

UNIVERSIDAD DE GUADALAJARA
CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS
DIVISIÓN DE ELECTRÓNICA Y COMPUTACIÓN
DEPARTAMENTO DE COMUNICACIONES Y ELECTRÓNICA
INGENIERÍA EN COMUNICACIONES Y ELECTRÓNICA
PROYECTO MODULAR
Vehículo Submarino ECO VAPO



Aldo Alexandro Vargas Meza
Luis Antonio Arreguin Sandoval
Ricardo Cenit Maldonado Ortiz

Guadalajara, Jalisco, 22 de mayo de 2018

ÍNDICE

INTRODUCCIÓN

Existe un gran interés en el desarrollo de robótica submarina en el mundo, tanto desde el punto de vista de la investigación como en el de la industria. Cada vez se hace más común el uso de robots acuáticos. Los Vehículos submarinos Operados Remotamente, ROVs, por sus siglas en inglés (Remotely Operated Vehicle); son robots que navegan bajo el agua a diferentes profundidades, debido a su propósitos, alcances y limitaciones que envían imágenes de video hacia una estación de control en la superficie, muchas veces poseen sensores para monitoreo y manipuladores para tareas específicas.

En la mayoría de las aplicaciones de los ROVs, el vehículo es utilizado para explorar toda un área de interés hasta que el operario encuentre su objetivo; luego, el operario busca que el vehículo permanezca enfocando un objetivo mientras el mismo móvil, u otro equipo, realiza una tarea en particular como la toma de muestras, el manipuleo de herramientas o cualquier otra, dependiendo de la aplicación predefinida.

El potencial de análisis con el que cuentan los ROVs abarca distintas áreas de oportunidad, en las que es necesario o de ayuda el uso de herramientas que puedan aumentar la seguridad del operario al recabar información al respecto. Como ejemplo de ello se encuentran los distintos entornos acuáticos en los que las actividades de inmersión involucren un alto riesgo. Uno de los entornos mencionados, que además es de alta importancia en el entorno que se desarrollan son los humedales.

Los humedales representan zonas donde coexisten en tiempo y espacio los sistemas acuáticos y terrestres, las formas de interacción entre ambos. Propician una gran diversidad biológica y servicios ambientales vitales para el bienestar humano. Son considerados uno de los ecosistemas biológicamente más diversos y son pieza clave en el ciclo hidrológico. La vegetación específicamente adaptada a estas condiciones se denomina hidrófita, y reemplaza a las especies terrestres normales.

En el presente trabajo se presenta el diseño del ROV, el cual se caracteriza por tener una estructura hermética que pueda alojar a los componentes electrónicos; es capaz de poder navegar con 3 grados de libertad, detenerse y encender luces cuando sea necesario. Además, se detallan los sensores y actuadores utilizados para proporcionar la capacidad de adquirir imágenes y auto propulsarse. Por último, se detallan las tarjetas y dispositivos de conexión electrónica que controlan a los actuadores y permiten la comunicación con la estación de control.

PLANTEAMIENTO DEL PROBLEMA

La exploración acuática supone un inmenso desafío para la humanidad ya que aproximadamente sólo conocemos el 5% de estos sistemas y se plantea su utilización como estímulo de vocaciones de los jóvenes hacia las disciplinas técnicas y científicas. La utilización de un ROV favorece a la investigación sin poner en peligro la vida de los investigadores.

El lago de Chapala siendo el humedal más grande de México y considerado de importancia internacional según la convención Ramsar, se encuentra próximo a ser insuficiente contra las demandas de la población que depende de él. Las causas principales son: la sobreexplotación de los recursos naturales, la contaminación de las aguas superficiales, así como la falta de programas que aseguren la sustentabilidad del mismo.

Debido a la contaminación que existe ocasionada principalmente por actividades industriales y la insuficiencia de plantas residuales, el agua del lago de Chapala no es apta completamente para contacto humano. Realizar una investigación de campo involucra una exposición directa con el medio contaminado lo cual dificulta la documentación de la flora, fauna y cambios visibles en zonas de riego. La utilización de un ROV favorece a la investigación sin poner en peligro la vida de los investigadores.

El vehículo submarino se sumergirá en un medio acuático. Por lo tanto, la plataforma deberá soportar diferentes fenómenos físicos, como son la flotabilidad, la presión y la hidrodinámica de los cuales dependerá su desempeño, por lo cual es imprescindible el buen diseño del mismo.

Debido a la contaminación que existe ocasionada principalmente por actividades industriales y la insuficiencia de plantas residuales, el agua del lago de Chapala no es apta completamente para contacto humano. Realizar una investigación de campo involucra una exposición directa con el medio contaminado lo cual dificulta la documentación de la flora, fauna y cambios visibles en zonas de riego.

JUSTIFICACIÓN

El submarino ECO VAPO es una herramienta de investigación, que facilitará el estudio e investigación de la flora y fauna submarina de los humedales, especialmente del Lago de Chapala, al igual que la exploración de lugares desconocidos o de difícil acceso.

Actualmente es reconocido el valor que tienen los humedales por sus funciones en el régimen hídrico, actuando como auténticos filtros. Además, son altamente productivos para albergar una vasta biodiversidad. a diferencia de antes, que se consideraban terrenos insalubres causantes de enfermedades. El lago de Chapala abastece más del 50% de las necesidades de Guadalajara y las zonas aledañas.



Una de las funciones más importantes, es que actúan como filtradores naturales de agua, esto se debe a que los tejidos de las plantas hidrófitas, almacenan y liberan agua, realizando un proceso

de filtración. También contribuyen a la protección contra desastres meteorológicos puesto que funcionan como reservorios de líquido.

La Comisión Nacional del Agua (CONAGUA) elaboró el inventario nacional de humedales (INH), gracias a esto sabemos que México cuenta con 6,331 humedales, de los cuales, por su importancia y características, 142 están consideradas dentro de la Convención sobre Humedales de Importancia Internacional. Actualmente son protegidos, dado que se ha reconocido su importancia ambiental, sin embargo, la falta de intereses y de buenas políticas de manejo no propician el correcto entorno para que los humedales puedan ser estudiados más a fondo. Si se contara con información clave de las condiciones en las que se encuentran los humedales, se desarrollarían mejores estrategias para la protección y aseguramiento de la sustentabilidad del humedal.

El estudio de estos ecosistemas puede servir para monitorear las especies presentes, así como su consumo de las poblaciones cercanas, garantizando así que los humanos que dependen de él directa o indirectamente puedan subsistir. Asimismo, se podrán elaborar programas de manejo sustentable de estas áreas de importancia para la conservación de la biodiversidad.

Las actividades de monitoreo y observación son fundamentales para la investigación, dado que permiten encontrar patrones, cambios y comparaciones de manera directa. Observar implica obtener información visual del objeto de observación, prestando atención en los elementos previamente mencionados. En el método científico empírico, la observación sirve para generar hipótesis sobre prácticamente cualquier situación.



En un ecosistema, una herramienta de monitoreo permitiría dar un acercamiento a las interacciones de los organismos dependiendo del contacto con el ser humano, la basura, los cambios de temperatura, los cambios en un periodo de tiempo y la modificación del entorno. El monitoreo como actividad para preservar busca como resultado de analizar los organismos, así como el impacto del ser humano, poder obtener datos fiables y actualizados para generar lineamientos para proceder con el uso de los recursos naturales, en este caso, de los humedales.

Con la aplicación del submarino ECO VAPO se obtendrán datos actualizados y relevantes acerca de las condiciones en las que se encuentra el agua. Estos datos pueden ser sumados para el diseño de un plan de acción para revertir el estado crítico en el que se encuentra este humedal en

particular. De aplicarse, se podría asegurar la sustentabilidad de una población que se desarrolla sus actividades aprovechando este recurso natural al mismo tiempo que se mantiene la riqueza de organismos.

La importancia en el desarrollo de un ROV submarino radica en las posibilidades que representa en cuanto a la satisfacción de necesidades de la investigación ambiental, el monitoreo de terrenos acuáticos preservados, la observación de especies marinas, así como una herramienta que facilite las actividades llevadas bajo el agua en el tema de la preservación de la biodiversidad.

OBJETIVOS DEL PROYECTO

• GENERAL

Desarrollar una herramienta tecnológica con una estructura hermética para cubrir los componentes electrónicos y que pueda desempeñar la tarea de recolectar evidencias contundentes, de imagen y video, así como registros y seguimiento de las condiciones y cambios manifestados en los humedales. Al utilizar esta herramienta y combinarla con un trabajo de investigación, será posible realizar el diseño de medidas preventivas y de recuperación del espacio natural. El enfoque ha sido aterrizado en el lago de Chapala, por la cercanía del lugar y por la relación que tiene con la ciudad de Guadalajara y la zona metropolitana.

• PARTICULARES

- Elaborar el diseño de una serie de diseños necesarios para el funcionamiento del ROV, los cuales involucran: un sistema de control remoto en un medio inalámbrico compatible con el protocolo Wifi, un sistema de navegación y control de estabilidad inercial basado en el módulo MPU6050.
- Codificar un programa completo sin fallos para el uso de los actuadores y sensores, así como una interfaz de usuario para la interpretación de datos.
- Fabricar un prototipo con una estructura hermética para cubrir los componentes electrónicos, que sea compatible con inmersión en medios acuáticos.
- Cumplir con la acreditación de este proyecto en cuanto a los módulos especificados para la acreditación del proyecto modular en las áreas correspondientes a los módulos propuestos en el área de electrónica.
- Fabricar el circuito impreso que conecta los diferentes módulos, microcontroladores y cableado necesario.
- Acondicionar los elementos y dispositivos utilizados para cumplir con los requerimientos de todas las áreas funcionales del proyecto.
- Configuración de dos servidores para la comunicación y transmisión de datos en tiempo real.
- Implementación de control mediante modelado matemático.

METODOLOGÍA

El equipo se le dedicó una gran parte del tiempo para encontrar una problemática real, después encontramos distintas soluciones. Una vez encontrada la problemática y la solución en mente, inicia la fase de planeación.

Para llevar a cabo la planeación del proyecto, primero fue necesario tomar en cuenta cual era la información y los recursos con los que se contaba. Después de analizar profundamente este aspecto, comenzó la fase de preparación en la que nos enfocamos a encontrar la forma de gestionar la investigación de la misma, así como la obtención de materiales necesarios.

Al inicio del proyecto, se llevó a cabo un análisis con los diferentes retos que había que vencer, así como una lluvia de ideas de distintas soluciones. Conforme se llevaba a cabo cada modulo funcional, el enfoque cambiaba del panorama global al modular. En esta etapa de trabajo, solo se trabajaba con el modulo que estaba en desarrollo.

La forma de realizar los avances en la construcción y diseño, se comenzaba por escribir a forma de boceto, escribiendo los pros y contras del diseño, así como los alcances que tiene y distintos pseudo funcionamientos. También se definían los parámetros de entrada, así como los resultados. Con esta información de nuestro lado, comenzaba la fase de investigación y planificación de fechas y presupuestos.

Cuando aparecían errores y malfuncionamiento en el diseño, se regresaba el proceso para llevar a cabo las reparaciones y cambios al diseño, unificando las ideas y buscando la solución del problema presente. Cuando dicho modulo cumplía con su función, se re integraba al sistema completo.

Para el desarrollo del prototipo el método utilizado se aplica en el diseño de prototipo de drones, el cual permite dividir el proyecto en 3 fases:

En la primera fase se trabajó la estructura del robot, para poder así evaluar las dimensiones, estructura física, materiales, procedimientos, tiempos y costos de construcción. El siguiente paso fue el diseño y construcción de los sistemas que componen al Prototipo (partes, elementos, componentes). En la fabricación y ensamble se determinó si es necesaria alguna modificación en los parámetros y las dimensiones de los componentes.

La segunda fase corresponde al control e implementación de la etapa de potencia donde se definió el circuito eléctrico que permitió controlar el movimiento, también se establecieron los requerimientos de los dispositivos eléctricos a utilizar (sensores, motores, cámara web etc.). Se determinó la velocidad, sentido de giro, potencia, corriente y voltajes con los que trabajaron los motores. Realizando posteriormente la caracterización de cada uno de los sensores y otros dispositivos a utilizar. El punto siguiente estableció la comunicación entre el prototipo y la base en tierra. Finalmente se implementó un sistema de control, que satisfacía las condiciones de desplazamiento básicas.

La última fase tiene como finalidad la realización de pruebas de tipo mecánico, electrónico y de software (interfaz de comunicación, control de dispositivos, etc.) que permitió modificar y mejorar los modelos usados anteriormente para el diseño del prototipo.

ANTECEDENTES Y ESTADO DEL ARTE

Dado que el dispositivo principal del desarrollo es un ROV, se realizará un análisis de los sumergibles y los distintos tipos que existen. Para clasificar los ROVs se subdividen en los siguientes modelos.

SUMERGIBLES

Entre esta gran familia de sumergibles se pueden subdividir a su vez:

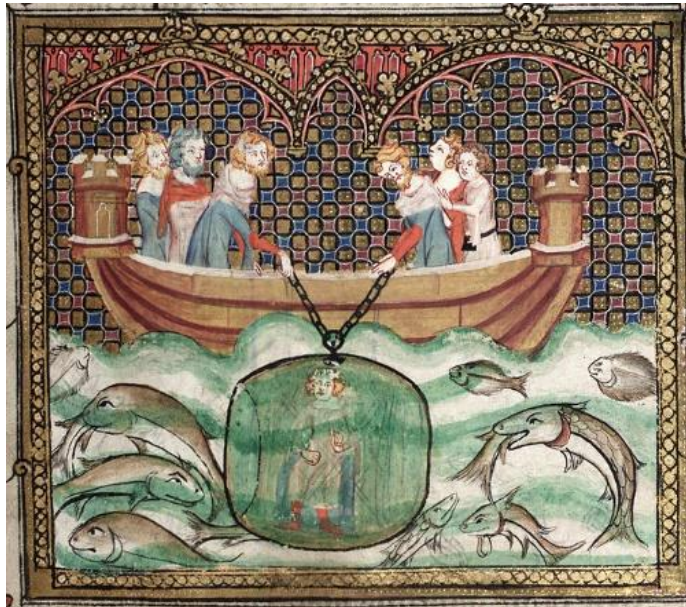
- **Submarinos militares.** Con fines y actividades estrictamente militares.
- **Submarinos no militares:** Realizan actividades submarinas de investigación, evaluación y monitoreo, abarcando distintas áreas.
- **Sumergible no tripulado:** Estos sumergibles, también están comprendidos en un número de diferentes subclases.
- **Vehículos sumergibles atados a una embarcación,** el cual funciona como una plataforma para el uso de distintos sensores.
- **ROV (Vehículo Operado Remotamente).** Son controlados y alimentados desde la superficie por un operador-piloto a través de un cordón umbilical o usando el control remoto.
- **UUV (Unmanned Underwater Vehicle).** Este vehículo “sin ataduras” no posee su propia capacidad de control a bordo, sino que está controlado por un operador remoto a través de algún tipo de enlace de comunicaciones.
- **AUVs (Autonomus Underwater Vehicle):** sistema submarino que contiene su propia capacidad de control de sí mismo durante el cumplimiento de una tarea predefinida.



ANTECEDENTES HISTORICOS

Así como dependiendo del desarrollo tecnológico que ha tenido la humanidad, ha incrementado la exploración espacial, también de la mano la navegación marítima y submarina ha avanzado a la misma velocidad. El lugar más inexplorado de nuestro planeta es el océano y sus profundidades. No obstante, el entorno submarino es igual o más peligroso que el vacío espacial. Una de las primeras menciones que se hacen sobre los aparatos tecnológicos submarinos, se realizó por el filósofo Aristóteles. Muchos de los diseños creados en ese tiempo vinieron de la mano de personas imaginativas y creativas. Desde la literatura antigua podemos encontrar

muchos personajes míticos como Poseidón, las Nereidas, las Sirenas y los Tritones no son sino la personificación del deseo de los hombres de dominar la profundidad del mar.



A lo largo de los siglos XVI, XVII y XVIII se producen cada vez con más frecuencia, dibujos y descripciones sobre ingenios capaces de navegar bajo las aguas, apareciendo nuevos cronistas como D. Francisco de Toledo, que narró el interés del Emperador Carlos I por tener un submarino, y el matemático británico William Bourne que describió en 1578 con todo lujo de detalles, un bote submarino que disponía de tanques de lastre variable y de tubo de aireación precursor del snorkel. Algunos ejemplos de proyectos submarinos que llevaron a cabo de manera formal operaciones bajo el agua son:

Van Drebbel

Diseñado por Cornelius Van Drebbel, inventor del Termómetro, es considerado como uno de los primeros intentos serios de realizar un vehículo submarino. Con el apoyo de Jacobo I de Inglaterra construyó un bote submarino en 1620, que navegó por el Támesis con 15 tripulantes desde Westminster a Greenwich haciendo el recorrido de ida y vuelta en inmersión a una velocidad media de 3 nudos.

El Mary

Durante el resto del siglo XVII y hasta bien avanzado el siglo XVIII se construyeron numerosos botes sumergibles inspirados en el de Van Drebbel. En uno de esos submarinos llamado Mary murió su constructor el norteamericano Jhon Day aplastado por la presión del mar a 30 m de profundidad.

Fue el primer submarino provisto de hélices, una de eje vertical para cambiar de cota y otra de eje horizontal para avanzar y retroceder.

Primera Guerra Mundial

Al finalizar la Primera Guerra, los sumergibles habían salido de la infancia. Todas las naciones participantes habían adquirido una importante experiencia en el uso de estos, pero sobre todo de

la experiencia alemana. Sin embargo, seguían teniendo una autonomía en superficie muy limitada.

El siguiente gran impulso tecnológico se produjo para la segunda Guerra Mundial. Esto se pudo apreciar notablemente con la entrada en Guerra de los EE.UU. con su gran potencial económico e industrial, y el gran impulso dado a la construcción naval, a la aviación y a los sistemas de localización y detección (sonar y radar).

Actualidad

Actualmente la mayoría de la financiación de programas de desarrollo de AUVs proviene de procesos políticos, estableciendo estos unos objetivos de desarrollo diferentes a los que pueda marcar las necesidades del mercado. A pesar de ello, como se ha indicado, no todos estos programas tienen sus objetivos establecidos erróneamente. Hay una serie de organizaciones, sobre todo en EE.UU., que trabajan activamente sobre los problemas importantes de la investigación.

Se aprecia también una serie de mercados de AUVs emergentes. Aunque no está claramente definido el nivel de interés por parte de personas y organizaciones, hay indicios que sugieren una mayor cota de oportunidades para la comercialización de AUVs en los próximos años.

La única tendencia que se ve actualmente, son los dos caminos que el mercado ha establecido:

- El desarrollo de dispositivos de bajo costo. Se prevé que estos sistemas, eventualmente, se utilicen en grupos de vehículos cooperantes.
- Sistemas sofisticados que contienen todo tipo de sensores complejos configurados para satisfacer las necesidades específicas del usuario. Aunque no son precisamente de bajo coste, pueden acometer tareas que, si se hace de cualquier otro modo, incrementaría el importe de las misiones.



Algunos submarinos comerciales en la actualidad son: **Alister AUV**, **Hugin 1000**, **AUV 62-MR**, **REMUS 600**.

MARCO TEÓRICO

ROV

Los vehículos operados remotamente, se traducen en un robot submarino no tripulado, conectado a una unidad de control en la superficie, por medio de un cable umbilical. La energía y los comandos u ordenes viajan bidireccionalmente por este medio.

El cable umbilical sirve de medio para transmitir también los datos de las cámaras de video del ROV, los datos de los sensores y de los sonares a la unidad de control en la superficie.

El ROV se encuentra constituido por distintos módulos funcionales que son operados desde una base en tierra.

La función principal de la base en tierra es monitorear procesos y valores del sistema. Los valores mostrados describen el estado actual del sistema.

La mayoría de los ROVs están equipados con al menos una cámara de vídeo y luces. El equipo adicional se agrega comúnmente para ampliar las capacidades del vehículo. Estos pueden incluir sonares, magnetómetros, cámaras fotográficas, un brazo manipulador, herramienta de corte, sistemas para toma de muestras, e instrumentos para medir parámetros.

HUMEDALES

Los humedales son considerados como una zona cuya superficie sufre de inundaciones de manera intermitente. El suelo al cubrirse de agua queda saturado, quedando desprovisto de oxígeno y dando a lugar un ecosistema híbrido entre la fauna acuática y terrestre.

