respectivamente.

Las pruebas de integración fueron realizadas con el fin de analizar que los múltiples programas desarrollados en diferentes plataformas y enlazados como se explica anteriormente, estén funcionando correctamente en conjunto, ya que se ha realizado la programación de forma escalable, es decir que cada elemento dentro de la arquitectura tenga su programación independiente, con el objetivo de que, en investigaciones futuras, la arquitectura pueda crecer tanto en hardware como software.

Para este fin se utilizó el programa honeycomb.io que tiene la facultad de depurar y mejorar los programas. También cuenta con la capacidad de brindar visualizaciones en tiempo real del sistema con lo que se pueden detectar problemas al mismo tiempo que se supervisa el proceso.

Para obtener los datos en la interfaz honeycomb.io se utilizó la extensión Libhoney para Python, que fue ejecutada dentro del “main”, con el fin de monitorear los datos de transmisión, recepción, latencias y errores generados durante el funcionamiento en condiciones normales de toda la arquitectura.

Tabla 59

Datos analizados por el programa honeycomb.io

Datos	Tx/Rx	Tipo	Descripción
Distancia	Tx	String	Medición del sensor ultrasónico
Latitud	Tx	String	Latitud otorgada por módulo GPS
Longitud	Tx	String	Longitud otorgada por módulo GPS
Recepción	Rx	String	Recepción de instrucción de HMI
Error	Tx	Bool	Excepciones ejecutadas del programa
Latencia	Tx	Int	Latencia en Rx y Tx

Nota: La presente tabla muestra a detalle los datos analizados en la prueba de integración por el programa honeycomb.io a través de la extensión Libhoney instalada en Python.

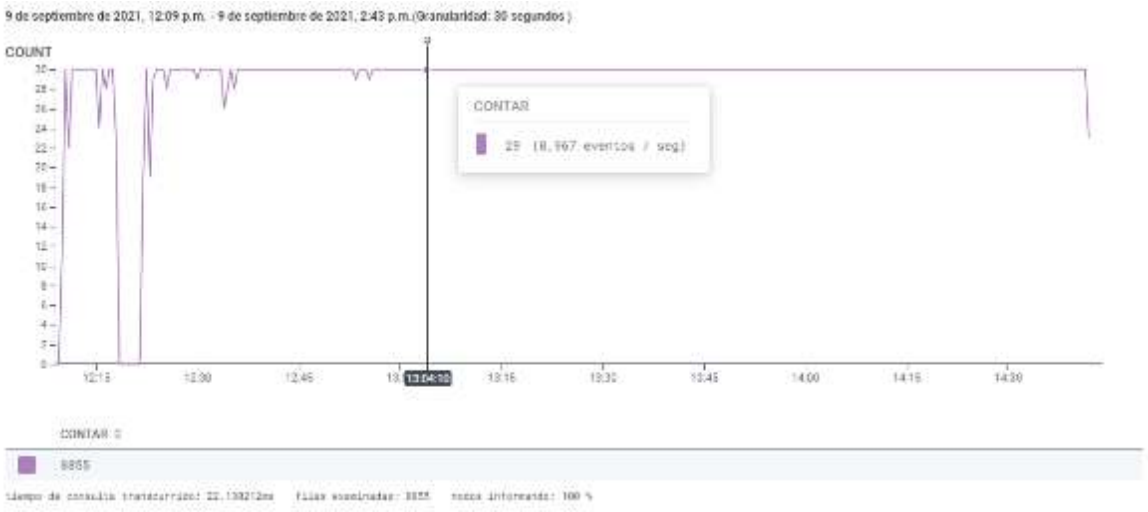
La prueba fue realizada dos veces, cada una consta de un tiempo de dos horas, que es el tiempo que dura la batería en condiciones normales de funcionamiento. Durante las pruebas se manipulo el robot y toda la arquitectura para de esta manera manejar un escenario real de funcionamiento y obtener un análisis acertado.

Prueba sin depuración

La primera prueba fue realizada en condiciones iniciales de funcionamiento, tratando de manipular el sistema al máximo para generar estrés y verificar las falencias en la depuración.

Figura 124

Prueba de integración en condiciones iniciales



Nota: Gráfica de flujo de datos del sistema para la verificación de eventos obtenida mediante el programa honeycomb.io

En la figura 124 podemos visualizar como la prueba empieza con regularidad, pero aproximadamente a las 12h20 se registra una caída del sistema, verificando un error que detuvo la transmisión y recepción de datos del sistema. Una vez depurado el programa se reinició la prueba, registrando un promedio de 30 eventos por segundo obteniendo mejores respuestas en la transmisión que en la recepción.

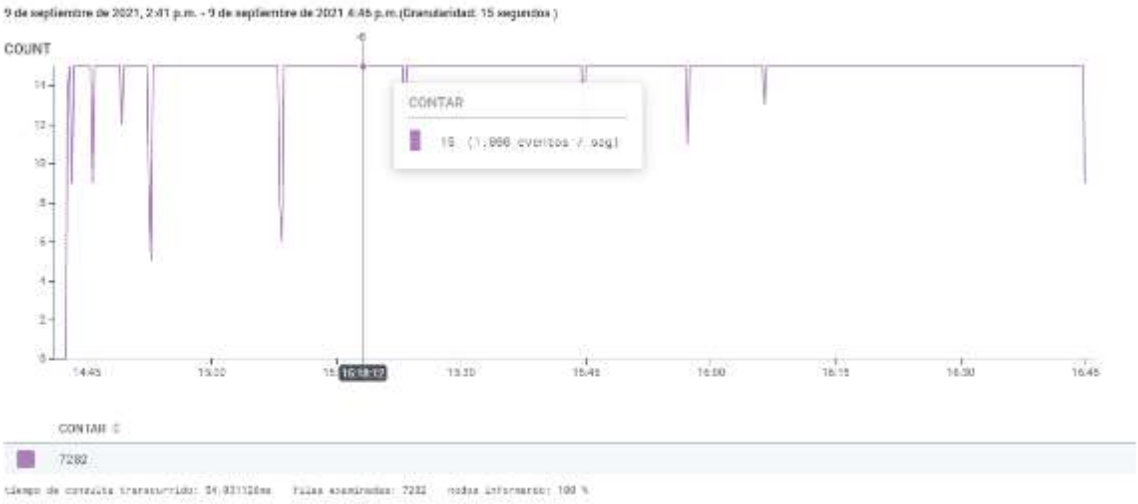
Prueba con depuración

Una vez realizada la prueba uno se procedió a depurar el sistema por completo, encontrando varias falencias que fueron corregidas mediante el manejo de excepciones que ayuden a mantener estable el sistema, también se organizó de mejor manera el

empaquetamiento de datos tanto para transmisión y recepción para de esta manera mejorar la calidad de comunicación y bajar el tiempo de latencia.

Figura 125

Prueba de integración con corrección de errores



Nota: Gráfica de flujo de datos del sistema para la verificación de eventos obtenida mediante el programa honeycomb.io después de una depuración completa en base a la primera prueba.

