Universidad de Ingeniería y Tecnología

Informe Final de Proyecto

Diseño de Sistemas Mecatrónicos

Autores:

Marcelo Contreras
Dpto. de Ingeniería Mecatrónica
marcelo.contreras@utec.edu.pe

César Guillén
Dpto. de Ingeniería Mecatrónica
cesar.guillen@utec.edu.pe

Claudia Pacori
Dpto. de Ingeniería Electrónica
claudia.pacori@utec.edu.pe

Sebastián Peralta
Dpto. de Ingeniería Mecatrónica
sebastian.peralta@utec.edu.pe

Diciembre 2022

Índice

1. Trabajo en grupo

El trabajo fue repartido de la siguiente manera entre los integrantes:

- Marcelo Contreras: Subsistema de comunicaciones (programación del Raspberry Pi), diagramas del sistema propuesto, control del motor DC (movimiento de la bandeja), procesamiento de imágenes (segmentación del color).
- César Guillén: Subsistema mecánico (armado de la estructura con el kit Tetrix), estado del arte, simulación en StateFlow, diseño CAD mecánico en Inventor, programación de la celda de carga (medición de masa).
- Claudia Pacori: Subsistema electrónico (cableado del prototipo), armado de la estructura con el kit Tetrix, diseño CAD electrónico (printed circuit board y modelo 3D del HX711), interfaz de usuario en Figma, integración del código de Arduino.
- Sebastián Peralta: Subsistema de potencia, simulación en StateFlow, diseño CAD mecánico en Inventor (celda de carga y prototipo final), control de los servomotores (remoción de los camarones).

2. Estado del arte

El problema a resolver se encuentra enmarcado en el rubro de crianza de camarones. Dadas las condiciones en las que se lleva a cabo el monitoreo de la alimentación en las granjas, resulta pertinente aplicar soluciones que mejoren el control de las cantidades de alimento que está siendo consumido por los camarones. De este modo, se consigue optimizar costos en cuanto al mantenimiento de la granja, se sostienen las condiciones de fauna en el criadero y se facilita la tarea de revisión que actualmente se lleva a cabo con completa dependencia en el ser humano.

2.1. Detección de alimentación

Dependiendo de la especie del camarón que se tiene en el criadero, los métodos pueden variar en cuanto a la variable que se desea medir. Sin embargo, los trabajos de Peixoto [1] y Reis [2] muestran que los sonidos por oclusión (mordida) generan marcas características para el proceso de alimentación. Es necesario realizar pruebas en las que se identifiquen los parámetros para cada alimento de manera que se adecúe la recopilación de información al criadero en cuestión. El análisis previo puede ser puesto a prueba al convertir los resultados con base en parámetros estadísticos como la tasa de conversión de alimento, el crecimiento diario promedio o el ratio de supervivencia (FCR, ADG y SR por sus siglas en inglés). [3].

2.2. Acústica de camarones

El monitoreo acústico pasivo (PAM) ha sido una de las aproximaciones que mayores avances ha brindado en cuanto al comportamiento de alimentación de camarones de distintas especies. No solamente se consiguen coeficientes de correlación de alrededor de 0.95 entre la detección de *feeding signatures* y *pellet consumption*, sino que sus implementaciones de

prueba obtienen incrementos de crecimiento en los camarones con base en la medición del peso de hasta 46 % [4]. En lo que respecta a *Litopenaeus vannamei*, la especie que representa el 80 % de la producción mundial de camarón, se sabe que es posible caracterizar el sonido de un "click" continuo producido por colisión y fricción de sus mandíbulas, pero es también ventajoso reconocer que el mayor sonido proviene de su movimiento para alcanzar la comida como tal [5]. Por ello, resulta relevante contar con un ambiente de prueba controlado en el que se puede comprobar el comportamiento de las señales que serán procesadas posteriormente por el sistema, ya que con estas se identifica la frecuencia de alimentación. La desventaja de este método es que necesita un estudio previo intensivo sobre el preprocesamiento que se debe realizar sobre la señales de audio para obtener solo las frecuencias asociadas a los camarones.

2.3. Implementaciones físicas de sistemas de alimentación

Al momento de poner en funcionamiento un sistema para mejorar las condiciones en las que se lleva a cabo la crianza de los camarones en las granjas, se opta principalmente por automatizar el proceso. Con este fin, la tecnología de detección de comida es críticamente relevante. Para esto, es frecuente observar sistemas que emplean fotografías de los recipientes destinados para el alimento, las cuales deben ser procesadas para conseguir la información requerida. Fundamentalmente, el algoritmo se basa en identificar qué píxeles pueden ser considerados como alimento y cuáles no, así se consigue la clasificación para determinar cantidades consumidas y restantes [6].