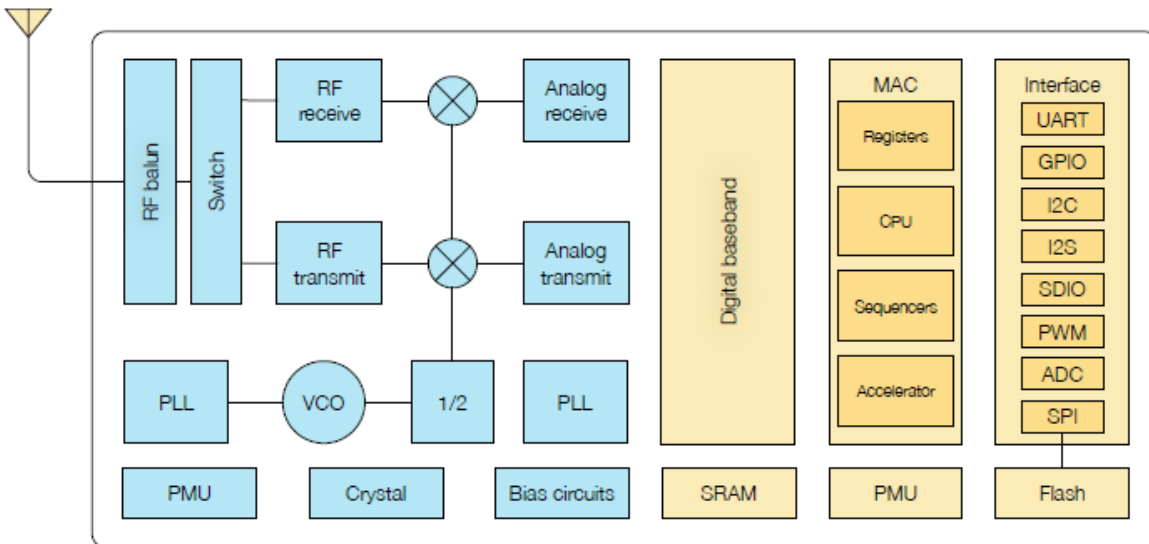
La categoría biológica de humedal comprende zonas de propiedades geológicas diversas: ciénagas, esteros, marismas, pantanos, turberas, así como las zonas de costa marítima que presentan anegación periódica por el régimen de mareas.

Se clasifican en función de los objetivos que se persiguen al estudiarlos. Los más comunes son: criterio morfológico utilizado principalmente para divulgación, hidrogenético que se basa según el origen y usos del agua o para demandas de la misma, funcional según sus hábitats con motivo de conservación medioambiental y por criterios estructurales enfocados en el punto de vista de gestión.

Se encuentran entre los ecosistemas más diversos y productivos. Proporcionan servicios esenciales y suministran toda nuestra agua potable.

ESP8266

Se trata de un módulo basado en un microcontrolador de 32 bits, con la capacidad de cumplir distintas funciones utilizando como medio el aire, mediante el protocolo WIFI. Sus funciones se logran utilizando un consumo mínimo de energía. Contiene precargado un sistema operativo en tiempo real que permite la utilización del 80% de la potencia de procesamiento disponible para la programación y desarrollo de aplicación del usuario.



PROTOCOLO TELNET

Consiste en un protocolo de internet estandarizado que permite interconectar terminales y aplicaciones en internet. El protocolo es aplicado a una conexión TCP que permite establecer una comunicación entre un cliente y servidor, por medio de caracteres ASCII, codificado a 8 bits. Por lo tanto, brinda un sistema de comunicación bidireccional.

Como propósito, el protocolo proporciona un servicio de comunicaciones orientado a bytes de 8 bits en general y bidireccional. Permite un método estándar de comunicar entre sí terminales, y procesos orientados a terminales. Está previsto que el protocolo se pueda usar también para la comunicación de enlace, así como proceso a proceso.

El protocolo se basa en tres conceptos básicos: Paradigma Terminal Virtual de Red (NVT) el principio de opciones negociadas y las reglas de negociación. A este protocolo base se le aplican en conjunto otros protocolos como TCP/IP: FTP, SMTP y POP3.

PROTOCOLO WIFI

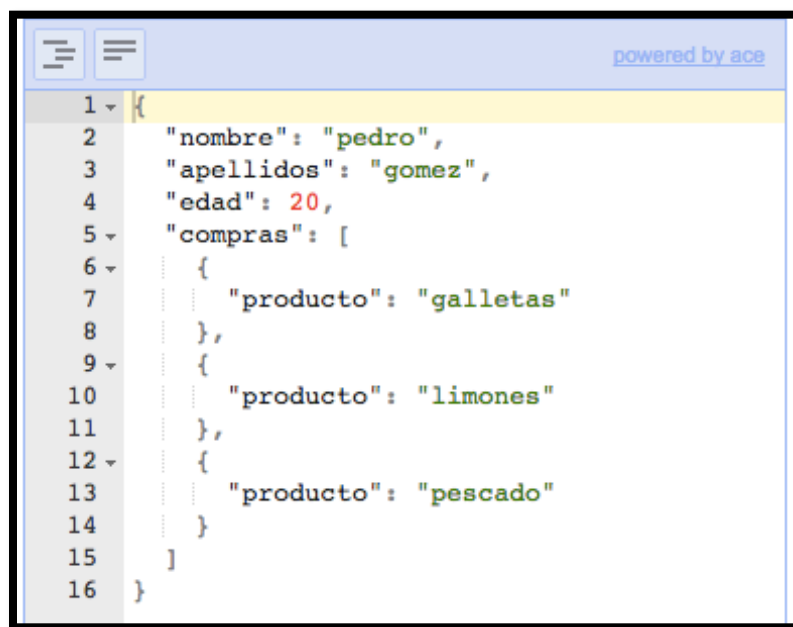
Se trata de un conjunto de especificaciones para redes locales inalámbricas basadas en el estándar IEEE 802.11. Su nombre se refiere a una abreviación del término Wireless Fidelity. Es posible implementar redes que conecten una gran variedad de dispositivos compatibles, siempre y cuando estén cercanos geográficamente. Estas redes no utilizan cables, ya que transmiten toda la información por radiofrecuencia.

Estas redes, conocidas como WLAN posibilitan el acceso a Internet de alta velocidad en radios menores a 100 metros, o sea, áreas relativamente pequeñas. Otra posibilidad es la conexión a través de altas frecuencias, pero en ese caso es necesario la autorización de un organismo competente. Este tipo de conexión, conocido como streaming, es cada vez más utilizado. De esta forma, la tecnología Wireless se presenta como una alternativa a las redes convencionales, ya que posibilita las mismas funcionalidades, pero de una forma flexible, de fácil configuración y con buena conectividad.

OBJETOS JSON

JSON es un formato de datos basado en texto que sigue la sintaxis de objeto de JavaScript, popularizado por Douglas Crockford. Aunque es muy parecido a la sintaxis de objeto literal de JavaScript, puede ser utilizado independientemente de JavaScript, y muchos ambientes de programación poseen la capacidad de leer (analizar; parse) y generar JSON.

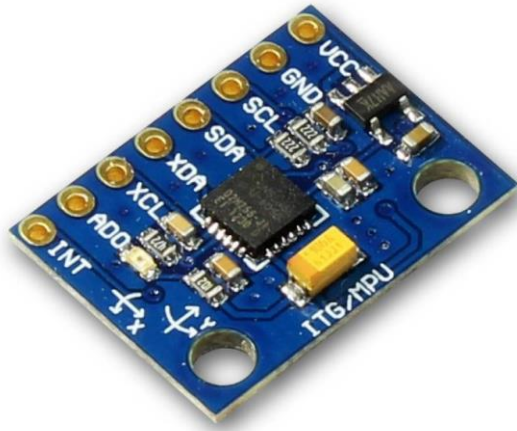
Los JSON son cadenas - útiles cuando se quiere transmitir datos a través de una red. Debe ser convertido a un objeto nativo de JavaScript cuando se requiera acceder a sus datos. Ésto no es un problema, dado que JavaScript posee un objeto global JSON que tiene los métodos disponibles para convertir entre ellos. es un formato de texto para la serialización de datos estructurados.



```
1 {  
2   "nombre": "pedro",  
3   "apellidos": "gomez",  
4   "edad": 20,  
5   "compras": [  
6     {  
7       "producto": "galletas"  
8     },  
9     {  
10      "producto": "limones"  
11    },  
12    {  
13      "producto": "pescado"  
14    }  
15  ]  
16 }
```

MPU6050

La unidad de procesamiento de movimiento fusiona sensores con 9-ejes, probado en el campo de teléfonos móviles, tabletas, aplicaciones, controladores de juegos, mandos a distancia, puntero de movimiento y otros dispositivos de consumo. Tiene integrado un giroscopio MEMS de 3 ejes, un acelerómetro de 3 ejes también MEMS. Cuenta además con un procesador digital de movimiento motor acelerador de hardware con un puerto I2C auxiliar. Cuando se conecta a un magnetómetro de 3 ejes, el MPU-60X0 entrega una salida completa de 9 ejes para su primario I2C o puerto SPI. Combina la aceleración y el movimiento de rotación más la información de rumbo en un único flujo de datos para la aplicación. Presenta un diseño más compacto y tiene ventajas de costos inherentes en comparación con soluciones discretas de giroscopio más acelerómetro. Está diseñado para interactuar con múltiples sensores digitales no inerciales, tales como sensores de presión, en su bus I2C auxiliar maestro.



Sensor Giroscópico

Un giroscopio es un dispositivo que funciona para medir velocidades angulares basándose en el mantenimiento del impulso de rotación. Si intentamos hacer girar un objeto que está girando sobre un eje que no es el eje sobre el que está rotando, el objeto ejercerá un momento contrario al movimiento con el fin de preservar el impulso de rotación total.

Acelerómetro

Mide la aceleración, inclinación o vibración y transforma la magnitud física de aceleración en otra magnitud eléctrica que será la que emplearemos en los equipos de adquisición estándar. Los rangos de medida van desde las décimas de g, hasta los miles de g.

QT CREATOR

Se trata de un entorno para crear rápidamente y con bajo coste. Permite diseñar, desarrollar, desplegar y mantener software compatible con múltiples dispositivos.



Lenguaje de Programación C++

Considerado un lenguaje híbrido basado en C, diseñado como una extensión del mismo por Bjarne Stroustrup integrando mecanismos que permitan la manipulación de objetos.

Maneja distintos paradigmas como lo son estructurada y la programación orientada a objetos. Por esto se suele decir que el C++ es un lenguaje de programación multiparadigma. Actualmente existe un estándar, denominado ISO C++, C# es un lenguaje propietario de Microsoft que mezcla las características básicas de C++ simplificándolas al estilo Java y ofreciendo un framework. C# forma parte de la plataforma .NET

Navegación Inercial

Se trata de una técnica de navegación autónoma que emplea mediciones de acelerómetros y giroscopos, y que permite, a través de cálculos, conocer la velocidad, posición y orientación de una nave a partir de un punto inicial con orientación y velocidad conocidos. Su principio de operación marca que, al medir la aceleración de un vehículo, podemos integrar la aceleración con respecto al tiempo para obtener la velocidad y luego integrar la velocidad para obtener posición. Por lo tanto, suponiendo que conocemos la posición y la velocidad iniciales, podemos determinar la posición del vehículo en cualquier momento posterior. Por otra parte, a través de los giroscopios podremos conocer el cambio de orientación del vehículo a partir de una orientación inicial.

Unidad de Navegación Inercial

La Unidad de Medición Inercial IMU es el corazón de todo sistema de navegación inercial, su propósito es suministrar señales de aceleración con respecto a un sistema de referencia. Las unidades de medida (IMU) suelen contener tres giroscopios ortogonales y tres acelerómetros ortogonales que miden la velocidad y aceleración angular y lineal respectivamente.

SERVOMOTORES

Es un motor eléctrico al que podemos controlar tanto la velocidad, como la posición del eje que gira (también llamada dirección del eje o giro del rotor). Los servomotores no giran su eje 360°, como los motores normales, solo giran 180° hacia la izquierda o hacia la derecha (ida y retorno).

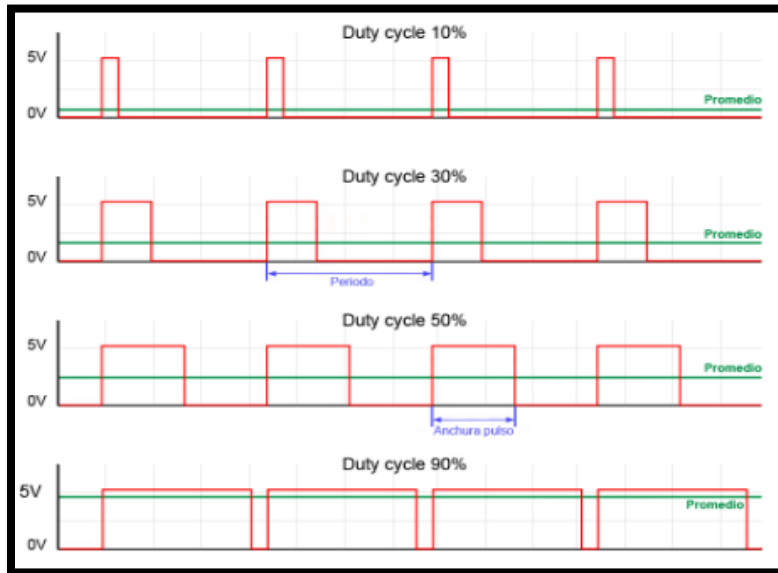


Composición de Servomotor

- Un motor eléctrico: es el encargado de generar el movimiento, a través de su eje.
- Un sistema de regulación: formado por engranajes, que actúan sobre el motor para regular su velocidad y el par. Mediante estos engranajes, normalmente ruedas dentadas, podemos aumentar la velocidad y el par o disminuirlas.
- Un sistema de control o sensor: circuito electrónico que controla el movimiento del motor mediante el envío de pulsos eléctricos.
- Un potenciómetro: conectado al eje central del motor que nos permite saber en todo momento el ángulo en el que se encuentra el eje del motor.

SEÑAL PWM

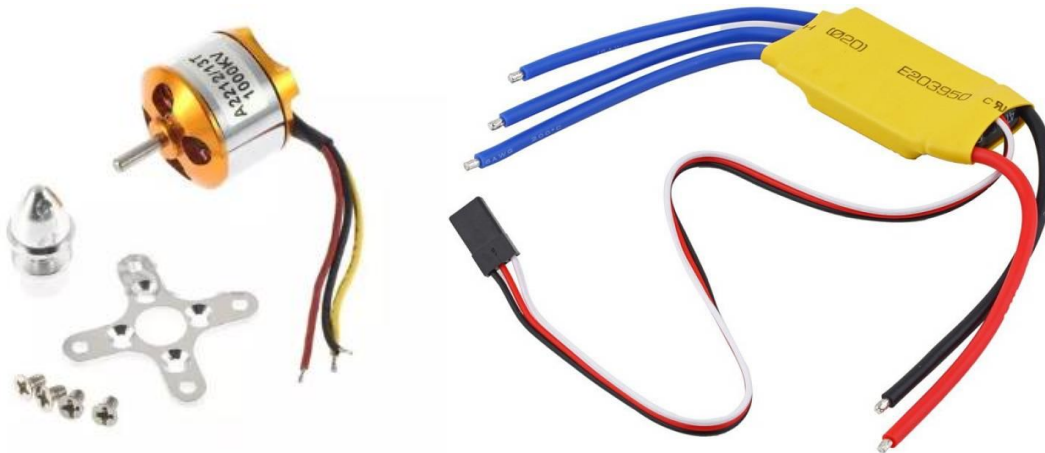
La modulación por ancho o de pulso es un tipo de señal de voltaje utilizada para enviar información o para modificar la cantidad de energía que se envía a una carga. Este tipo de señales es muy utilizado en circuitos digitales que necesitan emular una señal analógica. Las señales son de tipo cuadrada o sinusoidales en las cuales se le cambia el ancho relativo respecto al período de la misma, el resultado de este cambio es llamado ciclo de trabajo y sus unidades están representadas en términos de porcentaje.



MOTORES BRUSHLESS

Estos motores sin escobillas permiten que consigamos un gran rendimiento y una gran potencia a cambio de un gran consumo. Debido a esto, tenemos que utilizar baterías Lipo (Polímero de litio), son baterías con poca densidad de energía en comparación con otras, pero con una característica única, pueden entregar una gran cantidad de potencia.

La corriente eléctrica suministrada por una batería generalmente, pasa directamente por las bobinas dispuestas en el estator. La corriente eléctrica que circula por las bobinas genera pequeños campos magnéticos que obligan al rotor a girar. El rotor está compuesto por imanes permanentes. Estos imanes también pueden ser de ferrita, de hecho, los motores brushless de los encendidos electrónicos de las motocicletas se hace con este material. Estos motores van asociados con un variador. El variador lo que hace es enviar la tensión de alimentación a las bobinas de forma secuencial, de este modo, los polos del rotor se van moviendo según el campo magnético generado por las bobinas de forma secuencial. La velocidad del rotor y su eje, dependerá de la velocidad de secuenciación del variador. Pero el variador no se limita solamente a cambiar la velocidad de secuencia de alimentación, sino que además aumenta o disminuye la tensión de alimentación de las bobinas para extraer el máximo rendimiento al motor brushless.



Ventajas

- Mayor eficiencia (menos pérdida por calor)
- Mayor rendimiento (mayor duración de las baterías para la misma potencia)
- Menor peso para la misma potencia
- Requieren menos mantenimiento al no tener escobillas
- Relación velocidad/par motor es casi una constante
- Mayor potencia para el mismo tamaño
- Mejor disipación de calor
- Rango de velocidad elevado al no tener limitación mecánica.
- Menor ruido electrónico (menos interferencias en otros circuitos)

Desventajas

- Mayor costo de construcción
- El control es mediante un circuito caro y complejo
- Siempre hace falta un control electrónico para que funcione (ESC's), que a veces duplica el costo

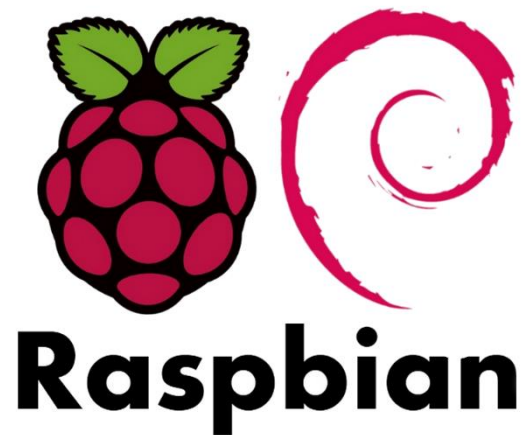
BATERIAS LIPO

Las baterías LiPo (abreviatura de Litio y polímero) son baterías recargables usadas en aplicaciones que requieren corrientes superiores a 1A con bajo peso y tamaño reducido. Son utilizadas en celulares, drones, sistemas de radio control, etc. Tienen una vida útil 2 a 3 años, traducido en 500 cargas completas aproximadamente y comparadas con otro tipo de baterías son más eficientes. Son utilizadas por manejar una alta tasa de descarga, la cual va desde 1 a 25 amperes.



RASPBERRY PI

Raspberry PI es una placa computadora de bajo coste, de un tamaño reducido que fue desarrollado en 2011, con el objetivo de estimular la enseñanza de la computación e informática en las universidades.



El concepto es el de un ordenador desnudo de todos los accesorios que se pueden eliminar sin que afecte al funcionamiento básico. Está formada por una placa que soporta varios componentes necesarios en un ordenador común y es capaz de comportarse como tal.

ARDUINO

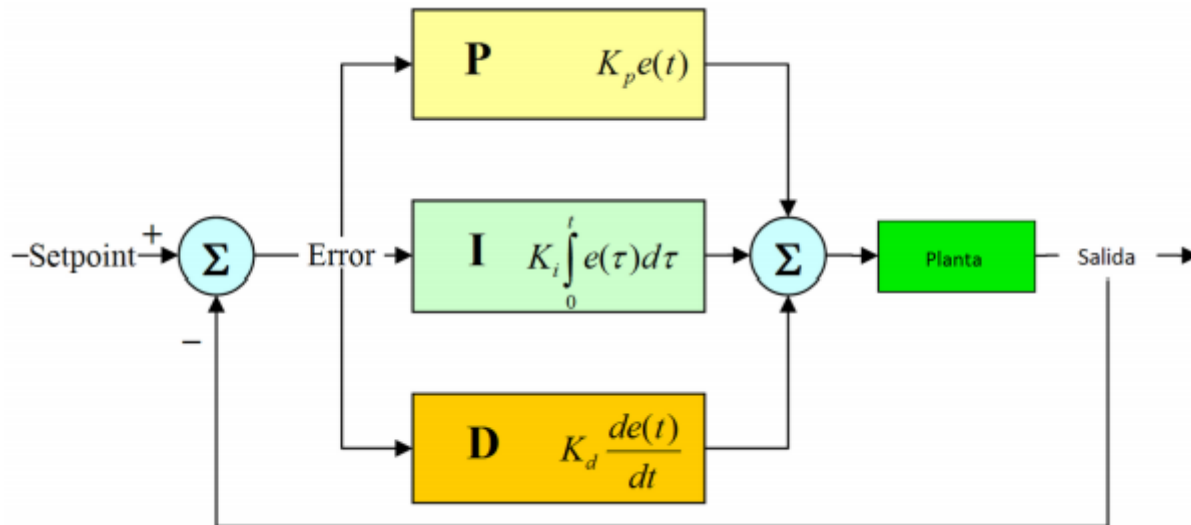
Arduino es una tarjeta electrónica que sirve como plataforma para distintos tipos de proyectos de código abierto. Tiene la capacidad de leer diferentes entradas para procesarlas y generar una salida proporcional. Utiliza su propio lenguaje de programación basado en Processing. Fue desarrollado en Ivrea Interaction Design Institute dirigido a estudiantes que carezcan de antecedentes en el área de la electrónica.