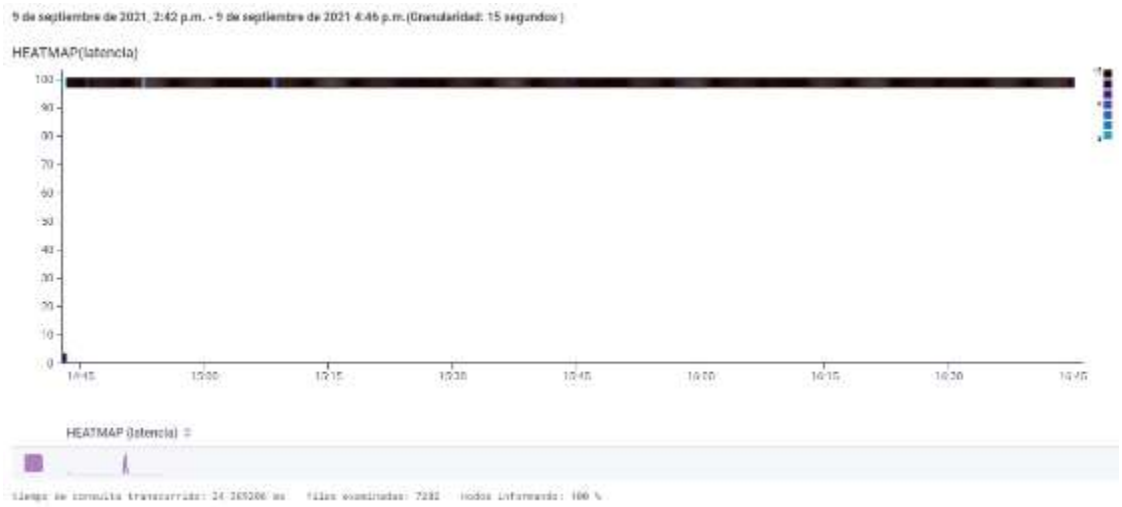
Una vez el sistema depurado podemos observar en la figura 125 que el sistema funciona de manera estable durante toda la prueba. Se realizó una manipulación del sistema en condiciones normales donde se trató de generar un alto flujo de eventos para obtener un mayor flujo de datos.

Se realizaron conexiones y desconexiones de sensores y actuadores para verificar que las excepciones funcionen correctamente. Por este motivo se puede observar picos descendentes que se traducen en el manejo de excepciones que por cortos lapsos de tiempo generan una disminución en la cantidad de eventos para que de esta manera el sistema logre reestablecerse y posteriormente pueda seguir funcionando con total normalidad.

El sistema logró un funcionamiento más óptimo, ya que el flujo de datos también fue afectado positivamente, ya que los tiempos de respuesta tanto en transmisión y recepción fueron semejantes y muy bajos, con un promedio de 100ms, como se puede ver en la figura 126.

Figura 126

Latencia del sistema



Nota: Gráfica de la medición de latencia de transmisión y recepción del sistema obtenida del programa honeycomb.io

Pruebas de carga y estrés de servicio Web

Las pruebas de estrés fueron aplicadas en el protocolo de comunicación MQTT que es lo más utilizado en el presente trabajo para envío y recepción de los datos en formato JSON, para dichas pruebas se utiliza la herramienta JMeter que tiene la capacidad de publicar una serie de peticiones por parte de una determinada cantidad de usuarios lo que permite obtener la información necesaria sobre el tiempo de respuesta y el porcentaje de error en caso de fallo. Se menciona que esta prueba se aplicó en la saturación de solicitudes para el envío de instrucciones hacia el manejo de las pinzas del robot esto con el objeto de realizar una evaluación de la arquitectura del sistema implementado.

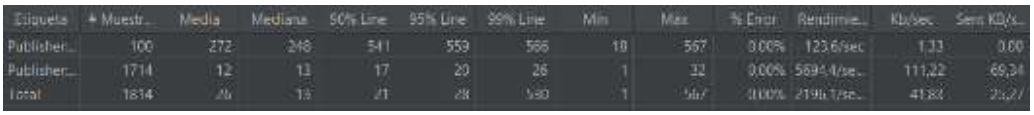
La herramienta JMeter fue configurada con un tiempo de ejecución de 30 segundos, con pruebas de 100, 200, 300, 400, 500, 1000, 5000 y 10000 usuarios en donde individualmente enviarán 10 peticiones.

Prueba 100 usuarios

En la herramienta JMeter se configuran los 100 usuarios junto con 1000 requerimientos durante 30 segundos con lo que se obtuvo los resultados presentados en la figura 128, se observa que existe un error del 0%, es decir que se atendieron todas las peticiones con una media del tiempo de respuesta de 809 ms.

Figura 127

Reporte de prueba de carga con 100 usuarios

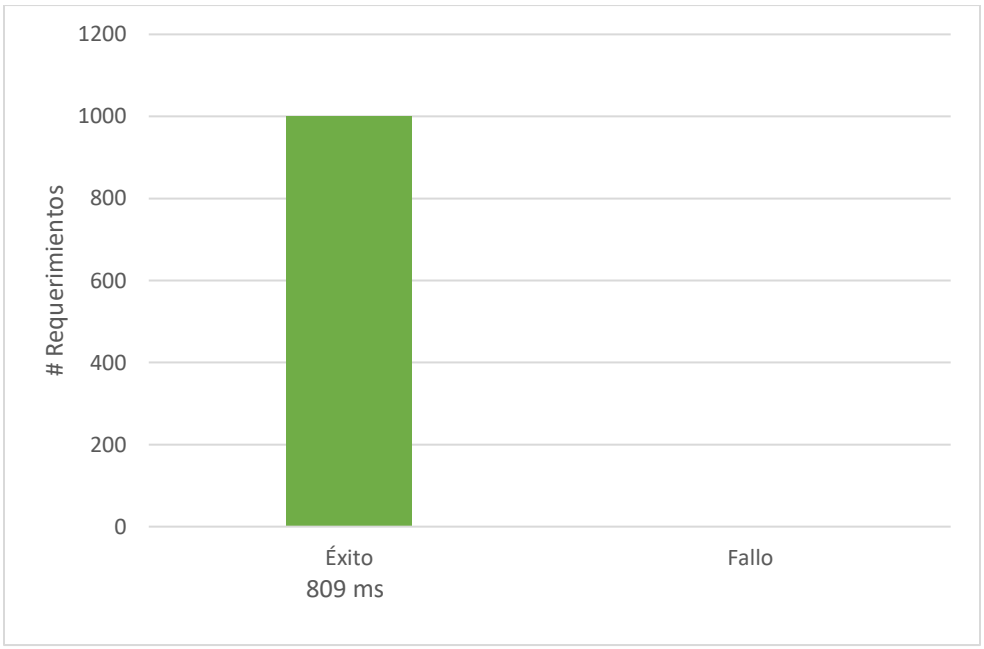


Etiqueta	# Muestr...	Media	Mediana	90% Line	95% Line	99% Line	Min	Max	% Error	Rendimie...	Kb/sec	Sent KB/s...
Publisher...	100	272	248	541	553	566	18	567	0.00%	123.6/sec	1.33	0.00
Publisher...	1714	12	13	17	20	26	1	32	0.00%	5694.4/se...	111,22	69,34
Total	1814	26	13	21	28	540	1	567	0.00%	2196.1/se...	41.83	25,27

Nota: Informe de resultados de prueba de 100 usuarios con la herramienta JMeter

Figura 128

Prueba de carga con 100 usuarios



Nota: Gráfica luego de someter al sistema ante una carga simultánea de 100 usuarios

Prueba 200 usuarios

En la herramienta JMeter se configuran los 200 usuarios junto con 2000 requerimientos durante 30 segundos con lo que se obtuvo los resultados presentados en la figura 130, se observa que existe un error del 0%, es decir que al igual que el caso anterior se atendieron todas las peticiones con una media del tiempo de respuesta de 2284 ms.