Actualmente gracias al crecimiento que ha tenido en la comunidad se utiliza para proyectos basados en internet de las cosas, dispositivos wereables, impresión 3d y entornos embebidos.

CONTROL PID

Utilizado como un sistema de control retroalimentativo, el control PID se usa comúnmente en la industria. El funcionamiento principal radica en que continuamente calcula el error entre el valor y el punto establecido como referencia del sistema. Una vez encontrada, el cálculo proporcional, derivativo e integral aplica una corrección en la salida. El nombre del método de control PID proviene de Proporcional, Integral y Derivativo.



Control proporcional

Proporcional es una acción de control que reproduce cambios de la entrada con cambios en la salida. La acción proporcional del controlador responde a los cambios presentes en la entrada y genera inmediatamente y proporcionalmente cambios en la salida.

Control Integral

La integral es una acción de control que provoca un cambio en la señal de salida respecto del tiempo a una razón proporcional de la cantidad de error. La acción integral del controlador responde a un error acumulado en el tiempo, cambiando la señal de salida tanto como se necesite para eliminar completamente el error. Si la acción proporcional (P) le dice a la salida tanto desplazarse cuando un error aparece, la acción integral (I) le dice a la salida que tan rápido moverse cuando un error aparece. Si la acción proporcional (P) actúa en el presente, la acción integral (I) actúa en el pasado. Por tanto, que tan rápido la señal de salida es controlada por la acción integral depende de la historia del error en el tiempo: cuánto error existió, y que duración. Esta acción de control maneja la salida para aumentar y aumentar su valor conforme haya una diferencia entre PV y SP.

Control Derivativo

La derivada, o razón de cambio, es una acción de control que realiza un desplazamiento en la señal de salida proporcional a la tasa a la cual cambia la entrada. La acción derivativa del controlador reacciona a que tan rápido cambia la entrada respecto al tiempo, alterando la señal de salida en proporción con la tasa de cambio de entrada. Si la acción proporcional (P) le dice a la salida que tan lejos ir cuando un error aparece, la acción derivativa (D) le dice a la salida que tan lejos ir cuando la entrada cambia. Si la acción proporcional (P) actúa en el presente y la acción

integral (I) actúa en el pasado, la acción derivativa (D) actúa en el futuro. Eficazmente “anticipa” el sobre impulso intentando una respuesta de salida acorde que tan rápido que tan rápido la variable de proceso está creciendo o cayendo.

HIPÓTESIS

La estructura de submarino puede sumergirse a mínimo 2 metros de profundidad sin poner en riesgo a los componentes internos del submarino, la cámara funciona logrando tomar fotos y video; y permite al usuario observar en tiempo real lo que suceda frente al submarino durante una inmersión.

Utilizada correctamente, la herramienta que diseñamos satisface y facilita la obtención de datos, viables y de primera mano de las condiciones hidrológicas que presenta el lago de Chapala. Al llevar al análisis dichos datos se podría comprender más acerca del deterioro del sistema subacuático, y se podría trabajar en establecer un método de reparación o de frenado del deterioro desmedido, que pueda ocasionar un desabasto de la población dependiente del lago.

DISEÑO DE PROTOTIPO

Resumen

El proyecto consiste en el diseño y armado de un robot submarino que sea capaz de controlar completamente sus ejes de movimiento y aceleración. Los movimientos del submarino se llevan a cabo por medio de un microcontrolador que se encarga de posicionar servomotores, así como acelerar y mantener motores brushless a prueba de agua, dependiendo de módulos de giroscopio y acelerómetros. El diseño cuenta con una cámara de alta definición.

El submarino estará unido a una antena flotante, que enlazará comunicación con el control del usuario. La unión se lleva a cabo con un cable en el cual se realiza la comunicación serial entre el submarino y la antena flotante. Dicha boya, recibirá datos del submarino y los comunicará bidireccionalmente entre él y el usuario. Los datos que transmitirá son las lecturas de los sensores, así como el estado y la activación de actuadores.

El control del submarino está basado Arduino y Raspberry. El primero ejecuta los actuadores del submarino por medio del microcontrolador además recibe los datos de los sensores y envía los datos a una base de operaciones alojada en una PC, incluyendo la imagen en video de la cámara a bordo. La PC se conecta por medio de WIFI con la antena flotante y el submarino está anclado a la boya.

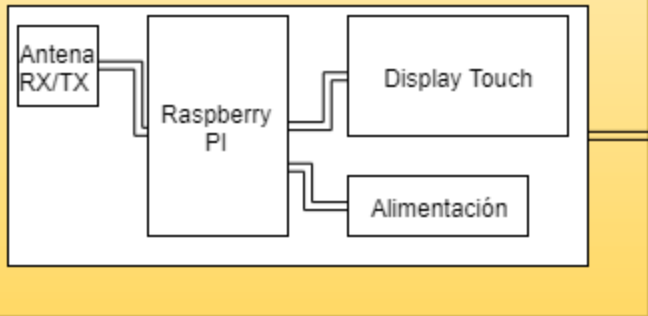
Base terrestre

Desde una computadora se podrá visualizar en tiempo real el camino que recorre el submarino grabando con la cámara a bordo. Se analizarán las variables del sistema, así como la integridad del mismo. Los datos de los sensores podrán ser monitoreados en tiempo real.

El programa estará enlazado por WIFI a la boya. Por medio de este enlace se enviarán los comandos de movimiento y funcionamiento para el ROV y se recibirá la información del sistema enviados y recibidos en formato JSON.

Interfaz de Usuario

Los controles serán botones Touch. El mismo display mostrará imágenes de la cámara, así como información del sistema explorador.

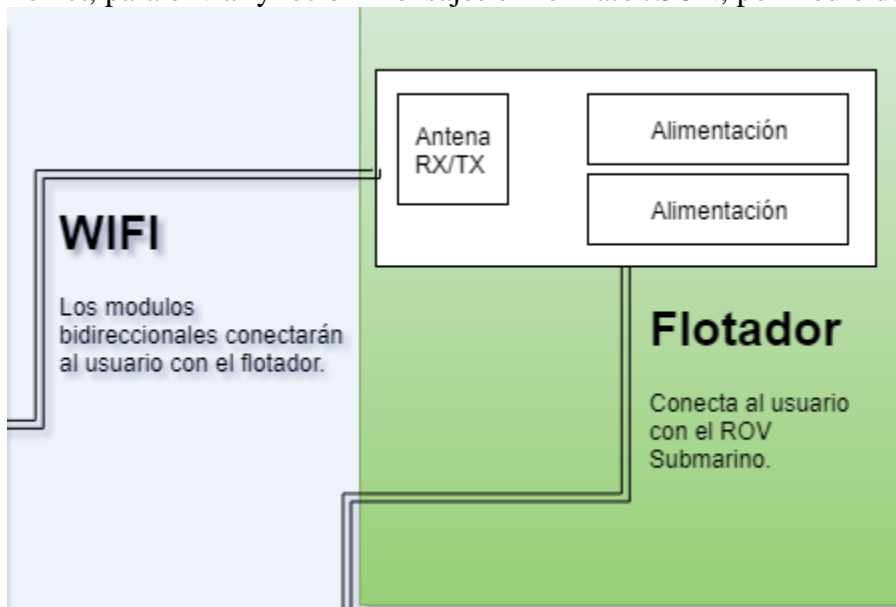


Base Flotante

En la superficie del agua, se encontrará situada encima de flotadores y con material plástico completamente aislada.

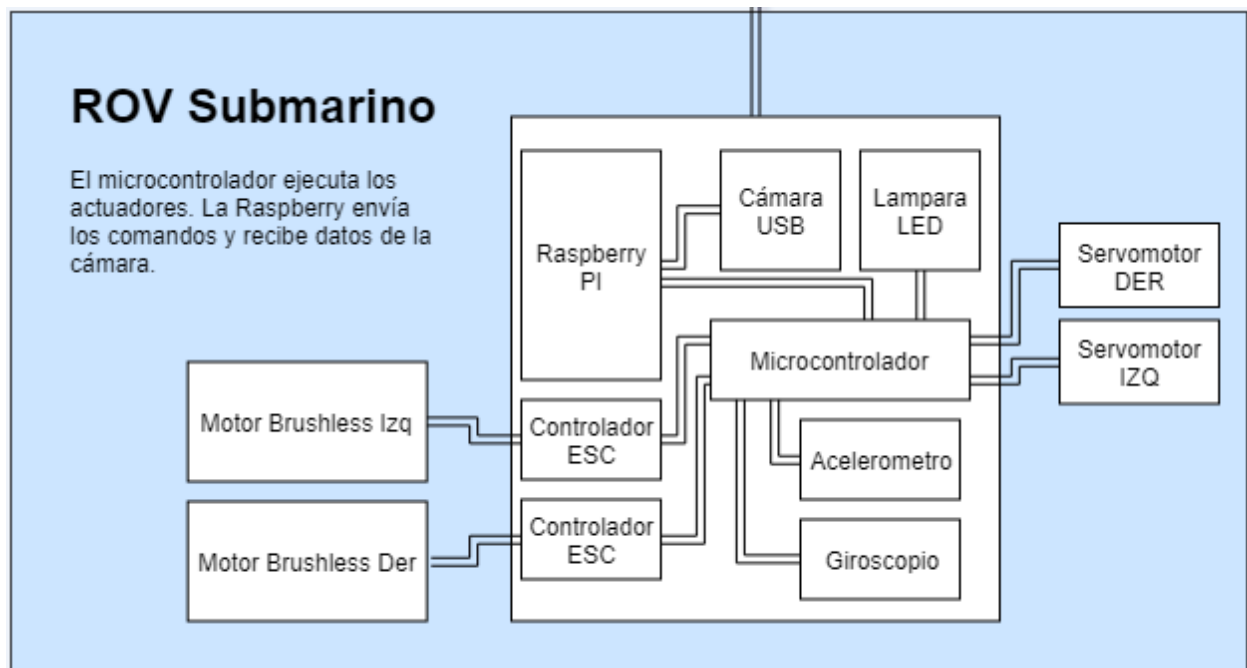
En esta base, se encontrará el punto de acceso de la base terrestre. Con este acceso se llevará a cabo envío y recepción de datos. Se busca la comunicación por cables desde la Raspberry al ESP8266 (Módulo de Wifi) y comunicación inalámbrica hasta la base terrestre.

La base flotante contiene el servidor de comunicación bidireccional utilizando el protocolo Telnet, para enviar y recibir mensajes en formato JSON, por medio del Wifi.



Submarino

Alimentado desde la base flotante, y además comunicado alámbricamente con ella. El sistema está basado en Raspberry apoyada en un microcontrolador para los movimientos del motor, así como el control de navegación y adquisición de video en tiempo real.

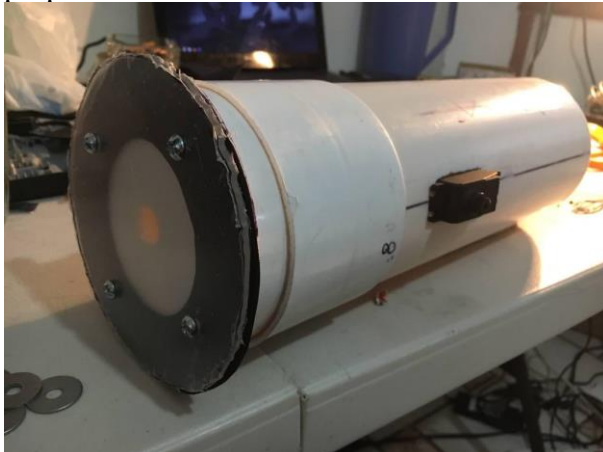


Diseño de Fuselaje

Para la elaboración del prototipo se buscaron materiales económicos que pudieran ser duros, al igual que livianos. Entre las opciones que se manejaron, aleaciones de plástico son las que mostraban un mayor potencial para ser utilizado. El PVC en especial, desde etapas tempranas del desarrollo se encaminó como la mejor opción.

El PVC es el nombre común con el que se conoce al policloruro de vinilo, un plástico que surge a partir de la polimerización de distintos plásticos. La utilización de PVC fue una buena opción a la impresión 3D, debido a que es un material de baja densidad, es resistente a la abrasión, es versátil, fácil de conseguir, económico y duradero.

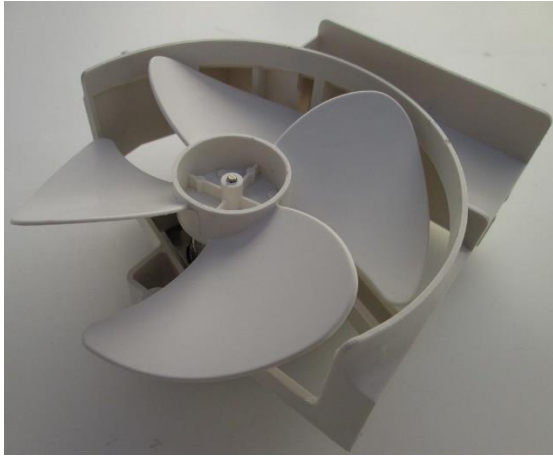
El diseño hecho consiste en un contenedor principal, tapa trasera, tapa delantera, aletas, cable umbilical, que contiene dentro una plataforma para anclar las tarjetas y circuito. El contenedor principal, consiste en un tubo PVC de 4 pulgadas, el cual ha sido perforado para mostrar los ejes de los motores laterales, así como el motor de la propela. Los motores salen del fuselaje para ejecutar funciones de movimientos en las aletas, así como el movimiento en el motor principal de propulsión.



La tapa trasera y delantera, son ambas tapas de PVC de 4 pulgadas, anteceditas por un cople de PVC de 4 pulgadas. La tapa delantera contiene una ventana de acrílico, sellada con silicón y reforzada con material blando de plástico, y tornillos de presión. La tapa trasera tiene salidas de los cables de alimentación del motor de propulsión.

Las aletas son dos trozos sellados de PVC, con una tapa hecha a medida, anclada firmemente a los motores servo los cuales se encuentran en los laterales del fuselaje. Los motores están colocados a 90 grados.

Los motores han sido separados en su totalidad, para ser llenados de grasa especial, que sirva como material hidrofóbico.



Acondicionamiento de motores

El motor principal consiste en un motor sin escobillas del modelo A2212, de 12 Amperes. Este motor será el principal, y con él se propulsará el submarino. La alimentación de este motor proviene de las baterías LiPo ZIPPY de 2200 mAh. La batería tiene 3 celdas y la salida es de 11.1 V 2200mAh de almacenamiento de carga. Esta batería es marca ZIPPY Compact, la cual es reconocida como una de las mejores del mercado.

El método de control tanto de los Servo Motores, así como del motor sin escobillas es por medio de señales PWM. Dichas señales en caso de los servos marcan los grados que se mantiene estacionado el torque de los mismos, así como las revoluciones por minuto del motor brushless.

Para la navegación del submarino, los motores brushless en los laterales se mueven opuestamente para ladear lo suficiente el submarino, paralelamente a ellos, el motor sin escobillas es desplazado por el tercer servo, el cual proporciona la mayor cantidad de torque.



El movimiento de los servomotores difiere del movimiento del motor sin escobillas. Los primeros se encargan meramente de mover los elementos que causan la dirección. El último, se encarga del empuje y es independiente del movimiento que realicen los demás actuadores.



Comunicaciones

La necesaria interacción del proyecto con el agua, advierte que es necesaria la utilización de medios inalámbricos para controlar y recibir información del submarino. La comunicación entre el control remoto y la base flotante se llevará a cabo entre la PC y el Módulo WIFI colocado en la boya.

Para el funcionamiento del sistema, existen dos enlaces de comunicaciones principales. El primero de ellos es el realizado para la comunicación entre el operador con la PC y la base flotante mediante WIFI. El segundo es la conexión entre el microcontrolador encargado de controlar los motores.

Los medios utilizados serán el aire para Wifi, y medios físicos para los microcontroladores. Se descartó el uso del agua como un medio, dado que a pesar de en estado puro ser un aislante, la conductividad del agua en la naturaleza incrementa por distintos factores. Mientras más conductividad exista en el agua mayor será la atenuación que presentan las señales que lo atraviesen.

Punto de Acceso Wifi

Las comunicaciones en el submarino desempeñan el papel del control remoto. Desde un punto de acceso personalizado, se conecta el submarino con la base terrestre. Una vez conectados, la información se mueve bidireccionalmente para ejecutar comandos de control.

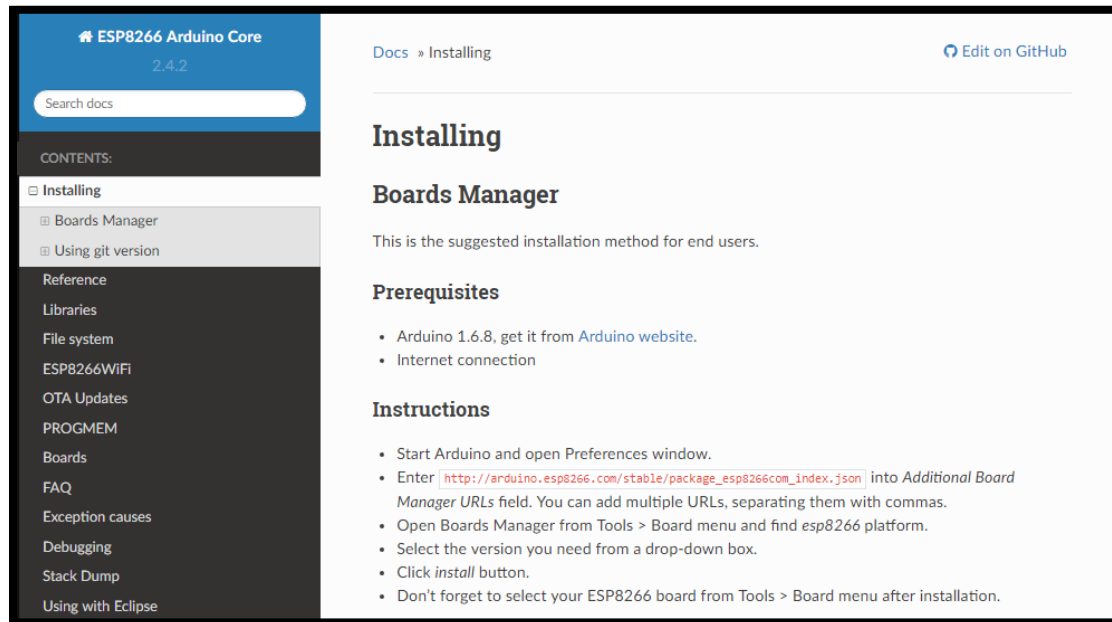
Basada enteramente en el ESP8266, para empezar a desarrollar la aplicación fue necesario instalar una actualización dentro del microcontrolador. Con esta actualización se utilizarán funciones ampliadas para cumplir con el cometido del proyecto. La configuración instalada se conoce como ESP Core.

Los pasos para realizar la actualización son dirigirte a la página web que contiene la documentación oficial. Una vez dentro, con el Arduino conectado y la IDE ejecutándose, en la sección de preferencias, Administración de tarjetas de desarrollo adicional. Se agrega la dirección del repositorio de la librería para obtener el instalador de la actualización.

Desde el menú, en Herramientas y después Tarjetas, seleccionamos el ESP8266. Se selecciona la versión y se instala desde este mismo menú.

Con esto, al seleccionar la tarjeta de desarrollo ESP8266, es posible ingresar código Arduino directo al microcontrolador.

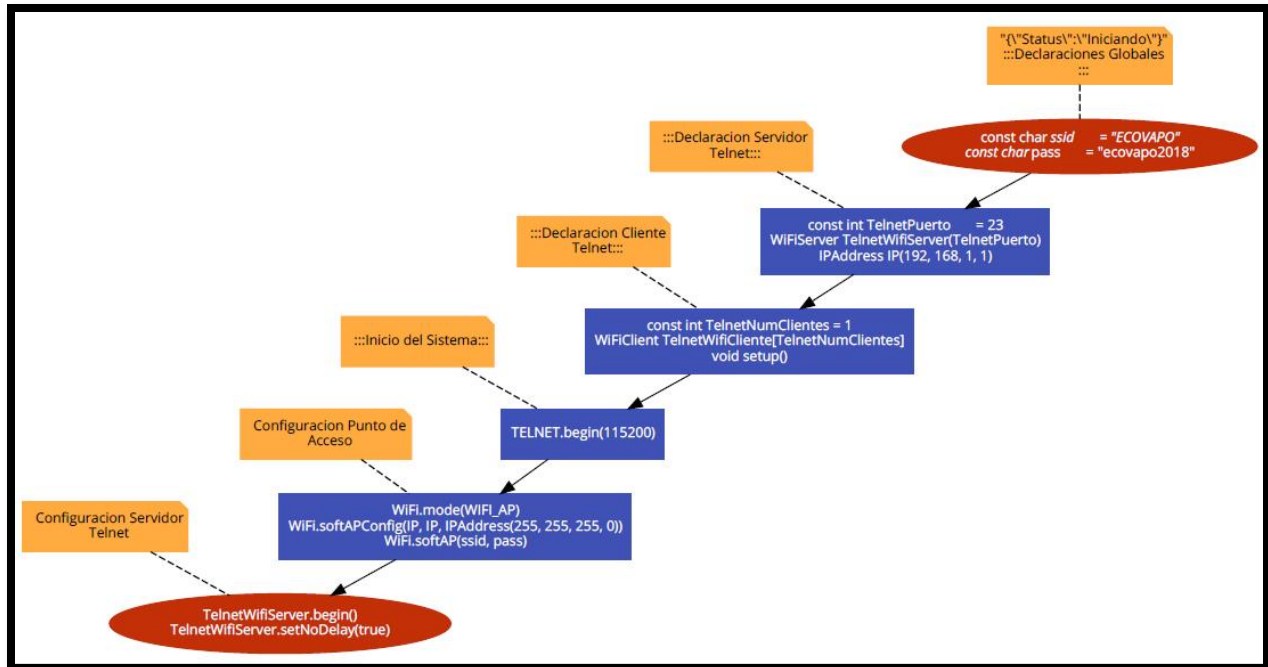
La librería principal utilizada es ESP8266WIFI, la cual se ha desarrollado basándose en ESP8266 SDK, utilizando la convención de nomenclatura y la filosofía de funcionalidad general de la biblioteca Arduino Wifi Shield.



Después de esto, ya es posible ingresar código en lenguaje Arduino al microprocesador dentro del ESP8266. Lo primero que se configuró es el punto de Acceso, para poder tener una línea de comunicación cerrada entre la Base Terrenal y el Submarino.

Al momento de importar las librerías necesarias para el funcionamiento de los módulos de WIFI, se declaran un par de variables las cuales contendrán el nombre de la red SSID y la contraseña para la conexión.

Se crea un tipo de dato especial para Servidor Wifi, así como un cliente Wifi. Con estas variables, se configura la red creada, agregándole el modo de trabajo, así como la dirección IP y la puerta de enlace predeterminada.



Los valores definidos como nombre de red y password, se envían como parámetros en una función la cual inicia el punto de acceso, rápidamente poniendo como visible la nueva red.



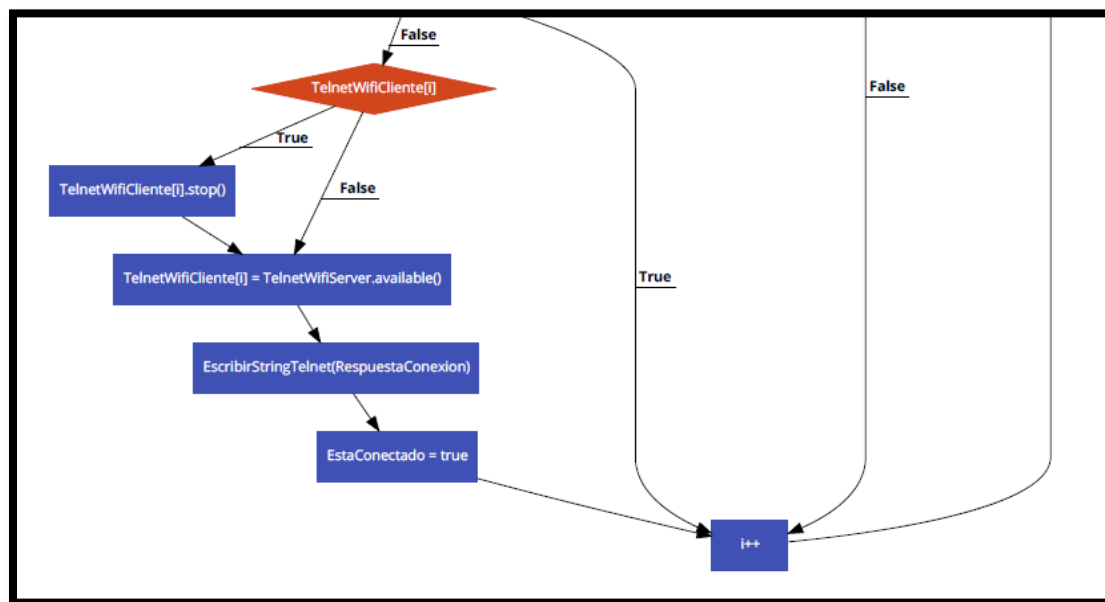
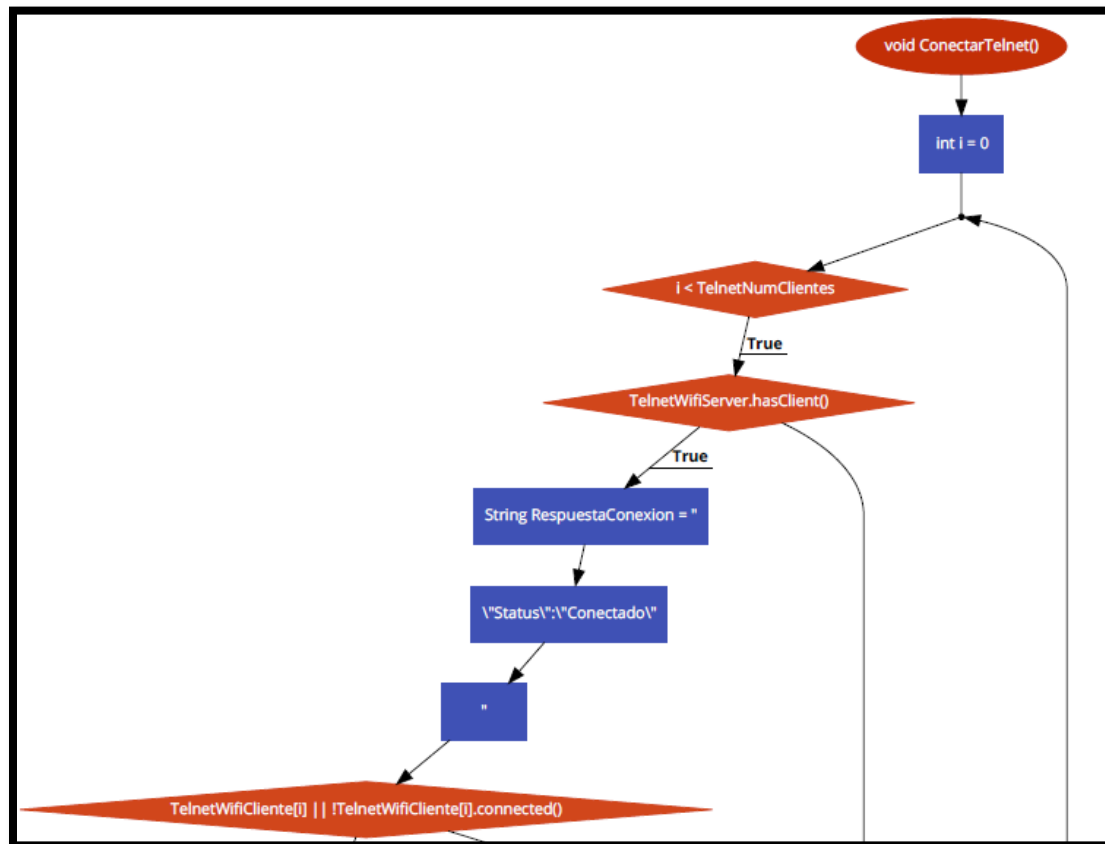
Servidor Telnet

Para poder recibir datos, se utilizará una conexión serial entre el ESP8266 y el Arduino. El principio del procedimiento dicta que al recibir algo dentro del puerto asignado para Telnet, la información pasará por comunicación serial, para ser visualizado y capturado dentro de Arduino.

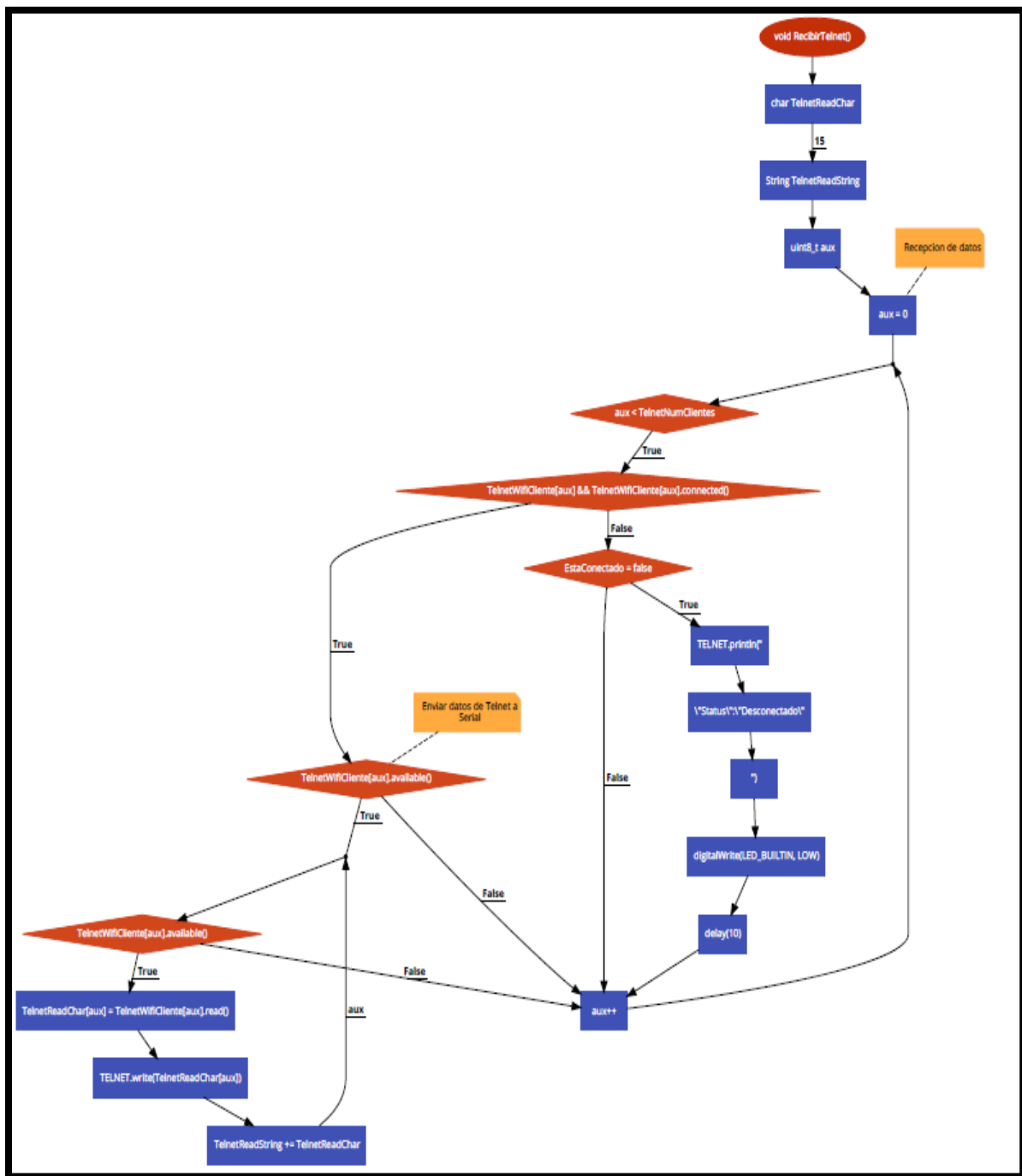
El esp8266 se encargará de enviar esta información tan pronto esté disponible. Para ello, dentro del servidor telnet, primeramente, se estableció el puerto 23 como entrada de datos.

Al inicio del programa, se cicla la función de conexión, la cual consiste en buscar dentro de las peticiones del servidor, un nuevo "cliente". Una vez encontrado, una bandera de conexión

cambia el flujo del código, para monitorear funciones de lectura y escritura de mensajes con el cliente conectado.

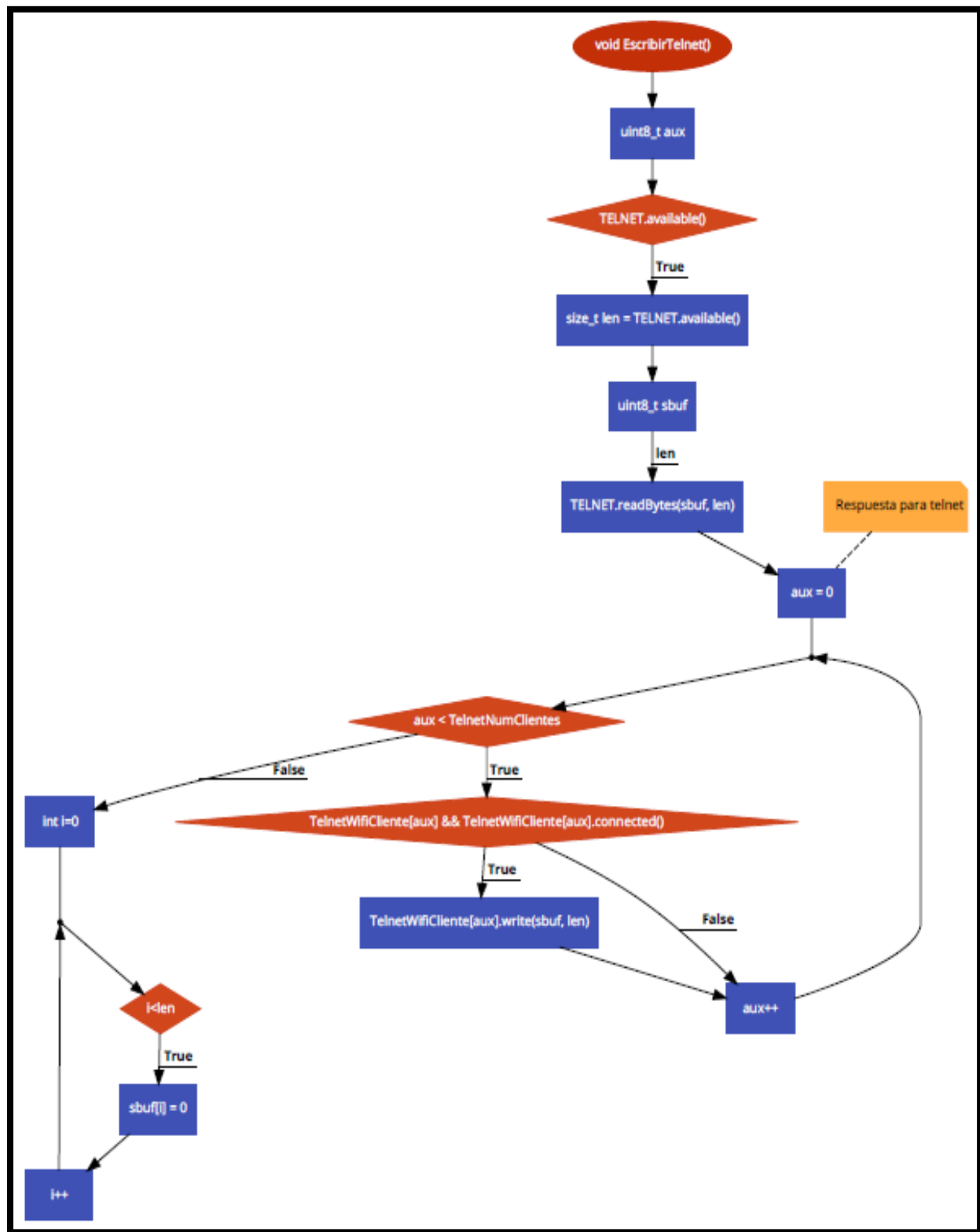


Para recibir un mensaje en el servidor, se recorría todo el mensaje recibido por serial, uno a uno, guardando en una cadena de caracteres, cada uno de los valores recibidos.



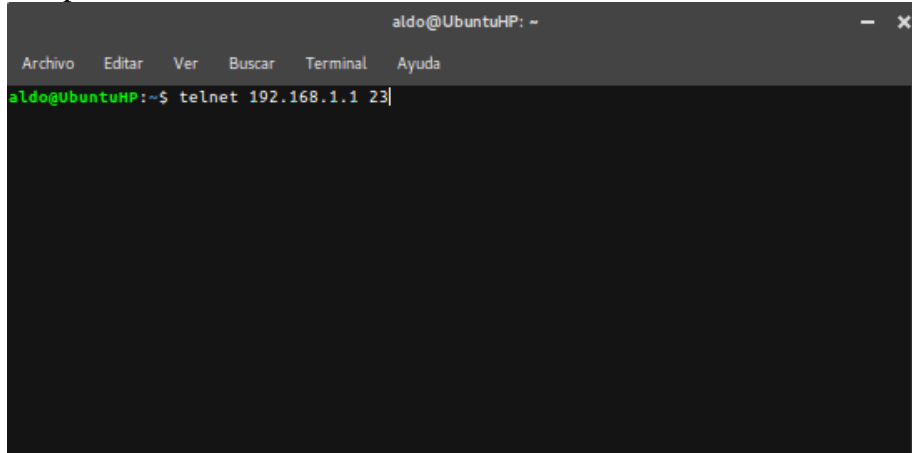
Para escribir un mensaje desde el "servidor" al cliente Telnet, que se desencadenaba cuando el puerto serial estaba disponible, se agregó a un buffer el mensaje en caracteres, se recorrió el buffer completo con un ciclo y uno a uno fueron enviados por medio de funciones de escritura

propias del ESP.