Figura 129

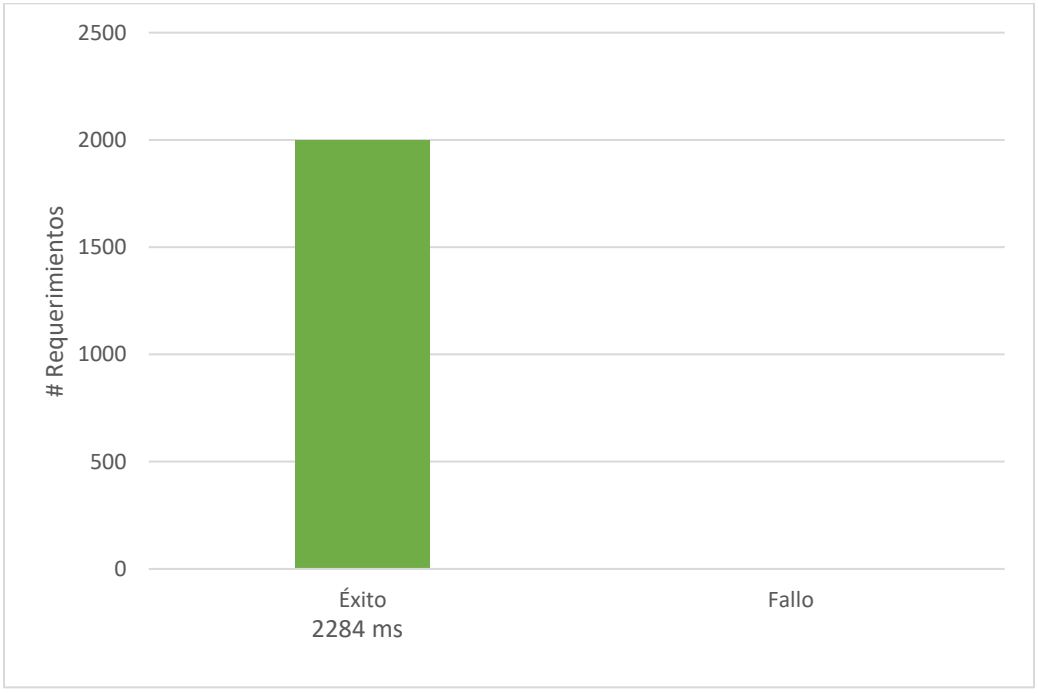
Reporte de prueba de carga con 200 usuarios

Etiqueta	# Muestr...	Media	Mediana	90% Line	95% Line	99% Line	Min	Max	% Error	Rendimie...	Kb/sec	Sent KB/s...
Publisher...	2000	582	568	919	1231	1539	4	1564	0,00%	875,3/sec	9,40	0,00
Publisher...	4270	56	51	112	130	174	1	212	0,00%	2373,5/se...	46,36	28,85
Total	6270	225	84	704	875	1482	1	1564	0,00%	2744,0/se...	45,90	22,72

Nota: Informe de resultados de prueba de 200 usuarios con la herramienta JMeter

Figura 130

Prueba de carga con 200 usuarios



Nota: Gráfica resultado luego de someter al sistema ante un estrés de carga simultánea de 200 usuarios

Prueba 300 usuarios

En la herramienta JMeter se configuran los 300 usuarios junto con 3000 requerimientos durante 30 segundos con lo que se obtuvo los resultados presentados en la figura 132, se observa que existe un error del 0%, es decir que al igual que el caso anterior se atendieron todas las peticiones con una media del tiempo de respuesta de 2760 ms.

Figura 131

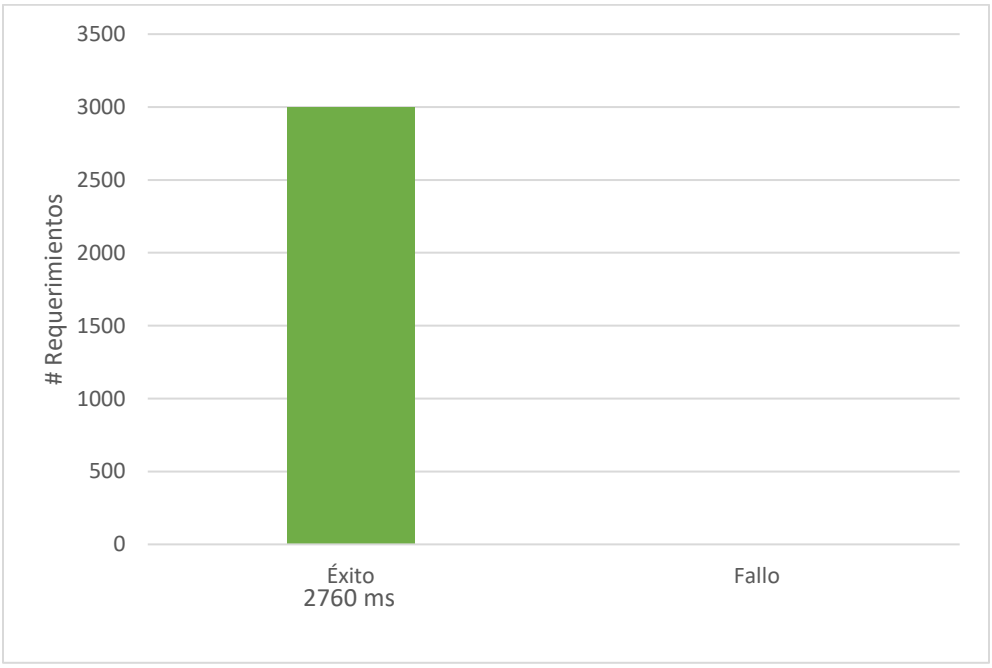
Reporte de prueba de carga con 300 usuarios

Etiqueta	# Muestr...	Media	Mediana	90% Line	95% Line	99% Line	Min	Max	% Error	Rendimie...	Kb/sec	Sent KB/s...
Publisher...	3000	1453	1570	2123	2293	2370	13	2440	0,00%	1087,0/se...	11,68	0,00
Publisher...	2596	107	98	208	215	223	7	259	0,00%	1272,5/se...	24,85	15,51
Total	5596	829	351	1909	2138	2328	7	2440	0,00%	2027,5/se...	30,05	11,46

Nota: Informe de resultados de prueba de 300 usuarios con la herramienta JMeter

Figura 132

Prueba de carga con 300 usuarios



Nota: Gráfica resultado luego de someter al sistema ante un estrés de carga simultánea de 300 usuarios

Prueba 400 usuarios

En la herramienta JMeter se configuran los 400 usuarios junto con 4000 requerimientos durante 30 segundos con lo que se obtuvo los resultados presentados en la figura 134, se observa que existe un error del 4.5%, en donde 180 peticiones tuvieron fallo y de forma exitosa se atendieron 11753 peticiones con una media del tiempo de respuesta de 2760 ms.

Figura 133

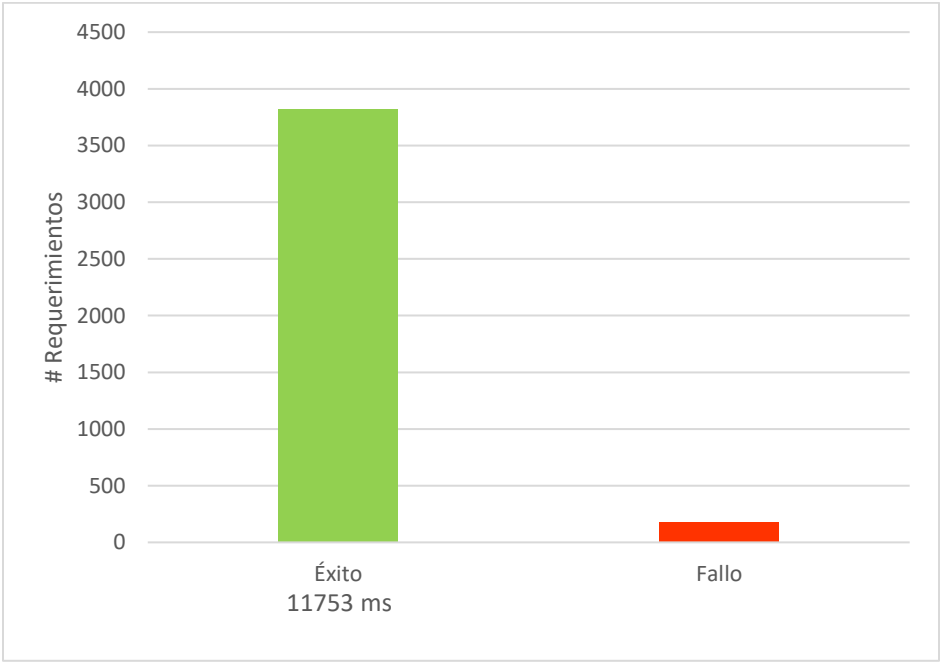
Reporte de prueba de carga con 400 usuarios

Etiqueta	# Muestr...	Media	Mediana	90% Line	95% Line	99% Line	Min	Max	% Error	Rendimie...	Kb/sec	Sent KB/s...
Publisher...	4000	1743	1786	2664	2815	3193	78	3421	4,50%	1021,7/se...	12,95	0,00
Publisher...	1690	134	128	241	273	291	6	301	0,00%	624,8/min	12,20	7,59
Total	5690	1265	1266	2265	2747	3189	6	3421	3,16%	1453,4/se...	21,38	5,25

Nota: Informe de resultados de prueba de 400 usuarios con la herramienta JMeter

Figura 134

Prueba de carga con 400 usuarios



Nota: Gráfica resultado luego de someter al sistema ante un estrés de carga simultánea de 400 usuarios.

Prueba 500 usuarios

En la herramienta JMeter se configuran los 500 usuarios junto con 5000 requerimientos durante 30 segundos con lo que se obtuvo los resultados presentados en la figura 136, se observa que existe un error del 10.6%, en donde 530 peticiones tuvieron fallo y de forma exitosa se atendieron 4470 peticiones con una media del tiempo de respuesta de 32786 ms.

Figura 135

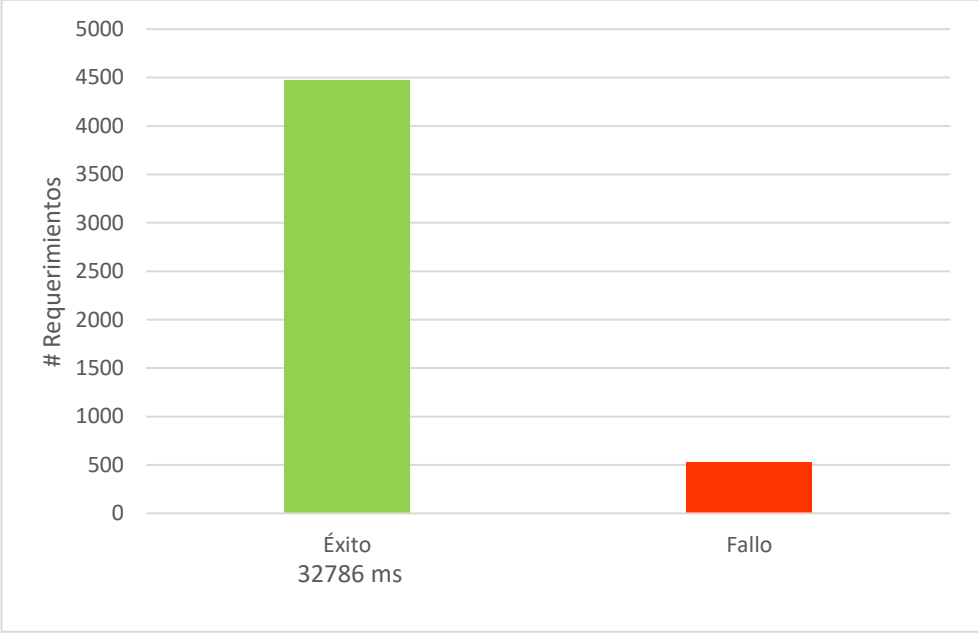
Reporte de prueba de carga con 500 usuarios

Etiqueta	# Muestr...	Media	Mediana	90% Line	95% Line	99% Line	Mín	Máx	% Error	Rendimie...	Kb/sec	Sent KB/s...
Publishec...	5000	2113	2068	3380	3549	3833	92	4097	10,60%	1075,7/se...	16,46	0,00
Publisher...	2083	155	166	208	219	520	15	538	0,00%	560,9/sec	10,95	6,82
Total	7083	1537	1424	3221	3456	3828	15	4097	7,48%	1523,9/se...	25,21	5,45

Nota: Informe de resultados de prueba de 500 usuarios con la herramienta JMeter

Figura 136

Prueba de carga con 500 usuarios



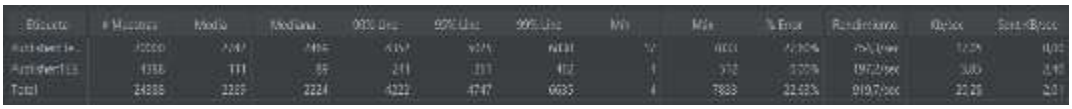
Nota: Gráfica resultado luego de someter al sistema ante un estrés de carga simultánea de 500 usuarios

Prueba 1000 usuarios

En la herramienta JMeter se configuran los 1000 usuarios junto con 10000 requerimientos durante 30 segundos con lo que se obtuvo los resultados presentados en la figura 138, se observa que existe un error del 27.6%, en donde 2760 peticiones tuvieron fallo y de forma exitosa se atendieron 7240 peticiones con una media del tiempo de respuesta de 39771 ms.

Figura 137

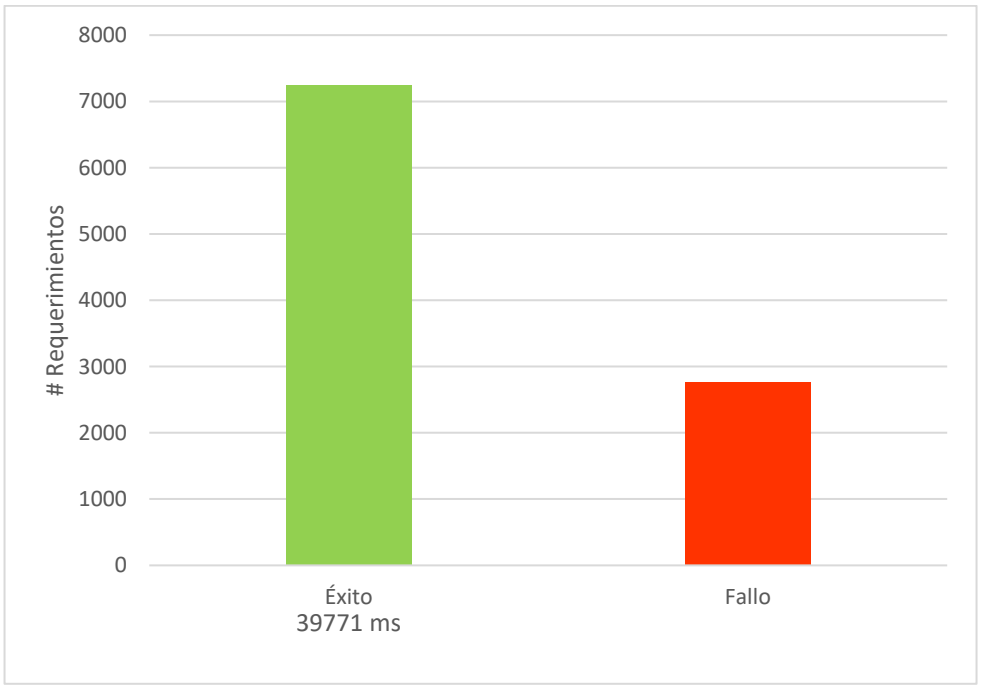
Reporte de prueba de carga con 1000 usuarios



Nota: Informe de resultados de prueba de 1000 usuarios con la herramienta JMeter

Figura 138

Prueba de carga con 1000 usuarios



Nota: Gráfica resultado luego de someter al sistema ante un estrés de carga simultánea de 1000 usuarios

Prueba 5000 usuarios

En la herramienta JMeter se configuran los 5000 usuarios junto con 50000 requerimientos durante 30 segundos con lo que se obtuvo los resultados presentados en la figura 140, se observa que existe un error del 100%, en donde las 50000 peticiones tuvieron fallo y no se obtuvo ninguna media del tiempo de respuesta.