Una vez codificado el punto de acceso, es posible conectarte al servidor como cliente con el uso de una consola de comandos. Enviando telnet con la IP y el puerto, se recibe un mensaje de

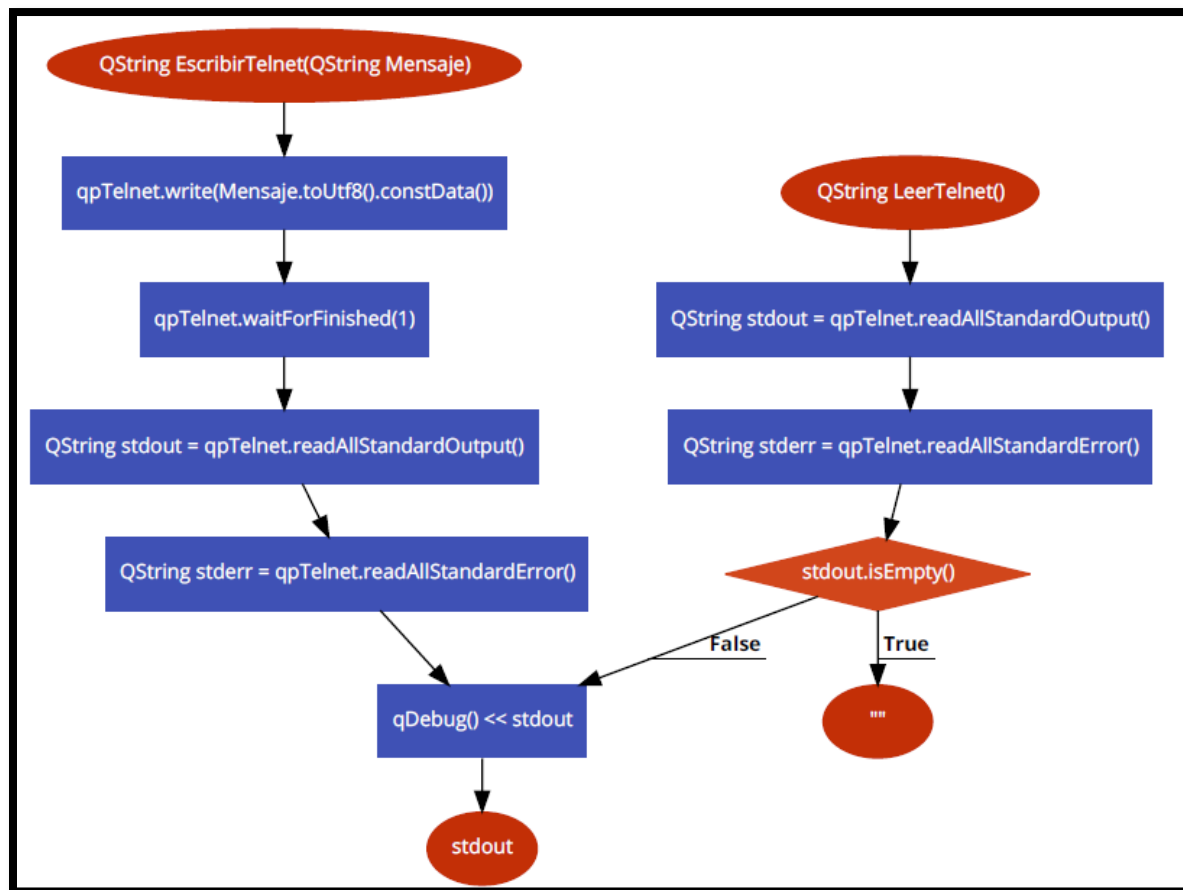
confirmación de enlace Telnet. Con esto, ya es posible realizar el envío de datos simultáneos, en tiempo real.



```
aldo@UbuntuHP: ~  
Archivo  Editar  Ver  Buscar  Terminal  Ayuda  
aldo@UbuntuHP:~$ telnet 192.168.1.1 23
```

Cliente QT Control Remoto

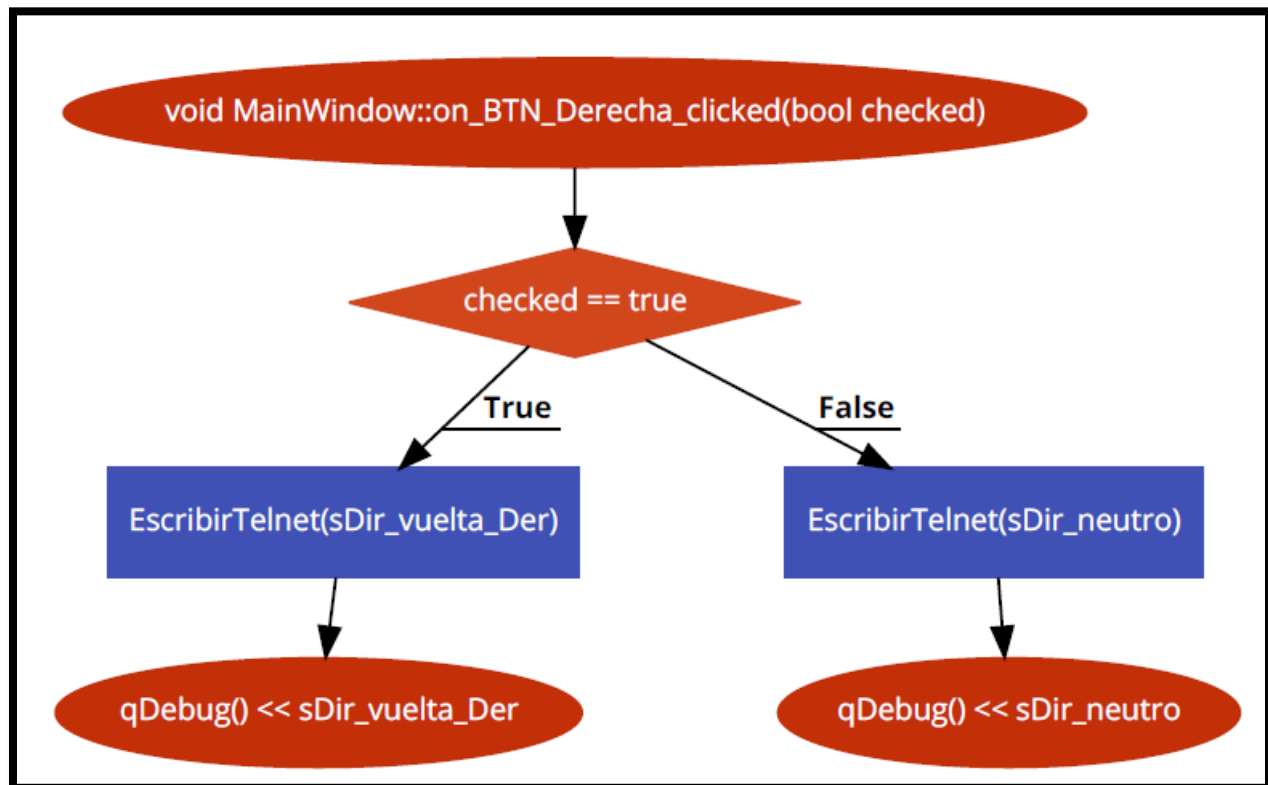
Una vez levantado el servidor, se desarrolló un cliente Telnet, para funcionar como control remoto. El funcionamiento a grandes rasgos es realizar una conexión a la red del servidor, el cual será el submarino. Con esta conexión, se enlazan los mensajes que funcionarán como comandos de control remoto.



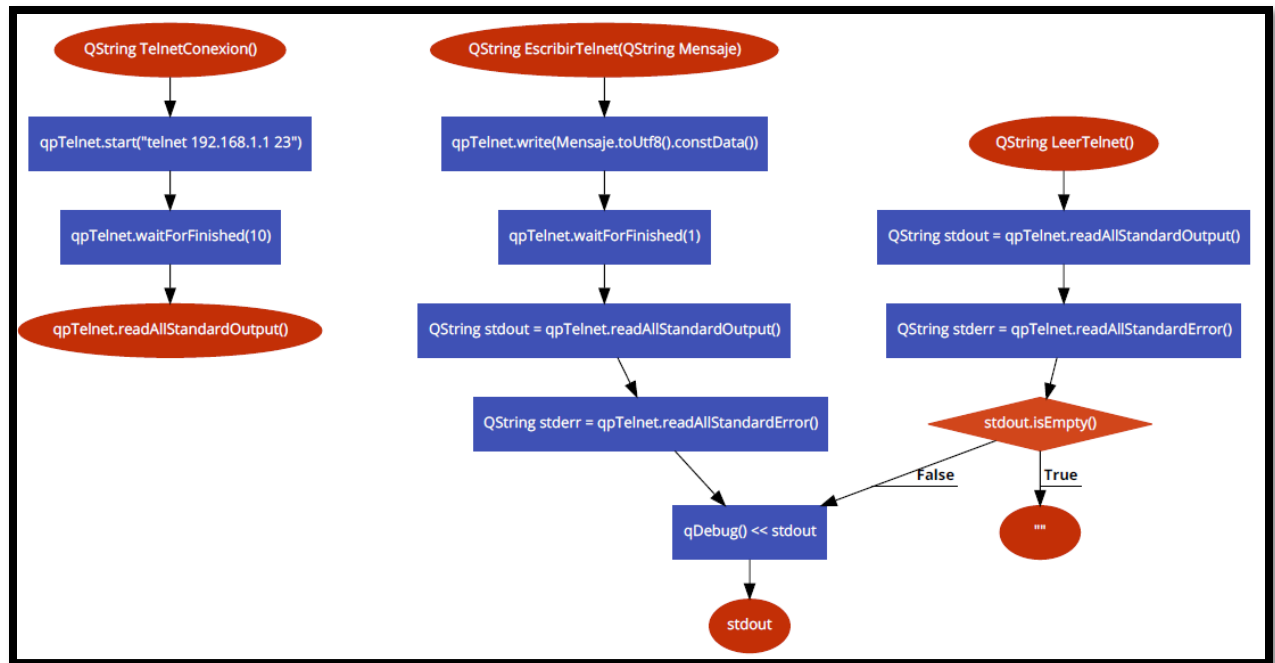
El desarrollo del control remoto se realizó en Qt, en donde la interfaz principal consiste en botones de dirección, una barra de aceleración y una ventana de visualización de la cámara, principalmente.

Cada botón tiene configurado un mensaje en formato JSON el cual se envía al detectar el evento del click del mismo, así como los cambios de valores de la barra de aceleración.

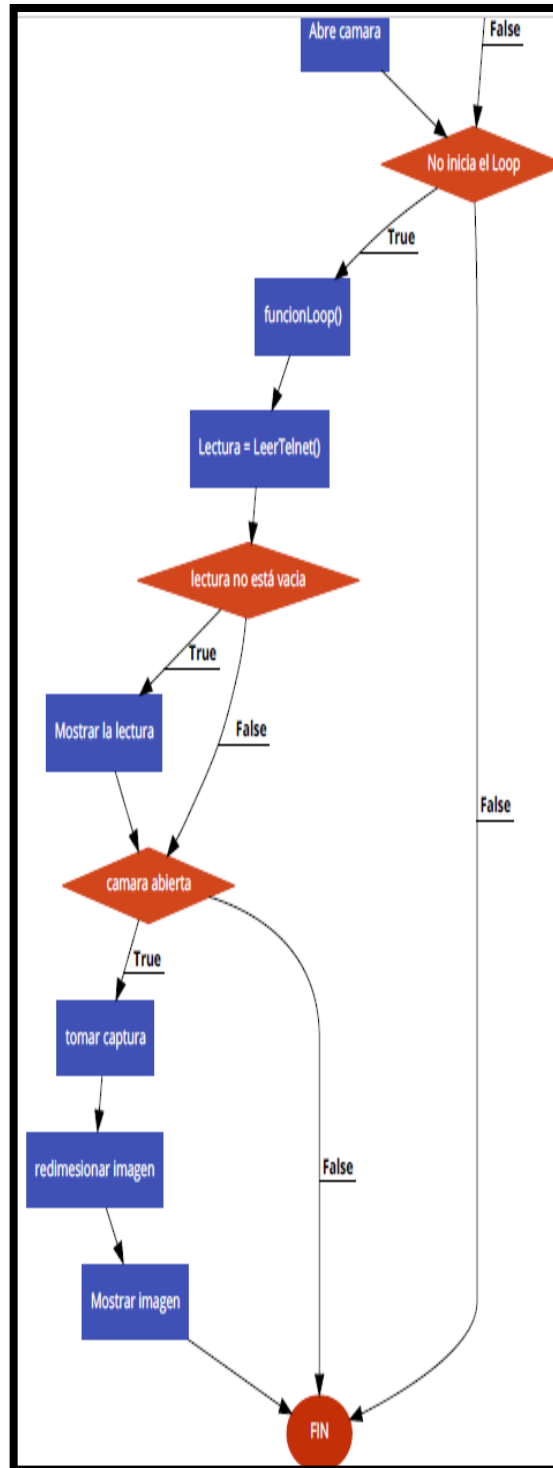
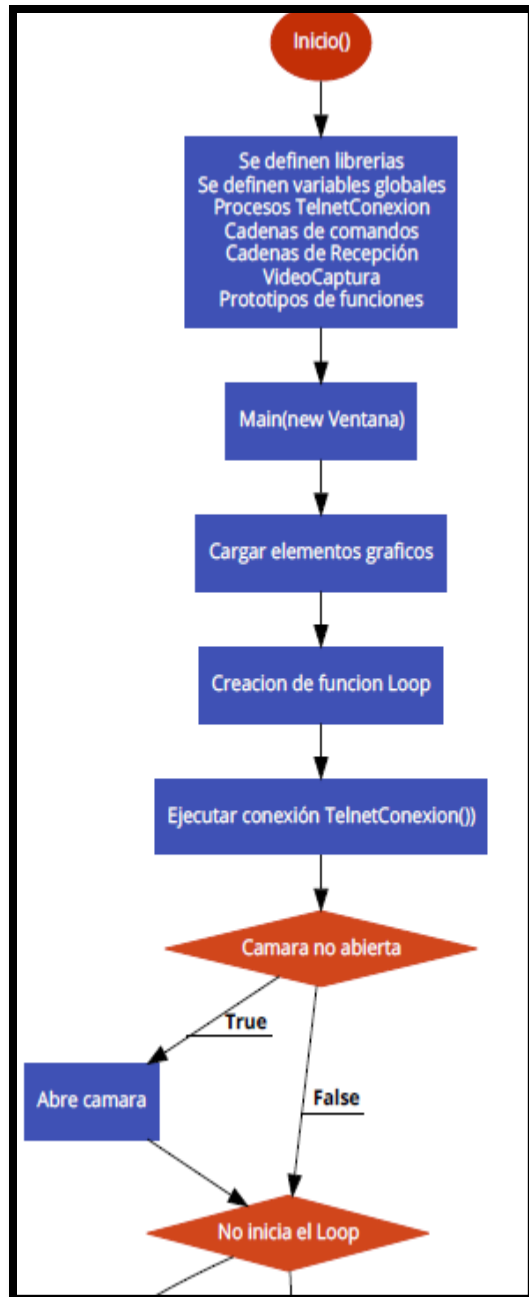
La conexión al servidor se hace con un QProcess, el cual realiza las funciones de una terminal de Linux. Con ello, se envía el comando de conexión y automáticamente se enlaza el servidor, una vez conectado al punto de acceso.



Las funciones de envío y recepción de datos, se realizan desde el mismo QProcess, el cual lee y envía datos desde la misma terminal encapsulada en la función.

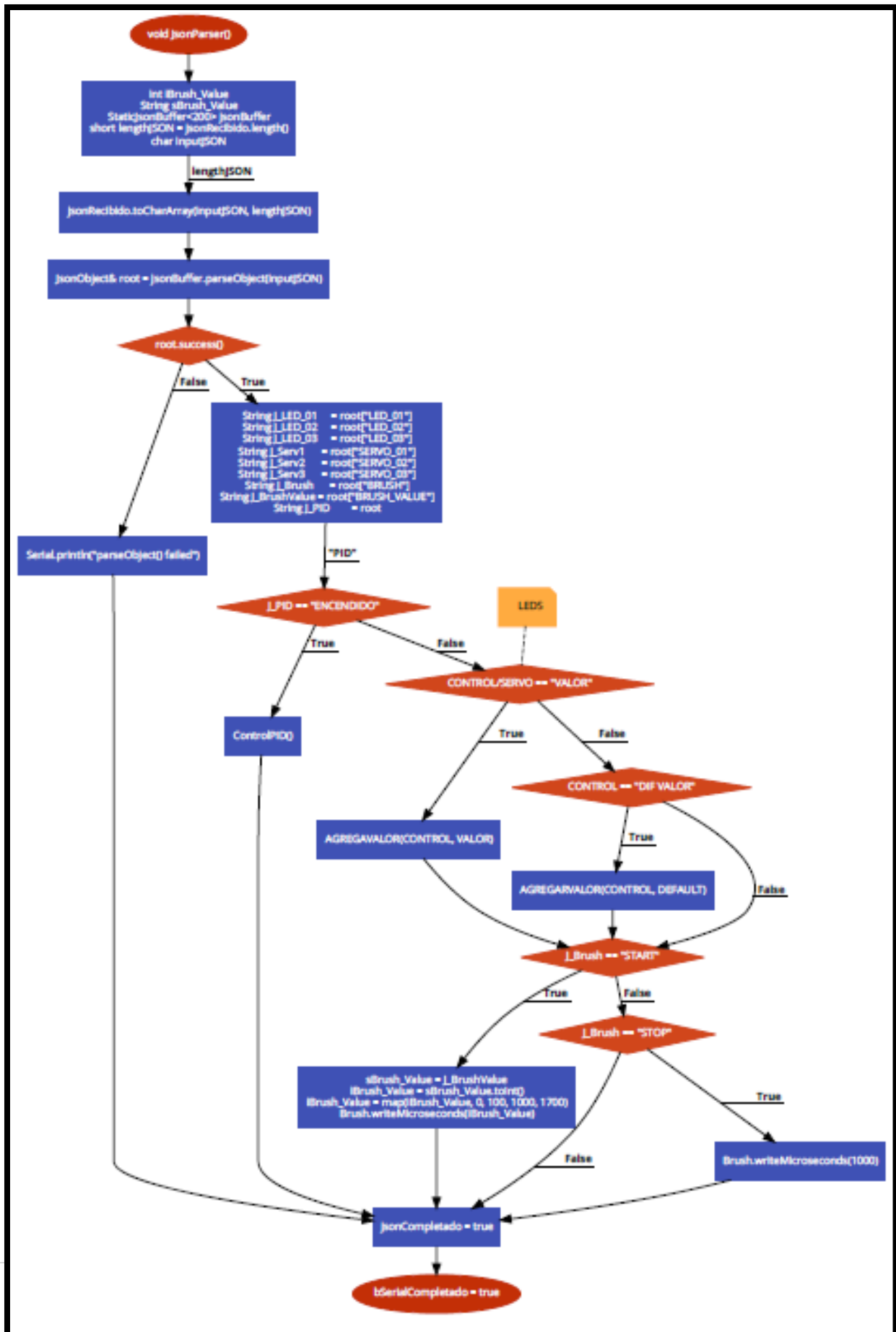


Desde esta parte del proyecto, también se recibe en tiempo real la imagen desde la cámara. Dicha imagen es capturada 30 veces por segundo, lo cual genera un flujo de imágenes como video. La información de la cámara es almacenada en variables para llevar a cabo un pequeño proceso de mapeo de bits y redimensión de datos.



JSON

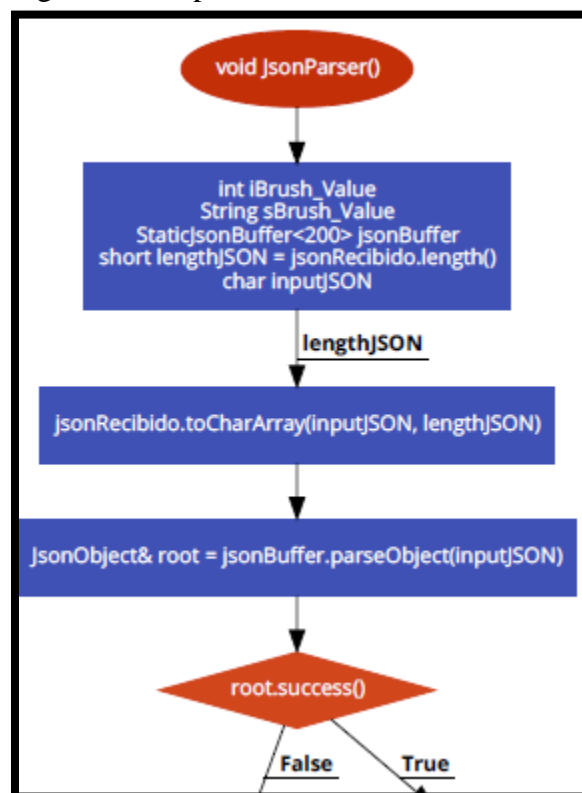
Se eligió JSON, debido a la facilidad con la que se pueden enviar ráfagas de datos para realizar los movimientos de los motores. Debido a la misma naturaleza del objeto, se guardaron en forma de cadena. Entre los motivos por los que se hizo así es, la facilidad para enviar por un buffer, recibir detectando el final de la misma, representado por el carácter especial '\n', ligero en datos, veloz al viajar por el servidor.