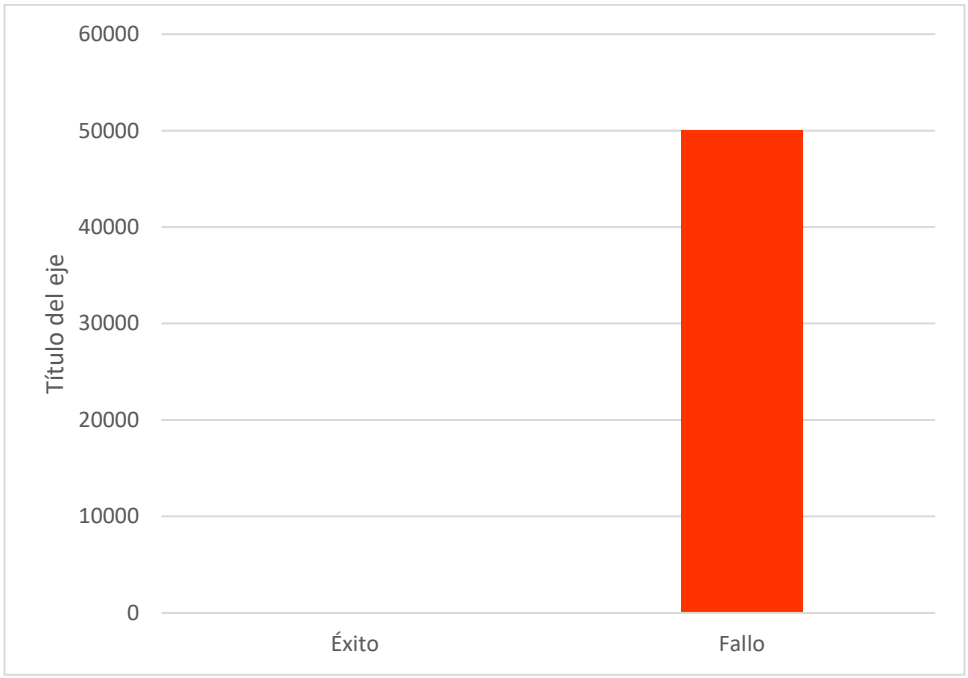
Figura 139

Reporte de prueba de carga con 5000 usuarios

Nota: Informe de resultados de prueba de 5000 usuarios con la herramienta JMeter

Figura 140

Prueba de carga con 5000 usuarios



Nota: Gráfica resultado luego de someter al sistema ante un estrés de carga simultánea de 5000 usuarios

Análisis pruebas totales de carga y estrés

Se da a conocer el resumen de los resultados registrados en la tabla 60 de lo cual se puede deducir que los datos dan a conocer el comportamiento del sistema expuesto ante diversos escenarios de estrés, pues en un comienzo hasta con 300 usuarios y 3000 peticiones no se encuentra error alguno y las publicaciones realizadas en el sistema por el protocolo MQTT son exitosas y atendidas totalmente, sin embargo vemos que desde los 400 usuarios con 4000 peticiones el rendimiento del sistema comienza a bajar pues ya comenzamos a presenciar pérdida de información, y si va subiendo se ve como el sistema va empeorando hasta llegar a un punto en el cual el error en la publicación de datos es del 100% pero ya con un elevado número de usuarios que son 5000 y 50000 peticiones.

Tabla 60.

Resultados pruebas de carga y estrés

# Usuarios	#Requerimientos	% Error	Tiempo de respuesta
100	1000	0%	809 ms
200	2000	0%	2284 ms
300	3000	0%	2760 ms
400	4000	4.5%	11753 ms
500	5000	10.6%	32786 ms
1000	10000	27.6%	39771 ms
5000	50000	100%	---

Nota: La presente tabla da a conocer todos los resultados obtenidos en las diferentes pruebas de carga y estrés para el análisis del comportamiento del sistema.

Pruebas de suministro de energía

Las pruebas realizadas en la presente sección fueron con el fin de obtener un promedio del tiempo de vida del robot, para esto se utilizó el arrancador de baterías Beatit B7 Pro QDSP, en la tabla 61 se registraron los datos del rango de porcentaje de batería, tiempo, y dos escenarios uno de uso moderado cuando el robot está conectado emitiendo datos y un escenario de uso excesivo cuando el robot está emitiendo datos y movilizándose continuamente.

Tabla 61.

Resultados pruebas del suministro energético

% Batería	**Tiempo de Uso (Minutos)**	
	Uso Moderado del Robot	**Uso excesivo del Robot**
100% - 90%	18 min	16 min
90% - 80%	18 min	16 min
80% - 70%	17 min	15 min
70% - 60%	16 min	13 min
60% - 50%	15 min	13 min
50% - 40%	15 min	12 min
40% - 30%	15 min	11 min
30% - 20%	16 min	10 min
20% - 10%	15 min	8 min
10% - 0%	15 min	6 min
Total	170 min	120 min

Nota: La presente tabla da a conocer todos los resultados de las pruebas de suministro de energía

Consecuentemente se observó que existe una diferencia de consumo energético cuando el robot está en reposo o movilizándose, sin embargo, el arrancador de batería al ser de alta capacidad tiene buena respuesta para poder energizar una Rasberry Pi 4, la placa madre del robot junto con todos los sensores y actuadores que están involucrados en el robot. Con un uso moderado el tiempo de vida de la batería es de 2 horas con 50 minutos y con un uso excesivo del robot es decir en constante movimiento la duración de la batería es de 2 horas.

Capítulo 6: Conclusiones y Recomendaciones

Conclusiones

Mediante una exhaustiva investigación se diseñó e implementó un sistema constituido de diversas tecnologías para desarrollo de un aplicativo en donde se tuvo una parte hardware conformada por un robot móvil además de las tecnologías de acceso para poder establecer comunicación con la parte de software en donde se realizó una orquestación para el uso de varios microservicios como video streaming, reconocimiento de imágenes, geolocalización y tratamiento de datos de sensores e instrucciones basado en el Cloud Computing a fin de estructurar toda una arquitectura Cloud IoT.

Se acopló el uso de tecnologías de comunicación a larga distancia con el protocolo LoRa para establecer conectividad entre el medio físico y el virtual en donde la información se transmitía y receptaba en formato JSON por su alta adaptabilidad en las plataformas y principalmente empleado por los servicios de mensajería para utilizar la información con los servicios Cloud.

Se establecieron parámetros de diseño escalables implementando una programación de software modular y el empleo de funciones anidadas en cada uno de los programas independientes correspondiente a cada elemento de la arquitectura, al igual que en el hardware donde se dejaron provistas futuras conexiones con el fin de que la arquitectura tenga la posibilidad de aceptar más dispositivos de campo en la capa de Edge Computing y proporcionarle la capacidad de establecer funciones futuras de robótica colaborativa.

Se implementó una arquitectura Cloud IoT estructurada en capas teniendo primero al robot móvil equipado con un controlador Rasberry Pi 4, sensores para detección de obstáculos y geolocalización así como los diferentes actuadores para el desplazamiento del robot y la movilidad del brazo robótico, luego se tiene una segunda capa incorporada

con la tecnología LoRa para realizar una transmisión y recepción de información a larga distancia hasta un Gateway el cual utiliza la tecnología de acceso WiFi y protocolos MQTT, cabe destacar que desde la primera capa el formato para el envío de los datos es en JSON, para finalizar esta la tercera capa en la cual se tiene todos los servicios REST usando principalmente Amazon Web Services, Web Map y MQTT que son orquestados con las herramientas de Node-RED y NodeJS, a fin de presentar toda la información en una interfaz y además recibir las instrucciones del usuario para el robot que se enviarían a la primera capa pues todo funciona de forma bidireccional.

Se pudo identificar un correcto dimensionamiento de los elementos que conforman el hardware de la arquitectura ya que, en la parte mecánica adaptada a la aplicación, se observó un aceptable desenvolvimiento de los componentes en las funciones de desplazamiento y cambios de dirección, así como también de la manipulación de objetos por parte del brazo robótico. Por otra parte, los elementos electrónicos cumplieron correctamente con sus funciones, en el caso de los sensores proporcionando los datos correctos y adecuados según las pruebas realizadas, así mismo los actuadores funcionando correctamente a la recepción de instrucciones desde la parte tele-operativa donde por último se evidenció un correcto dimensionamiento energético ya que el suministro de la batería pudo brindar una autonomía superior a dos horas que en aplicaciones de exploración a distancia es aceptable.

Se pudo desarrollar pruebas de varios ámbitos, destacando las pruebas de usabilidad donde se evidenció un alta aceptabilidad por parte de los usuarios a la arquitectura implementada, pues se identificó en las encuestas lo innovador e interactivo que consideraban el sistema, así también las pruebas de reconocimiento de imágenes donde se comprobó una alta eficiencia del servicio utilizado, las pruebas de integración para identificar y corregir falencias sobre la marcha, y de estrés identificando los puntos

débiles de la arquitectura que deben ser tomados en cuenta pero que generalmente no afectan al correcto funcionamiento del sistema.

Recomendaciones

En el caso del robot móvil y la raspberry que tiene la disponibilidad de levantar servidores locales se recomienda usar algún software independiente que ayude a crear túneles para acceder a la ruta HTTP de forma remota.

Cuando se requieran elementos de hardware que no necesitan un alto nivel de personalización especialmente en componentes mecánicos, se pueden adaptar kits prefabricados como el que se usó en esta aplicación, que hoy en día tienen una gran demanda en el mercado.

Existen servidores de paga que no suelen ser demasiado costosos con características básicas que puedan ayudar para adecuar la arquitectura y de esta manera el aplicativo pueda funcionar desde cualquier parte sin inconvenientes.

Aplicar una metodología de investigación que pueda ayudar a recolectar la información en base a diferentes pasos que se guíen bajo los requerimientos para el desarrollo del trabajo, de esta manera se simplifica la búsqueda y se crea un repositorio propio que sea adecuado a las temáticas tratadas además de completo para que no se creen inquietudes por parte del investigador.

Sustentar el proceso de diseño en alguna normativa de diseño existente, ya que estas fueron probadas y perfeccionadas durante años de estudio, para de esa manera poder garantizar un correcto funcionamiento y una alta calidad del producto después de su implementación.

Realizar un correcto dimensionamiento de los elementos electrónicos, especialmente de los módulos de comunicación que deben ser infalibles, ya que en el mercado existe una gran variedad, pero anticipadamente a su elección se deben revisar

las hojas técnicas y las garantías que brindan, ya que de la buena calidad de estos dependerá el correcto funcionamiento de la arquitectura.

Trabajos Futuros

Para los trabajos futuros se propone:

Desarrollo de una investigación de los módulos LoRa disponibles en el mercado y determinar el que tenga mayor rango de comunicación a distancia para aplicaciones de tele-operación con rápidos tiempos de respuesta en la transmisión y recepción de información.

Diseño e implementación de una arquitectura Cloud IoT para drones, dotada de tele-operación basada en LoRaWAN, para determinar la eficiencia de esta tecnología en aplicaciones aéreas.

Implementación de un robot, conectado a microservicios de exploración para dotarlo de autonomía en la tele-operación y funcionar únicamente para supervisión.

Diseño e implementación de una arquitectura Cloud IoT para el manejo de varios robots con tareas específicas controlados a larga distancia capacitados con algoritmos de machine learning.

Acrónimos

- SMS. Systematic Mapping Study.
- SLR. Systematic Literature Review
- AWS. Amazon Web Services.
- IoT. Internet of Things.
- HTTP. Hipertex Transfer Protocol.
- REST. Representational State Transfer.
- MQTT. Message Queuing Telemetry Transport.
- LoRa. Long Range.

Referencias

Adeept. (2021). *Adeept*. Obtenido de https://www.adeept.com/adeept-picar-pro-smart-robot-car-kit-2-in-1-4wd-car-robot-with-4-dof-robotic-arm-for-raspberry-pi_p0246_s0041.html

Adelantado, F., Vilajosana, X., Tuset-Peiro, P., Martinez, B., Melia-Segui, J., & Watteyne, T. (2017). *Understanding the limits of LoRaWAN*. Obtenido de IEEE Communications Magazine: 10.1109/mcom.2017.1600613

Aimacaña, J. P., & Columba, A. R. (2021). *Análisis comparativo de algoritmos de Machine Learning para la detección de plagas en los cultivos representativos de la sierra ecuatoriana.* Universidad Central del Ecuador.

Alava Bravo, M. J. (2019). Desarrollo de una arquitectura Cross-Device para Cloud Computing aplicada a un sistema EHEALTH. *Carrera de Ingeniería en Electrónica y Telecomunicaiones. Universidad de las Fuerzas Armadas ESPE*.

Arias, J. (2020). *Aplicaciones Web.* Madrid: Universidad Carlos III de Madrid.

AWS. (2020). *Amazon Web Services*. Obtenido de https://aws.amazon.com/es/

Banijamali, A., Heisig, P., Kristan, J., Kuvaja, P., & Oivo, M. (2019). *Software Architecture Design of Cloud Platforms in Automotive Domoin: An Online Survey*. Obtenido de 2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA): 10.1109/soca.2019.00032

Barrientos, V. R., García, J., & Silva, R. (2017). *Robots Móviles, Evolucion y Estado del Arte.* Distrito Federal, Mexico: Polibits.

Bengio, Y., Goodfellow, I., & Courville, A. (2015). *A Deep Learning Book in preparation for MIT.* USA.

Bertoleti, P. (2019). *Proyectos con ESP32 y LoRa.* Sao Paulo, Brasil: Instituto Newton C Braga.

BIOLOID. (2021). *RO-BOTICA*. Obtenido de RO-BOTICA: https://ro-botica.com/tienda/Controladores/

Brock, D., Bruce, R., & Cameron, M. (2013). *CHANGING THE WORLD WITH A RASPBERRY PI .* North Carolina: Asheville.

Burges, C. (1998). *A Tutorial on Support Vetor Machines for Pattern.* Boston: Kluwer Academic Publishes.

Casanova, R. A. (2021). *Procesamiento de algoritmos de visión artificial en la nube - Amazon AWS. Biblioteca Digital Universidad de Alcalá*. Obtenido de http://hdl.handle.net/10017/49068

Challenger, I., Días, Y., & Becerra, R. (2014). *El lenguaje de programación Python/The programming language Python.* Santiago de Cuba: Ciencias Holguin .

Chriboga Torres, A. (2020). Diseño e implementación de una solución con tecnología LORA para el monitoreo de ubicación vehicular con un aplicativo web . *Carrera de Ingeniería en Electrónica y Telecomunicaciones. Universidad de las Fuerzas Armadas ESPE*.

Cierco, D. (2011). *Cloud Computing, Retos y Oportunidades.* Madrid: Fundacion Ideas.

Córdova Lara, F. E., & Medina Torres, S. A. (2018). Diseño y construcción del prototipo de un robot móvil para telepresencia controlao a través de Internet. *Carrera de Ingeniería en Mecatrónica. Universidad de las Fuerzas Armadas ESPE*.

Corona, L., Abarca, G., & Carreño, J. (2014). *Sensores y Actuadores, Aplicaciones con arduino.* Mexico: Grupo editorial Patria.

Corral Plaza, D., Resinas, M., & Boubeta-Puig, J. (2018). Un recorrido por los principales proveedores de servicios de Mahine Learning y predicción en la nube. *Departamento de Ingeniería Informática, Universidad de Cádiz, España*.