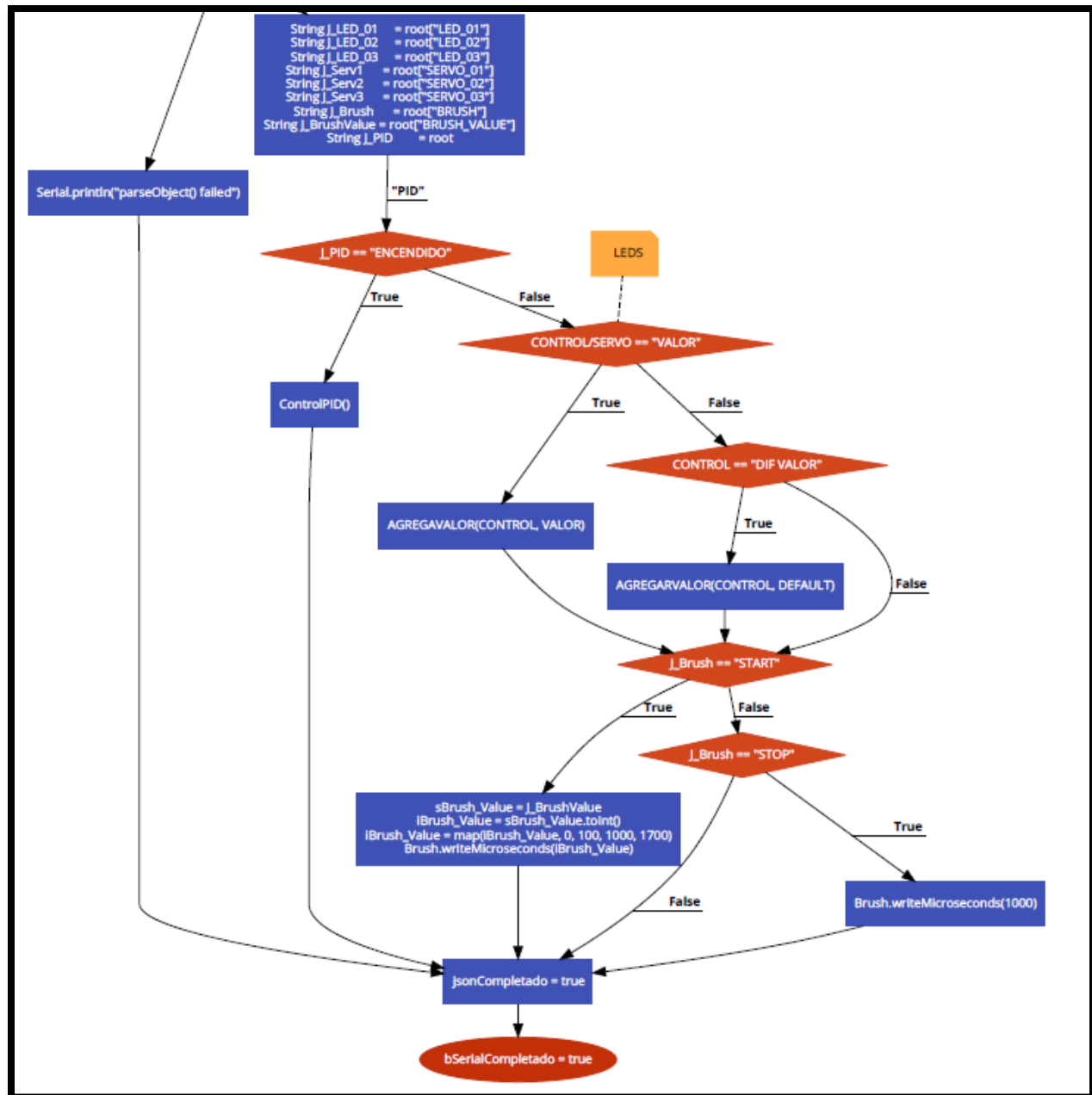


Dentro de Arduino, con la librería de JSONParser es posible recibirlos directamente del servidor, y obtener cada uno de los valores encapsulados. Al llegar un mensaje, se activa un evento serial el cual realiza la lectura de cada objeto JSON.

```
QString sLectura;
QString sDir_adelante = "{\"SERVO_01\": \"90\", \"SERVO_02\": \"90\", \"SERVO_03\": \"90\"}\n";
QString sDir_atras = "{\"SERVO_01\": \"90\", \"SERVO_02\": \"90\", \"SERVO_03\": \"90\"}\n";
QString sDir_vuelta_Izq = "{\"SERVO_01\": \"180\", \"SERVO_02\": \"90\", \"SERVO_03\": \"180\"}\n";
QString sDir_vuelta_Der = "{\"SERVO_01\": \"90\", \"SERVO_02\": \"0\", \"SERVO_03\": \"0\"}\n";
QString sDir_neutro = "{\"SERVO_01\": \"90\", \"SERVO_02\": \"90\", \"SERVO_03\": \"90\"}\n";
QString sAce_Brushless;
QString sPid_ControlOn = "{\"PID\": \"ENCENDIDO\"}\n";
QString sPid_ControlOff = "{\"PID\": \"APAGADO\"}\n";
```

Qt se encarga igualmente de enviar por consola el mensaje, agregando los finales de cadena manualmente. Con eso, se ingresan valores dependiendo de los botones que se opriman. Qt se encarga de mostrar una interfaz virtual de comandos, los cuales muestran el mando de control del submarino, así como la imagen tomada por la cámara a bordo.



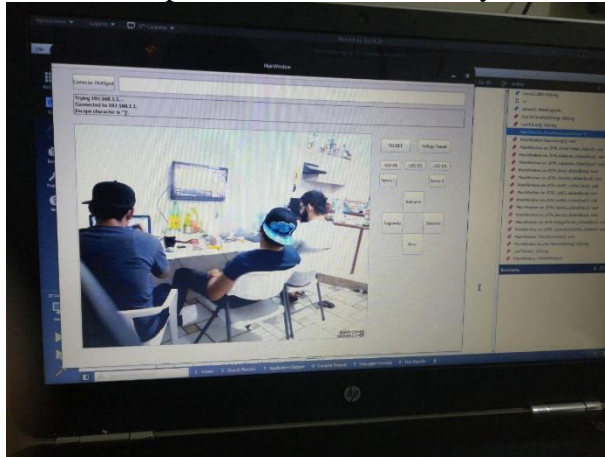


Servidor de Stream de Video

Para la transmisión de video en tiempo real, se optó por utilizar una librería de Raspberry PI conocido como Motion, el cual se utiliza principalmente para detectar movimientos dentro de un rango de visión de una o varias cámaras. Su forma de detectar movimiento, es cuando hay variación sustancial entre las imágenes que captura. Dicha librería se distribuye con licencia GNU GPL.

Para la instalación de Motion en Raspberry Pi, basta con escribir un comando en Linux solicitando los archivos del repositorio oficial. Seguido a eso, se instala un paquete de controladores de video. Ahora se conecta la cámara, y confirma que esté correctamente conectada.

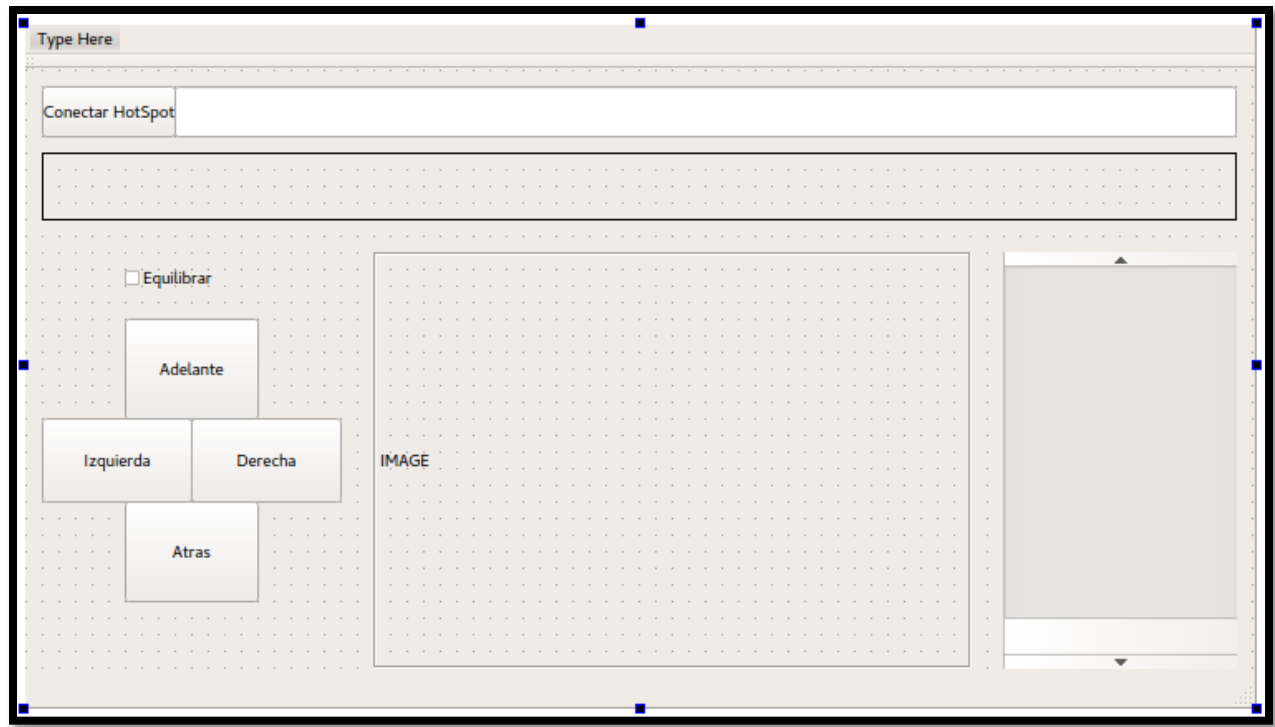
Una vez instalado en Raspberry Pi, el servidor motion se configura para las necesidades del proyecto. Lo anterior se realiza editando el archivo motion.config, el cual contiene todas las configuraciones de la librería. Entre los cambios que ejecutamos son la cantidad de cuadros por segundo, así como la calidad de la imagen y el retardo entre cuadros para mejorar la transmisión. El siguiente paso es controlar la ejecución del servicio de Motion, el cual inicia por default al inicio del sistema, después de reiniciar. De igual forma, desde Linux ejecutando una línea de comando es posible reiniciar, iniciar y detener el servicio de Motion.



Para comprobar que el servicio se encuentra correctamente iniciado, al ingresar a través de un navegador a la IP asignada a la Raspberry, por el puerto especificado, será posible monitorear la cámara conectada, transmitiendo en tiempo real.

Navegación

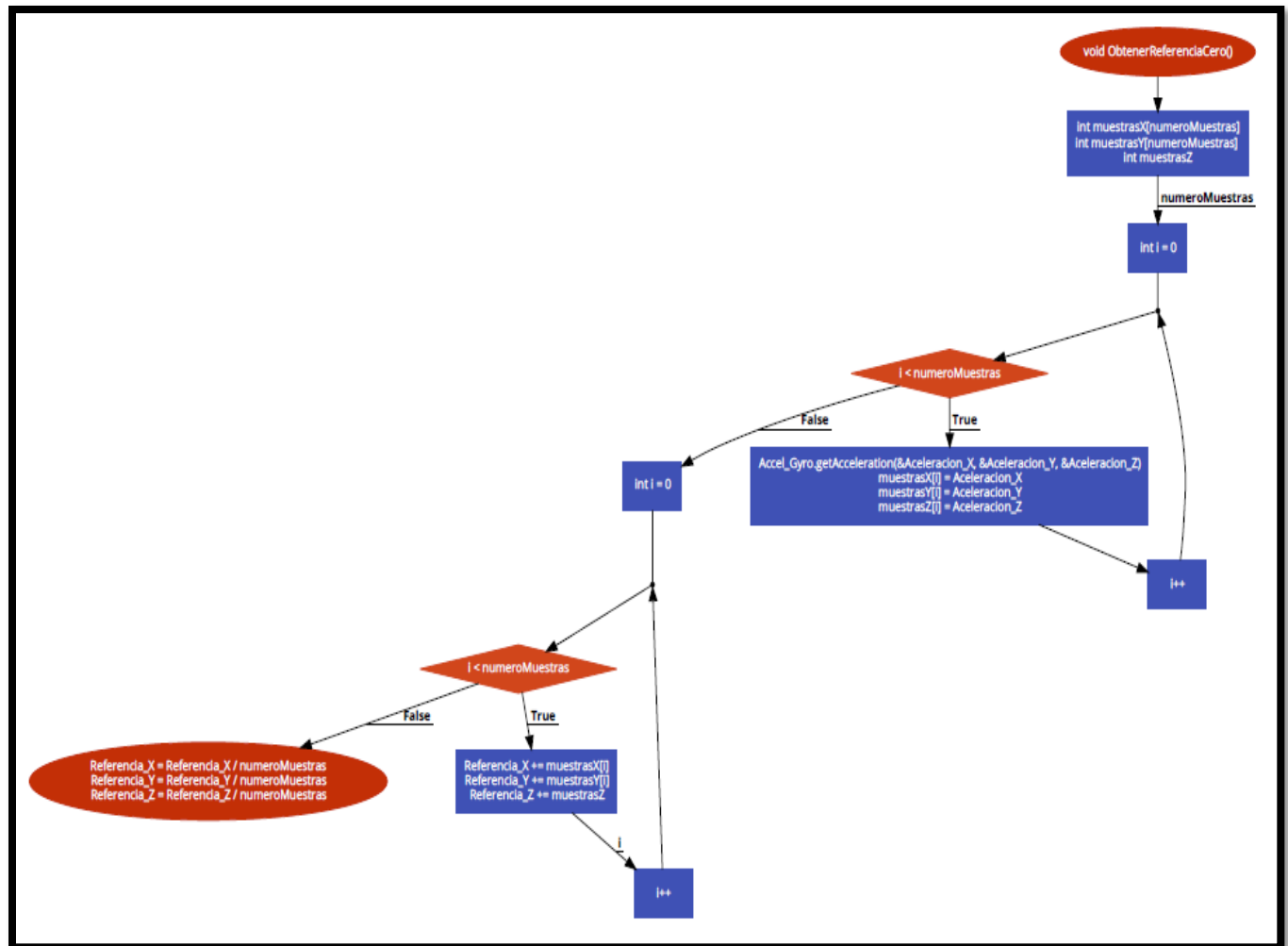
El movimiento del ROV será en 3 dimensiones. El sistema de propulsión se encargará de acelerar en ambos sentidos y de manera independiente. Los servomotores, utilizados para posicionamiento de los motores también serán independientes. La velocidad de los propulsores será controlable manualmente con ayuda del sistema de control. La parte posterior del ROV será transparente para poder obtener una imagen lo más clara posible del medio donde se navega. El control automático se llevará a cabo por medio de una función PID.



Control PID

Por medio de un código Arduino se logró llevar a cabo el mismo, empleándolo a partir de las lecturas recibidas por el MPU6050. La implementación del control PID dentro del Arduino, tiene como inicio los parámetros de entrada obtenidos de las lecturas en crudo que arroja el MPU6050. Sin embargo, dado que la función del PID toma en cuenta el tiempo, el inicio del algoritmo involucra una variable para retener el tiempo actual, así como una para almacenar el tiempo de la ejecución anterior. Con estas dos variables, por diferencia es posible obtener el tiempo transcurrido en la ejecución.

El primer paso, consiste en obtener la referencia espacial del submarino. Esto se logró con una función que obtiene 1000 muestras de los valores de los 3 ejes en crudo, los cuales son promediados para obtener el valor que correspondería a la posición actual del prototipo, con esto, las variaciones obtenidas cambian en razón de este valor, controlando así cualquier posición que se coloque como el punto de partida de la ejecución.

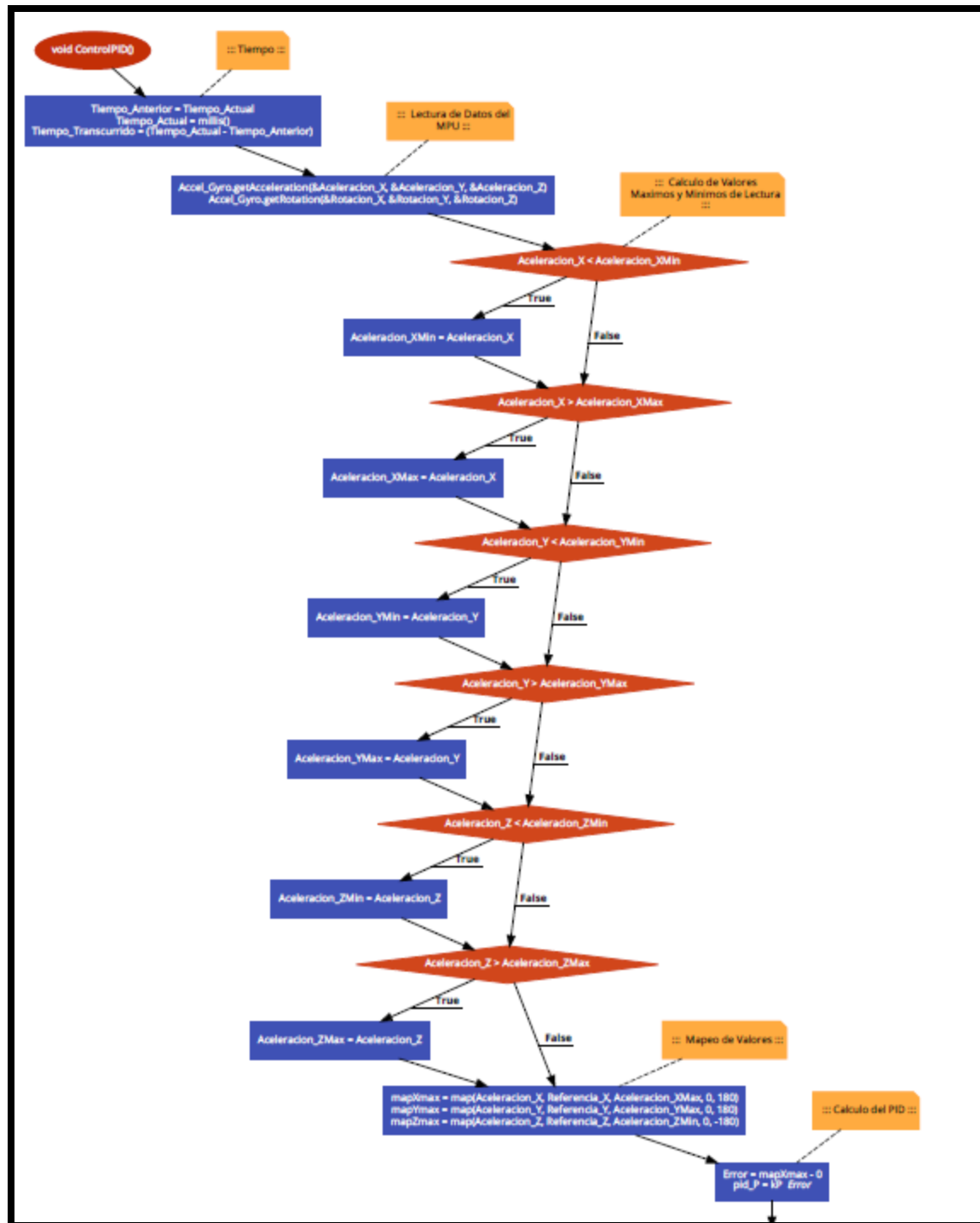


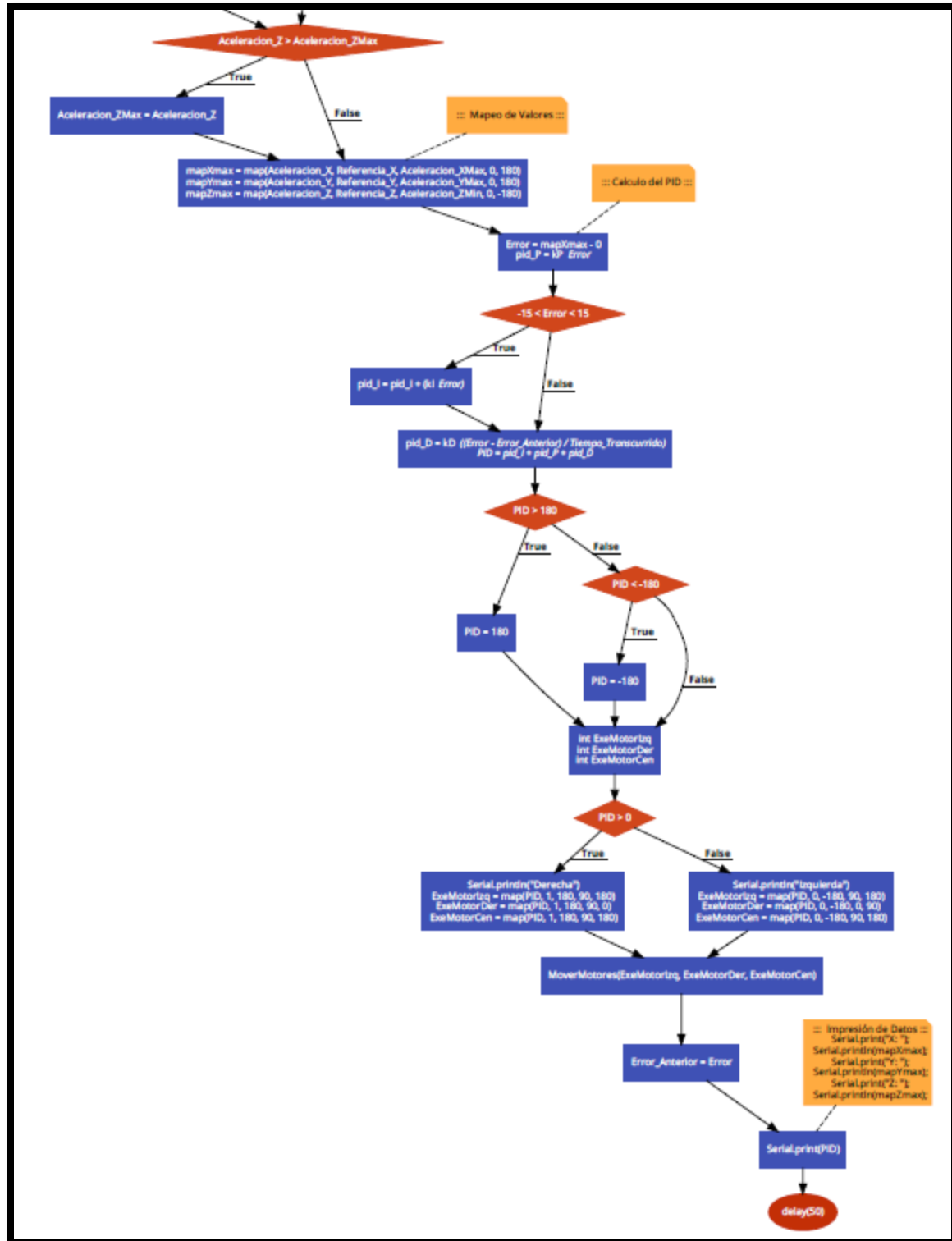
El siguiente paso es leer los valores de los giroscopios y acelerómetros de forma cruda (sin procesar). Una vez obtenidos, en cada iteración se guarda el valor máximo y mínimo arrojado por los mismos, a través de una serie de condicionales anidadas.