Darabkh, K., & Muqat, R. (2018). *An Efficient Protocol for Minimizing Long-distancia Communications over Wireless Sensor Networks*. Obtenido de 2018 15th International Multi-Conference on Systems, Signals & Devices (SSD): 10.1109/ssd.2018.8570659

Deng, D. (2020). *Imagen Recogniton Algorithm Based on Information Fusio Combining Sparsity and Synergy*. Obtenido de 2020 2nd International Conference on Information Technology and Computer Application (ITCA): 10.1109/itca52113.2020.00042

Diaz, J. F. (Marzo de 2020). *Reconocimiento facial: FaceNet y AWS rekognition. Repositorio Institucional UCA*. Obtenido de https://repositorio.uca.edu.ar/handle/123456789/11212

Eguíluz, J. (2009). *Introducción a JavaScript.* Creative Commons.

Figueroa Chiriboga , J. A., & Tiuma Chafla, C. A. (2019). Diseño y construcción de un prototipo de robot todo terreno, tele-operado y dotado de vision remota para el Centro de Investigación Científica y Tegnológica del Ejército CICTE. . *Carrera de Ingeniería en Mecatrónica. Universidad de las Fuerzas Armadas ESPE*.

Finstad , K. (2006). *The System Usability Scale and Non-Native English Speakers.* New Mexico: JUS.

Galvis, J., & Madrid, J. (2016). *Sistema de control difuso para motor de corriente continua sin escobillas (BLDC) sobre hardware embebido.* Caldas: Tekhne.

Gamme, E., Immich, R., & Bittencourt, L. (2018). *Towards a Multi-tier Fog/Cloud Architecture for Video Streaming*. Obtenido de 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companision (UCC Companion): 10.1109/ucc-companion.2018.00022

Gardaseive, G. (2017). The IoT Architectural Framework, Desing Issues and Application Domains. En *Wireless Pers Common* (págs. 127-148).

Geekstory. (2021). *Amazon*. Obtenido de https://www.amazon.com/-/es/Geekstory-navegaci%C3%B3n-compatible-microcontrolador-Raspberry/dp/B07PRGBLX7

Gehadi, A., Shatagopam, S., Raghav, R., Sarkar, M., & Paolini, C. (2021). *Appliation of 915 MHz Band LoRa for Agro-Informatics*. Obtenido de 2021 Wireless Telecommunication Symposium (WTS): 10.1109/wts51064.2021.9433712

Gonzales , D. (2020). *Yunbit.* Obtenido de Yunbit: https://www.yunbitsoftware.com/blog/2016/09/02/importancia-de-las-pruebas-en-desarrollo-de-software/

Gonzales Bonifaz, D., & Verdugo Cabrera, A. (2018). Diseño e implementación de una arquitectura IoT para robótica colaborativa. *Carrera de Ingeniería en Mecatrónica. Universidad de las Fuerzas Armadas ESPE*.

González García, C. (2018). En qué consiste el aprendizaje automático (machine learning) y qué está aportando a la Neurociencia Cognitiva. *Ciencia Cognitiva*.

HELTEC. (2021). *HELTEC*. Obtenido de https://heltec.org/product/wireless-stick-lite/

Hernandez, D., Peralta, G., Manero, L., Gomez, R., Bilbao, J., & Zubia, C. (2017). *Energy and coverage study of LPWAN schemes for Industry 4.0*. Obtenido de IEEE

International Workshop of Electronics, Control, Measrement, Signals and their applications to mechatronics: 10.1109/ecmsm.2017.7945893

Hinestroza, D. (2018). El Machine Learning a través de los tiempos y los aportes a la humanidad. *Universidad Libre Seccional Pereira*.

Honeycomb.io. (2016). *Capterra*. Obtenido de Capterra: https://www.capterra.ec/software/198473/honeycomb#about

Hossain, M., Khan, R., Noor, S., & Hasan, R. (2016). *Jugo: A Generic Architecture for Composite Cloud as a Service*. Obtenido de 2016 IEEE 9th International Conference on Cloud Computing (CLOUD): 10.1109/loud.2016.0112

Hou, L., Zhao, S., Xiong, X., & Zheng, K. (2016). *Internet of Things Cloud: Architecture and.* Bejing: IEEE.

Hu, P., & Chen, W. (2019). *Software Defined Edge Computing (SDEC): Principles, Open Systems Architecture and Challenges.* Obtenido de 2019 IEEE SmartWorld Ubiquitous Intelligence & Computing, Advance & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation: 10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00047

Hunkeler, U., Truong, H. L., & Stanford-Clark, A. (2008). *MQTT-S - A Publish/Suscribe Protocol for Wireless sensor networks.* Obtenido de International Conference on Communication Systems Software and Middleware and Workshop: 10.1109/COMSWA.2008.4554519

Jansch, J., & Birkhofer, H. (2006). *THE DEVELOPMENT OF THE GUIDELINE VDI 2221 - THE CHANGE OF DIRECTION.* Dubrovnik, Croacia: DESIGN .

JMeter. (2021). *The Apache Software Fundation*. Obtenido de The Apache Software Fundation: https://jmeter.apache.org/

Kim, W., Jang, H., Choi, G., Hwang, S., & Youn, C. (2016). *A WebRTC based Live Sreaming Service Platform with Dynamic Resource Provisioning in Cloud*. Obtenido de 2016 IEEE Region 10 Conference (TENCON): 10.1109/tencon.2016.78484466

Kitchenhan, B., & Charters, S. (2004). *Guidelines for performing systematic literature reviews in software engineering: EBSE technical report EBSE-2007-01.* Durham: Keele University; Durham University Joint Report.

Komoda, N. (8 de January de 2008). *Service Oriented Architecture (SOA) in Industrial Systems*. Obtenido de 10.1109/INDIN.2006.275708

Larrucea, X., Santamaria, I., Colombo-Palacios, R., & Ebert, C. (4 de Mayo de 2018). *Microservices*. Obtenido de 10.1109/MS.2018.2141030

Laskey, K., & Laskey, K. (13 de July de 2009). *Service oreinted architecture*. Obtenido de Wires Computational Stadistics: https://doi.org/10.1002/wics.8

Leivi, A. E. (2019). Análisis de la implementación de Machine Learning en el diagnóstico de imágenes. *Universidad de San Andrés*.

Li, L., & Wu, C. (4 de July de 2011). *Desing and Describe REST API without Violating REST: A Petri Net Based Approach*. Obtenido de 10.1109/ICWS.2011.54

Liu, J., Hao, S., Zhang, X., Wang, C., Sun, J., Yu, H., & Li, Z. (2016). *Research on Web Service Dynamic Coposition Based on Execution Dependency Relationship.* Obtenido de 2016 IEEE World Congress on Service (SERVICES): 10.1109/services.2016.24

Markham, S. (2013). *The impact of front-end innovation activities on product performance .* Obtenido de Product Development & Management Association: 10.1111/jpim.12065

Mazzara, M., & Govoni, S. (2008). *A Case Study of Web Services Orchestration.* Obtenido de Coordination Models and Languajes: https://doi.org/10.1007/11417019_1

Mechatronics, N. (2021). *SENSOR ULTRASONIDO HC-SR04*. Obtenido de https://naylampmechatronics.com/sensores-proximidad/10-sensor-ultrasonido-hc-sr04.html

Memon, A., Banerjee, I., Nguyen, B., & Robbins, B. (2015). *The First Decade of GUI Ripping: Extensions, Applications, and Broader Impacts.* USA: University of Maryland.

Merchan, J. (2020). *Analisis de seguridad en plataformas de computacion distribuida con arquitectura de Edge Computing .* Guayaquil: Universidad de Guayaquil.

Miaibe, N. (2018). *Competing in the Age of Artificial Intelligence: The State of the Art of AI & Interpretation of Complex Data*. Obtenido de Focus (SCOR Global P&C).

MINTEL. (2019). Cloud Computing. En *Libro Blanco de Territorios Digitales en Ecuador* (pág. 168). Quito.

MINTEL. (2019). Edge Computing. En *Libro Blanco de Terrirtorios Digitales en Ecuador* (pág. 166). Quito.

MINTEL. (2019). Industria Manufacturera. En *Libro Blanco de Territorios Digitales en Ecuador* (págs. 93-94). Quito.

MINTEL. (2019). Inteligencia Artificial y Machine Learning. En *Libro Blanco de Territorios Digitales en Ecuador* (págs. 159-160). Quito.

Moguel Márquez, J. E. (2018). *Una arquitectura orientada a servicios y dirigda por eventos para el control inteligente de UAVs multipropósito.* Badajoz: Universidad de Extremadura.

Molina, K. (2021). Diseño e implementación de un sistema de robótica colaborativa basado en el robot KUKA KR3 R540 y controlador KUKA KRC4 compact. *Carrera de Ingeniería en Electrónica, Automatización y Control. Universidad de las Fuerzas Armadas ESPE*.

Navarrete, T. (2008). *El lenguaje de JavaScript.* Academia Accelerating the world's recearch.

Navarro, C., & Ramírez, M. (2018). *Mapeo sistemático de la literatura sobre evaluación docente*. Obtenido de Tecnológico de Monterrey: http://dx.doi.org/10.1590/S1678-4634201844185677

Negrete Fernández, A. O. (Mayo de 2021). *Redes Neuronales*. Obtenido de http://ofeliayorquesta.com/articulos/Redes_Neuronales_001.pdf

Pandey, S. (2019). *Clarion Technologies*. Obtenido de Clarion Technologies: https://www.clariontech.com/blog/cloud-computing-architecture-what-is-front-end-and-back-end

Panizzi, M. (2019). *Establecimiento del estado del arte sobre la Minería de Datos Educacional en el Nivel Superior: Un Estudio de Mapeo Sistemático.* Buenos Aires: Universidad de Morón.

Pecka, M., Zimmermann, K., Reinstein, M., & Svoboda, T. (2016). *Controlling Robot Morphology from Incomplete Measurements*. Obtenido de IEEE Transactions on Industrial Electronics, 64(2), 1773-1782: 10.1109/tie.2016.2580125

Peltz, C. (2003). *Web Services Orchestration and Choreography.* Obtenido de 10.1109/MC.2003.1236471

Pi, R. (2021). *Raspberry*. Obtenido de Raspberry: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/

Raspberry. (2019). *Amazon* . Obtenido de Amazon: https://www.amazon.com/-/es/Raspberry-Modelo-2019-Quad-Bluetooth/dp/B07TC2BK1X

Ray, S. (2019). *A Quick Review of Machine Learning Algorithms*. Obtenido de 2019 International Conference on Machine Learning, Big Data, Cloud and ParaLLeL Computing (COMITCon): 10.1109/comitcon.2019.8862451

Rebolledo, C. A., & Guerra, A. S. (2015). *Análisis de funcionamiento de servicios streaming.* Valparaíso: Universidad Técnica Federico Santa María.

Restrepo Jaramillo, M., & Restrepo García, N. (2014). *Materiales y equipos para las Tecnologías de Acceso.* Medellín: Institución Universitaria de Envigado.

Rodriguez, R. (2020). *Análisis de las tecnologías de redes inalámbricas WIMAX para centros poblados rurales en el departamento de Huanuco.* Pimentel: Universidad Señor de Sipán.

Rojas, E. M. (2020). Machine Learning: análisis de lenguajes de programación y herramientas para desarrollo. *Revista Ibérica de Sistemas de Información*, 586-599.

Sadeq, A., Hassan, R., Al-rawi, S., Jubair, A., & Aman, A. (2019). *A QoS Approach for Internet of Things (IoT) Enviroments using MQTT Protocol*. Obtenido de 2019 International Conference on Cybersecurity (ICoCSec): 10.1109/icocsec47621.2019.8971097

Sanchez Samaniego, F. (2014). *Diseño y contruccion de un robot para inspeccion visual de tuberia operado remotamente para la empresa FSB recubrimientos Industriales.* Sangolqui: Repositorio ESPE.

Sandoval, L. J. (15 de Octubre de 2018). *Algoritmos de aprendizaje automático para análisis y predicción de datos*. Obtenido de Consorcio de Bilbiotecas Universitaria de El Salvador: http://hdl.handle.net/10972/3626

Santos, J., Gilmore, A., Hempel, M., & Sharif, H. (2017). *Behavior-Based Robotics Programming for a Mobile Robotics ECE Course Using the CEENBoT Mobile Robotics Platform*. Obtenido de 2017 IEEE International Conference on Electro Information Technology (EIT) : 10.1109/EIT.2017.8053431

Settles, B. (2009). *Active Learning Literatura Survey.* Univesity of Wisconsin-Madison.

Singh, M., Rajan, M., Shivraj, V., & Balamuralidhar, P. (6 de April de 2015). *Secure MQTT for Internet of Things (IoT).* Obtenido de Fifth International Conference on Comunication Systems and Network Technologies: 10.1109/CSNT.2015.16

SMIT, J. (2016). *Industria 4.0*. Obtenido de Dictorate General for Internal Policies. European Parliament.

Song, Z., & Tilevich, E. (2019). *Euivalence-Enhanced Microservice Worflow Orchestation to Efficiently Increase Reliability*. Obtenido de 2019 IEEE International Conference on Web Services (ICWS): 10.1109/icws.2019.00076

Suárez, F. J. (2011). *Tecnologías Streaming.* Oviedo: Universidad de Oviedo - Area de Arquitectura y Tecnología de Computadores.

Surwase, V. (2016). REST API Modeling Languages - A Developer's. *IJSTE - International Journal of Science Technology & Engineering Volume 2*. Obtenido de REST API Modeling Languages - A Developer's Perspective.

T3ALLIANCE. (10 de April de 2019). *Streaming RPi Camera* . Obtenido de https://t3alliance.org/lessons/rpi-node-red-streaming-rpi-camera-to-dashboard/

Tao, Y., Li, T., Liu, S., Deng, T., & Xin, D. (2018). *Research of Universal Modular Cooperation Robot Control System 2018 2nd International Conference on Robotic and Automation Sciences (ICRAS)*. Obtenido de 10.1109/icras.2018.8443206

Teodovich Sosa, L. J., & Carelli, R. (2008). *Control Hibrido para Posicionamiento de un Robot tipo Ackerman.* San Juan, Argentina: Universidad Nacional de San Juan.

The Royal Society. (2017). Obtenido de Machine Learning: the power ad promise of computers that learn by example. Report by the Royal Society (Vol. 66): https://doi.org/10.1126/scitranslmed.3002564

Torrente Artero, O. (2013). *ARDUINO, Curso práctico de formación.* España: RC Libros.

Villalonga, A. (2016). *Sistemas Ciberfisicos: Principales estrategias de control y sus aplicaciones en la mecánica.* Cuba: Universidad de Matanzas - Sede "Camilo Cienfuegos".

Wang, J., Xu, W., & Wang, J. (2016). *A Study of Live Video Streaming System for Mobile Devices*. Obtenido de 2016 IEEE International Conference on Computer Communication and the Internet (ICCCI): 10.1109/cci.2016.7778898

Xiao, Z., Wijegunarathe, I., & Qiang, X. (20 de Marzo de 2017). *IEEE Xplore*. Obtenido de Reflections on SOA and Microservices: 10.1109/ES.2016.14

Yue, M., Ruiyang, Y., Jianwei, S., & Kaifeng, Y. (2017). *A MQTT Protocol Message Push Server Based on RocketMQ*. Obtenido de 2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA): 10.1109/icicta.2017.72

Anexos