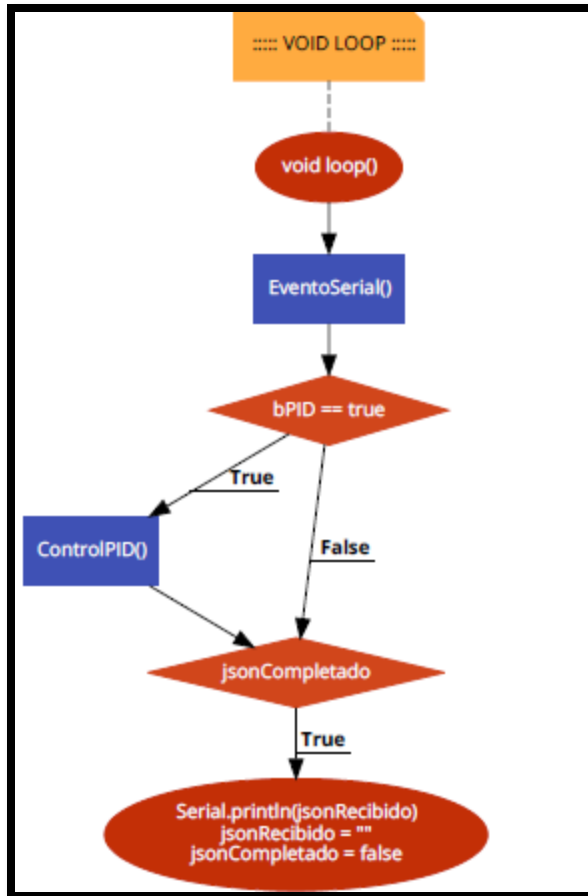
Una vez obtenidas las lecturas se mapean los valores, para igualar los valores a un rango de movimiento en grados manejable. Con esto, las lecturas mínimas arrojadas, serán a su vez los movimientos y giros hacia un Angulo negativo y viceversa.

Se calcula el error de la lectura del movimiento, restando de la referencia, el valor obtenido ya mapeado. Con este valor, se sabe cuántos grados se encuentra desfasado el sensor en cuanto a la posición inicial que se desea.

El primer calculo que se realiza es el proporcional, el cual multiplica una constante definida, por el error obtenido, con la finalidad de aumentar la resolución de la lectura. Después, dependiendo si el error es proporcionalmente pequeño, se calcula la parte integral, que afinará la señal para obtener un acercamiento más exacto, y después la constante derivativa se multiplicará por la resta de los errores, tanto del actual como del anterior, divididos entre el tiempo que ha transcurrido.







La suma de estos 3 cálculos nos dará los movimientos necesarios que los motores deben realizar, para llevar el submarino a su estado de referencia.

El programa, al iniciar, toma como estado de referencia esa posición, a partir de ahí se calculan los movimientos. Para ejecutarlos, una variable mapea las lecturas obtenidas, en los grados que se busca puedan moverse los actuadores, para ejecutar una función que escribe una señal pwm correspondiente.

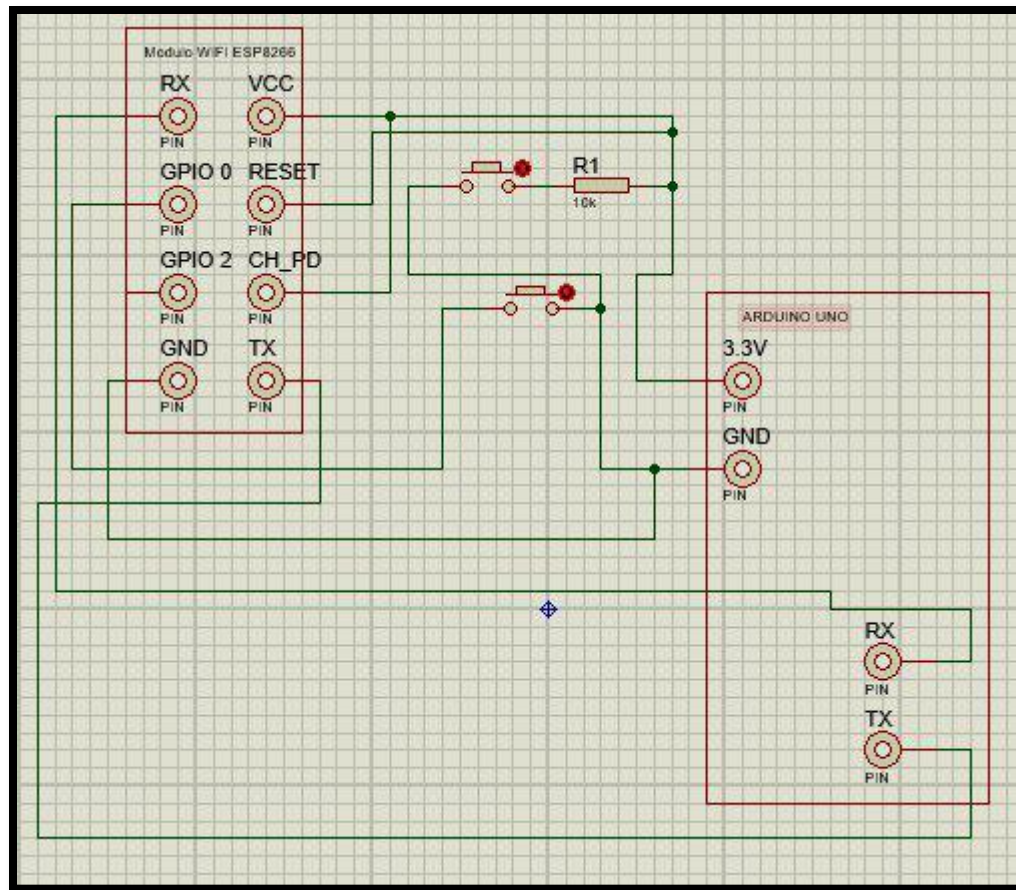
Antes de finalizar el programa, se vuelve a calcular el error del ciclo, para iniciar nuevamente con un valor actualizado.

PCB

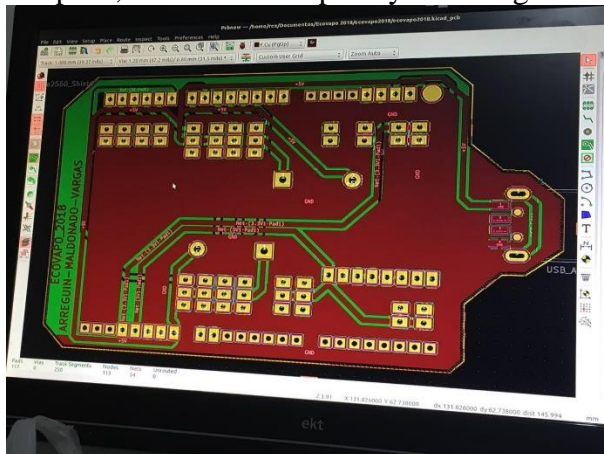
La circuitería del prototipo fue encapsulada dentro de una tarjeta de circuito impreso, fabricada de manera artesanal. En ella se colocaron los componentes del proyecto, los cuales necesitan un soporte personalizado para ser compatibles con el Arduino.

Partiendo de un diseño esquemático, se realizó la planeación y la distribución de los componentes, los cuales optimizan el espacio dentro de la tarjeta, además de que su colocación necesita un soporte para las conexiones.

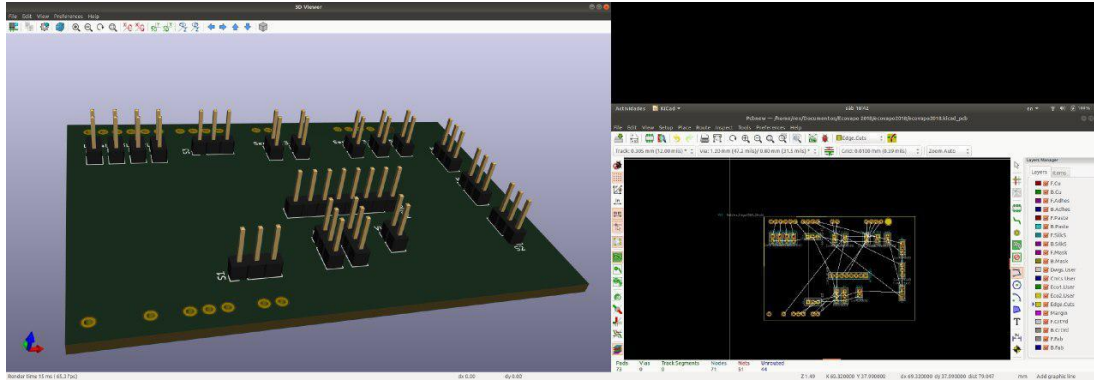
La fabricación de la tarjeta se llevó a cabo con un proceso artesanal, el cual consiste en imprimir una plantilla conocida como "footprint" en una hoja de papel especial con un recubrimiento plástico, el cual al ser sometido al calor desprende su misma imagen grabada en tinta. Una vez fabricada esta impresión, se coloca sobre de una placa de cobre para baquelita, de doble cara, y se plancha a temperatura media por un cuarto de hora aproximadamente.



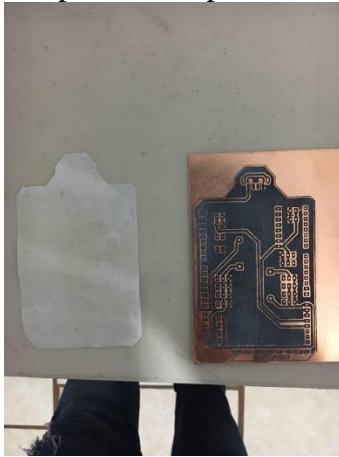
El diseño de la PCB se realizó a través del software libre utilizando la herramienta de KiCad Schematic Layout Editor en donde se realizó el diagrama eléctrico. En este paso se agrega el Arduino Mega, así como los pines de conexión para los dispositivos y se asignan los footprint. Después, se routea cada pista y se le asignan capas al diseño de la PCB.



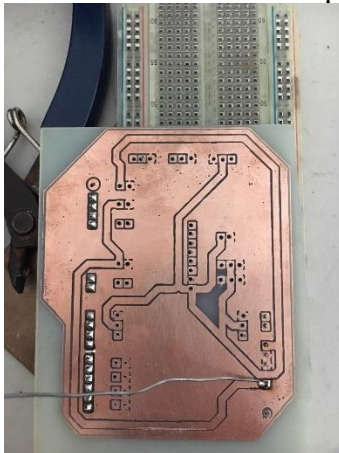
Después de este proceso, se sumerge la tarjeta completa en ácido férrico, el cual se encarga de retirar todo el cobre de la baquelita, sin ocasionar daños en las partes donde la tinta ha sido adherida al cobre, debido al proceso de planchado.



Los componentes utilizados, tienen en común el juego de entradas que consiste principalmente en las dos entradas de las terminales de alimentación, y una salida de señal. El sensor MPU al no ser compatible con ninguna tarjeta por la colocación de sus pines, la PCB se diseñó completamente partiendo de las necesidades de este mismo.



En la cara superior de la tarjeta impresa, se colocaron las entradas de alimentación, los pines y puertos del ESP8266, MPU6050, Servo Motores y motor Brushless. En esta misma capa, también se diseñaron rutas con un calibre de 2mm para asegurar la distribución de energía en el sistema, evitar pérdidas o tener un corto circuito. Para el ruteo de las señales utilizamos la parte inferior con un ancho de pista 1.8mm para la transmisión de señal.



Conclusión

Debido a distintas experiencias documentadas en video y en blogs relacionadas al tema pudimos concluir que la radio comunicación bajo el agua no es una opción atractiva para controlar el submarino ya que requiere frecuencias de operación muy bajas o grandes sistemas de antenas. Eso sumado a la inexperiencia del equipo confirma la selección de comunicación por medio físicos.

Este trabajo tiene como resultado el cumplimiento de los objetivos que fueron marcados con antelación. Dicho proyecto fue bastante interesante dado las diferentes técnicas que se tuvieron que aprender y/o desarrollar para lograr que el sistema funcione. A lo largo del desarrollo pudimos percatarnos de la importancia que tiene el dimensionar el alcance del proyecto; puesto que se deben tomar en cuenta aspectos tales como: costos, mecanismos, comunicación, control y tiempos dispuestos para alcanzar los objetivos propuestos.

A través de la elaboración de diferentes prototipos en la primera etapa se logra determinar la forma adecuada, equipamiento y los adecuados materiales, esto permitió corregir las deficiencias que se tuvieron en versiones anteriores.

El diseño y elaboración del prototipo acuático no fueron tareas sencillas, debido a la poca experiencia que se tenía en el diseño de este tipo de sistemas y al elevado costo que tienen los materiales propios para la elaboración de un submarino.

El trabajar en un ambiente diferente al terrestre como lo es el acuático, trae desventajas, la principal es la comunicación entre el prototipo y el pc a través del módulo bluetooth debido a la baja potencia

La hermeticidad en este tipo de robot juega un papel muy importante, puesto que se necesita asegurar que no ingrese el agua y dañe los elementos electrónicos.

Referencias Bibliográficas

Anexo 01 – Código Servidor Telnet + Punto de Acceso + Serial Parser

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#define TELNET Serial

// "{\"Status\":\"Iniciando\"}"

/* ::Declaraciones Globales :: */
const char* ssid      = "ECOVAPO";
const char* pass       = "ecovapo2018";

bool EstaConectado = false;

/* ::Declaracion Servidor Telnet:: */
const int TelnetPuerto = 23;
WiFiServer TelnetWifiServer(TelnetPuerto);
IPAddress IP(192, 168, 1, 1);

/* ::Declaracion Cliente Telnet:: */
const int TelnetNumClientes = 1;
WiFiClient TelnetWifiCliente[TelnetNumClientes];

void ConectarTelnet();
void EscribirStringTelnet();
void ConfigurarTelnet();
void RecibirTelnet();
void EscribirTelnet();

void setup() {
    ConfigurarTelnet();
}

void loop() {
    ConectarTelnet();
    RecibirTelnet();
    EscribirTelnet();
}

void ConfigurarTelnet() {
    /* ::Inicio del Sistema:: */
    TELNET.begin(115200);
    TELNET.print("{\"Status\":\"Iniciando\"}");

    pinMode(LED_BUILTIN, OUTPUT);

    /* Configuración Punto de Acceso */
    TELNET.println("{\"Status\":\"Iniciando Punto de Acceso\"}");
    WiFi.mode(WIFI_AP);
    WiFi.softAPConfig(IP, IP, IPAddress(255, 255, 255, 0));
    WiFi.softAP(ssid, pass);

    /* Configuración Servidor Telnet */
    TelnetWifiServer.begin();
    TelnetWifiServer.setNoDelay(true);

    /* Finalización de Configuración */
    TELNET.print("{\"Status\":\"Iniciado\"}");
}

void EscribirTelnet() {
    uint8_t aux;
    if (TELNET.available()) {
        size_t len = TELNET.available();
        uint8_t sbuf[len];
        TELNET.readBytes(sbuf, len);

        /* Respuesta para telnet */
        for (aux = 0; aux < TelnetNumClientes; aux++) {
            if (TelnetWifiCliente[aux] && TelnetWifiCliente[aux].connected()) {
                TelnetWifiCliente[aux].write(sbuf, len);
            }
        }
        for (int i=0; i<len; i++){
            sbuf[i] = 0;
        }
    }
}
```

```

    else {
        return;
    }
}
void RecibirTelnet() {
    char TelnetReadChar[15];
    String TelnetReadString;
    uint8_t aux;
    /* Recepcion de datos */
    for (aux = 0; aux < TelnetNumClientes; aux++) {
        if (TelnetwifiCliente[aux] && TelnetwifiCliente[aux].connected()) {

            /* Enviar datos de Telnet a Serial */
            if (TelnetwifiCliente[aux].available()) {
                while (TelnetwifiCliente[aux].available()) {
                    TelnetReadChar[aux] = TelnetwifiCliente[aux].read();
                    TELNET.write(TelnetReadChar[aux]);
                    TelnetReadString += TelnetReadChar[aux];
                }

            }

        } else {
            if (EstaConectado = false) {
                TELNET.println("{\\\"Status\\\":\\\"Desconectado\\\"}");
                digitalWrite(LED_BUILTIN, LOW);
                delay(10);
            }
        }
    }
    return;
}
void ConectarTelnet() {
    for (int i = 0; i < TelnetNumClientes; i++) {
        if (TelnetwifiServer.hasClient()) {
            String RespuestaConexion = "{\\\"Status\\\":\\\"Conectado\\\"}";

            if (!TelnetwifiCliente[i] || !TelnetwifiCliente[i].connected()) {
                if (TelnetwifiCliente[i]) {
                    TelnetwifiCliente[i].stop();
                }

                TelnetwifiCliente[i] = TelnetwifiServer.available();
                EscribirStringTelnet(RespuestaConexion);
                EstaConectado = true;
            }
        }
    }
    return;
}

void EscribirStringTelnet(String Mensaje) {
    size_t t = Mensaje.length() + 1;
    uint8_t buff[t];

    Mensaje.getBytes(buff, t);

    for (int i = 0; i < TelnetNumClientes; i++) {
        if (TelnetwifiCliente[i] && TelnetwifiCliente[i].connected()) {
            TelnetwifiCliente[i].write(buff, t);
        }
        else {
            return;
        }
    }
}

```

Anexo 02 – Código Arduino Serial Event + JSON Parser + Control PID + Control Brushless Motor y Servomotor

```
#define SERIAL_BAUD    115200
#define STRING_LENGTH  200
#define BRUSH_STARTV   1000
#define SERVO_IZQ      6
#define SERVO_CEN      7
#define SERVO_DER      8
#define BRUSHLESS      5
#define LED_01         13
#define LED_02         12
#define LED_03         11

// ::::: Bibliotecas :::::

#include "Servo.h"
#include "ArduinoJson.h"
#include "Wire.h"
#include "I2Cdev.h"
#include "MPU6050.h"

// ::::: Variables Globales :::::

Servo ServoIzq;
Servo ServoDer;
Servo ServoCen;
Servo Brush;

short int ServoIzq_Valor;
short int ServoDer_Valor;
short int ServoCen_Valor;
short int Brushless_Valor;

String jsonRecibido;

bool jsonCompletado;
bool bSerialCompletado;
bool bPID;

// ::::: Variables Globales MPU6050 :::::

MPU6050 Accel_Gyro;

int Aceleracion_X, Aceleracion_Y, Aceleracion_Z;
int Aceleracion_XMin, Aceleracion_YMin, Aceleracion_ZMin;
int Aceleracion_XMax, Aceleracion_YMax, Aceleracion_ZMax;

int Rotacion_X, Rotacion_Y, Rotacion_Z;
int rotMinX, rotMinY, rotMinZ;
int rotMaxX, rotMaxY, rotMaxZ;

float Angulo_X, Angulo_Y;

int mapAceleracion_X, mapAceleracion_Y, mapAceleracion_Z;
int mapRotacion_X, mapRotacion_Y, mapRotacion_Z;
int mapXmin, mapXmax;
int mapYmin, mapYmax;
int mapZmin, mapZmax;

long Referencia_X = 0;
long Referencia_Y = 0;
long Referencia_Z = 0;

long Error = 0;
long Error_Anterior = 0;

float pid_P = 0;
float pid_I = 0;
float pid_D = 0;

double kP = 2.5;
double kI = 0.005;
double kD = 3.2;

float Tiempo_Actual;
float Tiempo_Transcurrido;
```



```

float Tiempo_Anterior;

float PID;

int numeroMuestras = 1000;

// ::::: Funciones Globales Prototipos :::::

void EventoSerial();
void JsonParser();
void ControlPID();
void MoverMotores(int izq, int der, int centro);
void ObtenerReferenciaCero();

// ::::: VOID SETUP :::::

void setup() {
    pinMode(LED_01, OUTPUT);
    pinMode(LED_02, OUTPUT);
    pinMode(LED_03, OUTPUT);

    Serial.begin(SERIAL_BAUD);

    jsonRecibido.reserve(String_LENGTH);
    jsonRecibido = "";

    jsonCompletado = false;
    bSerialCompletado = true;

    ServoIzq.attach(SERVO_IZQ);
    ServoDer.attach(SERVO_DER);
    ServoCen.attach(SERVO_CEN);

    Brush.attach(BRUSHLESS);
    Brush.writeMicroseconds(BRUSH_STARTV);

    // ::::: MPU6050 :::::

    Accel_Gyro.initialize();
    Tiempo_Actual = millis();
    Wire.begin();

    if (Accel_Gyro.testConnection()) {
        Serial.println("Sensor iniciado correctamente");
        ObtenerReferenciaCero();
    }
    else {
        Serial.println("Error al iniciar el sensor");
    }

    MoverMotores(90, 90, 90);
    delay(5000);

}

// ::::: VOID LOOP :::::

void loop() {
    EventoSerial();

    if (bPID == true) {
        ControlPID();
    } else {
    }

    if (jsonCompletado) {
        Serial.println(jsonRecibido);
        jsonRecibido = "";
        jsonCompletado = false;
    }
}

// ::::: Funciones Globales Definicion :::::

void EventoSerial() {

```

```

while (Serial.available()) {
    char inChar = (char)Serial.read();
    jsonRecibido += inChar;
    if (inChar == '\n') {
        JsonParser();
    }
}

void JsonParser() {
    int iBrush_Value;
    String sBrush_Value;
    StaticJsonBuffer<200> jsonBuffer;
    short lengthJSON = jsonRecibido.length();
    char inputJSON[lengthJSON] = {0};
    jsonRecibido.toCharArray(inputJSON, lengthJSON);
    JsonObject& root = jsonBuffer.parseObject(inputJSON);
    if (root.success()) {
        String J_LED_01    = root["LED_01"];
        String J_LED_02    = root["LED_02"];
        String J_LED_03    = root["LED_03"];
        String J_Serv1     = root["SERVO_01"];
        String J_Serv2     = root["SERVO_02"];
        String J_Serv3     = root["SERVO_03"];
        String J_Brush     = root["BRUSH"];
        String J_BrushValue = root["BRUSH_VALUE"];
        String J_PID       = root["PID"];

        if (J_PID == "ENCENDIDO") {
            bPID = true;
        } else {
            bPID = false;
        }

        if (bPID == true) {
            ControlPID();
        } else {
            // LEDs
            if (J_LED_01 == "HIGH") {
                digitalWrite(LED_01, HIGH);
            } else if (J_LED_01 == "DOWN") {
                digitalWrite(LED_01, LOW);
            }

            if (J_LED_02 == "HIGH") {
                digitalWrite(LED_02, HIGH);
            } else if (J_LED_02 == "DOWN") {
                digitalWrite(LED_02, LOW);
            }

            if (J_LED_03 == "HIGH") {
                digitalWrite(LED_03, HIGH);
            } else if (J_LED_03 == "DOWN") {
                digitalWrite(LED_03, LOW);
            }

            // :: Servomotores
            if (J_Serv1 == "180") {
                ServoIzq.write(180);
            } else if (J_Serv1 == "100") {
                ServoIzq.write(100);
            } else if (J_Serv1 == "90") {
                ServoIzq.write(90);
            } else if (J_Serv1 == "0") {

```

```

        ServoIzq.write(0);
    }

    if (J_Serv2 == "180") {
        ServoDer.write(180);
    } else if (J_Serv2 == "100") {
        ServoDer.write(100);
    } else if (J_Serv2 == "90") {
        ServoDer.write(90);
    } else if (J_Serv2 == "0") {
        ServoDer.write(0);
    }

    if (J_Serv3 == "180") {
        ServoCen.write(180);
    } else if (J_Serv3 == "90") {
        ServoCen.write(90);
    } else if (J_Serv3 == "0") {
        ServoCen.write(0);
    }

    if (J_Brush == "START") {
        sBrush_Value = J_BrushValue;

        iBrush_Value = sBrush_Value.toInt();

        iBrush_Value = map(iBrush_Value, 0, 100, 1000, 1700);

        Brush.writeMicroseconds(iBrush_Value);

    } else if (J_Brush == "STOP") {
        Brush.writeMicroseconds(1000);
    }
}

} else {
    Serial.println("parseObject() failed");
}

jsonCompletado = true;
bSerialCompletado = true;
}

void ControlPID() {
    Tiempo_Anterior = Tiempo_Actual;
    Tiempo_Actual = millis();
    Tiempo_Transcurrido = (Tiempo_Actual - Tiempo_Anterior);

    /* ::: Lectura de Datos del MPU ::: */
    Accel_Gyro.getAcceleration(&Aceleracion_X, &Aceleracion_Y, &Aceleracion_Z);
    Accel_Gyro.getRotation(&Rotacion_X, &Rotacion_Y, &Rotacion_Z);

    /* ::: Calculo de Rotación Angular ::: */
    Angulo_X = atan(Aceleracion_X / sqrt(pow(Aceleracion_Y, 2) + pow(Aceleracion_Z, 2))) * (180.0 / 3.14);
    Angulo_Y = atan(Aceleracion_Y / sqrt(pow(Aceleracion_X, 2) + pow(Aceleracion_Z, 2))) * (180.0 / 3.14);

    /* ::: Calculo de Valores Maximos y Minimos de Lectura ::: */
    if (Aceleracion_X < Aceleracion_XMin) Aceleracion_XMin = Aceleracion_X;
    if (Aceleracion_X > Aceleracion_XMax) Aceleracion_XMax = Aceleracion_X;

    if (Aceleracion_Y < Aceleracion_YMin) Aceleracion_YMin = Aceleracion_Y;
    if (Aceleracion_Y > Aceleracion_YMax) Aceleracion_YMax = Aceleracion_Y;

    if (Aceleracion_Z < Aceleracion_ZMin) Aceleracion_ZMin = Aceleracion_Z;
    if (Aceleracion_Z > Aceleracion_ZMax) Aceleracion_ZMax = Aceleracion_Z;

    /* ::: Mapeo de Valores ::: */
    mapXmax = map(Aceleracion_X, Referencia_X, Aceleracion_XMax, 0, 180);
    mapYmax = map(Aceleracion_Y, Referencia_Y, Aceleracion_YMax, 0, 180);
    mapZmax = map(Aceleracion_Z, Referencia_Z, Aceleracion_ZMin, 0, -180);

    /* ::: Calculo del Error ::: */
    Error = mapXmax - 0;

    /* ::: Calculo del proporcional del PID ::: */

```

```

pid_P = kP * Error;

/* ::: Calculo del Integral del PID ::: */
if (-15 < Error < 15) {
    pid_I = pid_I + (kI * Error);
}

/* ::: Calculo del Derivativo del PID ::: */
pid_D = kD * ((Error - Error_Anterior) / Tiempo_Transcurrido);

PID = pid_I + pid_P + pid_D;

if (PID > 180) {
    PID = 180;
} else if (PID < -180) {
    PID = -180;
}

int ExeMotorIzq;
int ExeMotorDer;
int ExeMotorCen;

if (PID > 0) {
    Serial.println("Derecha");
    ExeMotorIzq = map(PID, 1, 180, 90, 180);
    ExeMotorDer = map(PID, 1, 180, 90, 0);
    ExeMotorCen = map(PID, 1, 180, 90, 180);
} else {
    Serial.println("Izquierda");
    ExeMotorIzq = map(PID, 0, -180, 90, 180);
    ExeMotorDer = map(PID, 0, -180, 0, 90);
    ExeMotorCen = map(PID, 0, -180, 90, 180);
}

MoverMotores(ExeMotorIzq, ExeMotorDer, ExeMotorCen);

Error_Anterior = Error;

/* ::: Impresión de Datos ::: */
// Serial.print("X: "); Serial.println(mapXmax);
// Serial.print("Y: "); Serial.println(mapYmax);
// Serial.print("Z: "); Serial.println(mapZmax);

Serial.print(PID);
delay(50);
}

void MoverMotores(int izq, int der, int centro) {
    ServoIzq.write(izq);
    ServoDer.write(der);
    ServoCen.write(centro);
}

void ObtenerReferenciaCero() {
    int muestrasX[numeroMuestras] = {0};
    int muestrasY[numeroMuestras] = {0};
    int muestrasZ[numeroMuestras] = {0};

    for (int i = 0; i < numeroMuestras; i++) {
        Accel_Gyro.getAcceleration(&Aceleracion_X, &Aceleracion_Y, &Aceleracion_Z);
        muestrasX[i] = Aceleracion_X;
        muestrasY[i] = Aceleracion_Y;
        muestrasZ[i] = Aceleracion_Z;
    }

    for (int i = 0; i < numeroMuestras; i++) {
        Referencia_X += muestrasX[i];
        Referencia_Y += muestrasY[i];
        Referencia_Z += muestrasZ[i];
    }

    Referencia_X = Referencia_X / numeroMuestras;
    Referencia_Y = Referencia_Y / numeroMuestras;
    Referencia_Z = Referencia_Z / numeroMuestras;
}

```

Anexo 03 – Codigo QT Creator Emisor JSON + Video Stream + Cliente Telnet Wifi

```
#define CAM_STRING "http://192.168.1.101:8080/video"
#define LOOP_FRECMS 30

#include "mainwindow.h"
#include "ui_mainwindow.h"

using namespace cv;

/*::::::::::::::::::::::::::VARIABLES GLOBALES::::::::::::::::::::::::::*/
QProcess qpTelnet;
QProcess qpWifiConexion;

QString sLectura;
QString sDir_adelante = "{\"SERVO_01\":\"90\",\"SERVO_02\":\"90\",\"SERVO_03\":\"90\"}\n";
QString sDir_atras = "{\"SERVO_01\":\"90\",\"SERVO_02\":\"90\",\"SERVO_03\":\"90\"}\n";
QString sDir_vuelta_Izq = "{\"SERVO_01\":\"180\",\"SERVO_02\":\"90\",\"SERVO_03\":\"180\"}\n";
QString sDir_vuelta_Der = "{\"SERVO_01\":\"90\",\"SERVO_02\":\"0\",\"SERVO_03\":\"0\"}\n";
QString sDir_neutro = "{\"SERVO_01\":\"90\",\"SERVO_02\":\"90\",\"SERVO_03\":\"90\"}\n";
QString sAce_Brushless;
QString sPid_ControlOn = "{\"PID\":\"ENCENDIDO\"}\n";
QString sPid_ControlOff = "{\"PID\":\"APAGADO\"}\n";

VideoCapture vcCamara(CAM_STRING);

/*::::::::::::::::::::::::::PROTOTIPOS DE FUNCIONES ::::::::::::::::::::::::::::*/
QString EscribirTelnet(QString Mensaje);
QString LeerTelnet();
QString TelnetConexion();

/*:::::::::::::::::::::::::: FUNCIONES QT-CREATOR ::::::::::::::::::::::::::::*/
MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    QTimer* loop = new QTimer(this);
    ui->LBL_Respuesta->setText(TelnetConexion());

    if(!vcCamara.isOpened()) {
        vcCamara.open(CAM_STRING);
    }

    connect(loop, SIGNAL(timeout()), this, SLOT(funcionLoop()));
    loop->start(LOOP_FRECMS);
}

void MainWindow::funcionLoop(){
    sLectura = LeerTelnet();

    if(!sLectura.isEmpty()){
        ui->LBL_Respuesta->setText(sLectura);
    }

    if(vcCamara.isOpened()){
        Mat IMAGEN;
        Mat IMAGENpequena;
        QImage qImage;
        QPixmap pixmap;

        vcCamara >> IMAGEN;
        cv::resize(IMAGEN, IMAGENpequena, Size(640, 480));

        qImage = Mat2QImage(IMAGENpequena);
        pixmap = QPixmap::fromImage(qImage);

        ui->LBL_Image->clear();
        ui->LBL_Image->setPixmap(pixmap);
    }
}
```

```

}

/*:::::::::::::::::::::::::::::::::::: FUNCIONES TELNET ::::::::::::::::::::::::::::::*/
QString TelnetConexion(){
    qpTelnet.start("telnet 192.168.1.1 23");
    qpTelnet.waitForFinished(10);
    return qpTelnet.readAllStandardOutput();
}

QString EscribirTelnet(QString Mensaje){
    qpTelnet.write(Mensaje.toUtf8().constData());
    qpTelnet.waitForFinished(1);

    QString stdout = qpTelnet.readAllStandardOutput();
    QString stderr = qpTelnet.readAllStandardError();
    qDebug() << stdout;
    return stdout;
}

QString LeerTelnet(){
    QString stdout = qpTelnet.readAllStandardOutput();
    QString stderr = qpTelnet.readAllStandardError();

    if(!stdout.isEmpty()){
        qDebug() << stdout;
        return stdout;
    }
    return "";
}

void MainWindow::on_SB_Aceleracion_sliderMoved(int position)
{
    sAce_Brushless = "{\"BRUSH\":\"START\", \"BRUSH_VALUE\": \"\"";
    sAce_Brushless.append(QString::number(position));
    sAce_Brushless.append("\"}\n");

    qDebug() << sAce_Brushless;
    EscribirTelnet(sAce_Brushless);
}

/*:::::::::::::::::::::::::::::::::::: FUNCIONES CONTROLES ::::::::::::::::::::::::::::::*/
void MainWindow::on_BTN_Adelante_clicked(bool checked)
{
    if(checked == true){
        EscribirTelnet(sDir_adelante);
        qDebug() << sDir_adelante;
    } else {
        EscribirTelnet(sDir_neutro);
        qDebug() << sDir_neutro;
    }
}

void MainWindow::on_BTN_Atras_clicked(bool checked)
{
    if(checked == true){
        EscribirTelnet(sDir_atras);
        qDebug() << sDir_atras;
    } else {
        EscribirTelnet(sDir_neutro);
        qDebug() << sDir_neutro;
    }
}

void MainWindow::on_BTN_Izquierda_clicked(bool checked)
{
    if(checked == true){
        EscribirTelnet(sDir_vuelta_Izq);
        qDebug() << sDir_vuelta_Izq;
    } else {
        EscribirTelnet(sDir_neutro);
        qDebug() << sDir_neutro;
    }
}

void MainWindow::on_BTN_Derecha_clicked(bool checked)
{
    if(checked == true){
        EscribirTelnet(sDir_vuelta_Der);
        qDebug() << sDir_vuelta_Der;
    } else {

```

```

        EscribirTelnet(sDir_neutro);
        qDebug() << sDir_neutro;
    }
}

void MainWindow::on_CBX_ControlPID_toggled(bool checked)
{
    if(checked == true){
        EscribirTelnet(sPid_Controlon);
        qDebug() << sPid_Controlon;
    }else{
        EscribirTelnet(sPid_Controloff);
        qDebug() << sPid_Controloff;
    }
}

void MainWindow::on_BTN_ConectarHotSPot_clicked()
{
    qpWifiConexion.start("nmcli c up ECOVAPO");
    qpWifiConexion.waitForFinished(-1);
    ui->LBL_Respuesta->setText(qpWifiConexion.readAllStandardOutput());
}

MainWindow::~MainWindow(){delete ui;}

```

Anexo 04 – Código Interfaz Qt Creator

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
  <class>MainWindow</class>
  <widget class="QMainWindow" name="MainWindow">
    <property name="geometry">
      <rect>
        <x>0</x>
        <y>0</y>
        <width>742</width>
        <height>411</height>
      </rect>
    </property>
    <property name="windowTitle">
      <string>MainWindow</string>
    </property>
    <widget class="QWidget" name="centralWidget">
      <widget class="QScrollBar" name="SB_Aceleracion">
        <property name="geometry">
          <rect>
            <x>590</x>
            <y>110</y>
            <width>141</width>
            <height>251</height>
          </rect>
        </property>
        <property name="minimum">
          <number>0</number>
        </property>
        <property name="maximum">
          <number>100</number>
        </property>
        <property name="pageStep">
          <number>10</number>
        </property>
        <property name="orientation">
          <enum>Qt::Vertical</enum>
        </property>
        <property name="invertedAppearance">
          <bool>true</bool>
        </property>
      </widget>
      <widget class="QPushButton" name="BTN_Derecha">
        <property name="geometry">
          <rect>
            <x>100</x>
            <y>210</y>
            <width>91</width>
            <height>51</height>
          </rect>
        </property>
        <property name="text">
          <string>Derecha</string>
        </property>
        <property name="checkable">
          <bool>true</bool>
        </property>
      </widget>
      <widget class="QPushButton" name="BTN_Atras">
        <property name="geometry">
          <rect>
            <x>60</x>
            <y>260</y>
            <width>81</width>
            <height>61</height>
          </rect>
        </property>
        <property name="text">
```



```

        <string>Atras</string>
    </property>
    <property name="checkable">
        <bool>true</bool>
    </property>
</widget>
<widget class="QPushButton" name="BTN_Izquierda">
    <property name="geometry">
        <rect>
            <x>10</x>
            <y>210</y>
            <width>91</width>
            <height>51</height>
        </rect>
    </property>
    <property name="text">
        <string>Izquierda</string>
    </property>
    <property name="checkable">
        <bool>true</bool>
    </property>
</widget>
<widget class="QPushButton" name="BTN_Adelante">
    <property name="geometry">
        <rect>
            <x>60</x>
            <y>150</y>
            <width>81</width>
            <height>61</height>
        </rect>
    </property>
    <property name="text">
        <string>Adelante</string>
    </property>
    <property name="checkable">
        <bool>true</bool>
    </property>
</widget>
<widget class="QLabel" name="LBL_Image">
    <property name="geometry">
        <rect>
            <x>210</x>
            <y>110</y>
            <width>361</width>
            <height>251</height>
        </rect>
    </property>
    <property name="frameShape">
        <enum>QFrame::Box</enum>
    </property>
    <property name="frameShadow">
        <enum>QFrame::Sunken</enum>
    </property>
    <property name="text">
        <string>IMAGE</string>
    </property>
</widget>
<widget class="QLabel" name="LBL_Respuesta">
    <property name="geometry">
        <rect>
            <x>10</x>
            <y>50</y>
            <width>721</width>
            <height>41</height>
        </rect>
    </property>
    <property name="font">
        <font>
            <weight>75</weight>
            <bold>true</bold>
        </font>
    </property>
    <property name="frameShape">
        <enum>QFrame::Box</enum>
    </property>
    <property name="text">
        <string/>
    </property>
</widget>
<widget class="QLineEdit" name="EDT_ComandoTelnet">
    <property name="geometry">

```

```

    <rect>
      <x>90</x>
      <y>10</y>
      <width>641</width>
      <height>31</height>
    </rect>
  </property>
</widget>
<widget class="QPushButton" name="BTN_ConectarHotSpot">
  <property name="geometry">
    <rect>
      <x>10</x>
      <y>10</y>
      <width>81</width>
      <height>31</height>
    </rect>
  </property>
  <property name="text">
    <string>Conectar HotSpot</string>
  </property>
</widget>
<widget class="QCheckBox" name="CBX_ControlPID">
  <property name="geometry">
    <rect>
      <x>60</x>
      <y>110</y>
      <width>91</width>
      <height>31</height>
    </rect>
  </property>
  <property name="text">
    <string>Equilibrar</string>
  </property>
</widget>
</widget>
<widget class="QMenuBar" name="menuBar">
  <property name="geometry">
    <rect>
      <x>0</x>
      <y>0</y>
      <width>742</width>
      <height>18</height>
    </rect>
  </property>
</widget>
<widget class="QToolBar" name="mainToolBar">
  <attribute name="toolBarArea">
    <enum>TopToolBarArea</enum>
  </attribute>
  <attribute name="toolBarBreak">
    <bool>false</bool>
  </attribute>
</widget>
<widget class="QStatusBar" name="statusBar"/>
</widget>
<layoutdefault spacing="6" margin="11"/>
<resources/>
<connections/>
</ui>

```

Anexo 05

Anexo 06

Anexo 07

Anexo 08

Anexo 09

Anexo 10