Para operar con base en ese algoritmo, Huang, et al. recurrieron a inteligencia artificial con *deep learning* para atacar las dificultades presentadas por la turbidez del agua de los criaderos y la baja visibilidad propia de las imágenes submarinas [7]. No es un requerimiento que el sistema cuente con una avanzada capacidad de procesamiento en todos los casos, depende de la característica que se pretenda optimizar, pero es una de las alternativas actualmente exploradas para disminuir la dependencia de operarios de los sistemas de alimentación. Además, es relevante tener en cuenta el consumo de energía que está habilitado para el entorno en el que se lleve a cabo la aplicación. En ese sentido, una de las plataformas de desarrollo IoT más utilizadas es la Raspberry-PI es una opción, aunque resulta ventajoso incluir, donde sea posible, el microcontrolador Atmega328P dado su bajo consumo. Un aspecto importante es que mientras menos instancias (objetos) se encuentren en la foto, la estimación de la comida será menos ruidosa. Por lo tanto, es crítico que antes de tomar la foto, el comedero no presente camarones o sea capaz de identificarlos para rechazarlos en la estimación. [8].

3. Propuestas de solución y análisis de factibilidad

Para las propuestas de solución, se plantean diagramas básicos que muestren su funcionamiento correspondiente. Cada una fue analizada en términos de costos, riesgos técnicos y su aceptabilidad, tanto social, como cultural.

3.1. Solución 1

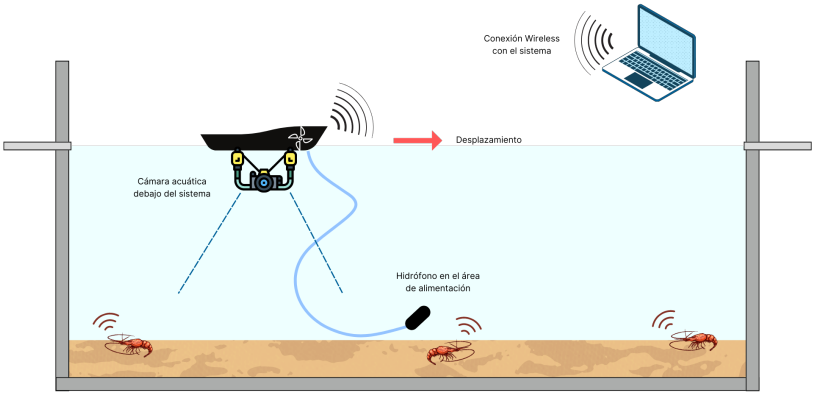


Figura 1: Representación del sistema de barco autónomo con cámara e hidrófono

La primera propuesta de solución consiste en la implementación de un barco autónomo que se dirija a los puntos en los que son colocadas las bandejas de alimentación para los camarones. Con ayuda de una cámara acuática en la parte inferior de su casco, captura una fotografía que puede ser procesada al enviarse por comunicación inalámbrica a un dispositivo operado en un ambiente de supervisión del criadero. El movimiento está dado por un motor conectado a una hélice y también se registra información con ayuda de un hidrófono respecto al comportamiento de los camarones en los puntos de alimentación.

3.2. Solución 2

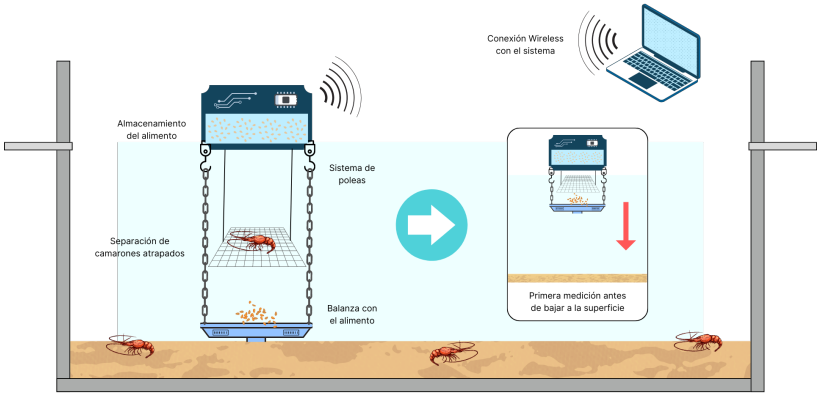


Figura 2: Representación del sistema de plataforma flotante con alimentación automatizada, balanza integrada y bandeja

La segunda propuesta consiste en una plataforma flotante, la cual almacena comida en su interior, la cual desplegará una bandeja, en la que se verterá el alimento, y una malla que los cubrirá. Es decir, el alimento se encontrará entre la bandeja y la malla, por lo que esta última debe ser lo suficientemente porosa para que los camarones puedan alimentarse, pero no tanto como para que puedan pasar a través.

3.3. Solución 3

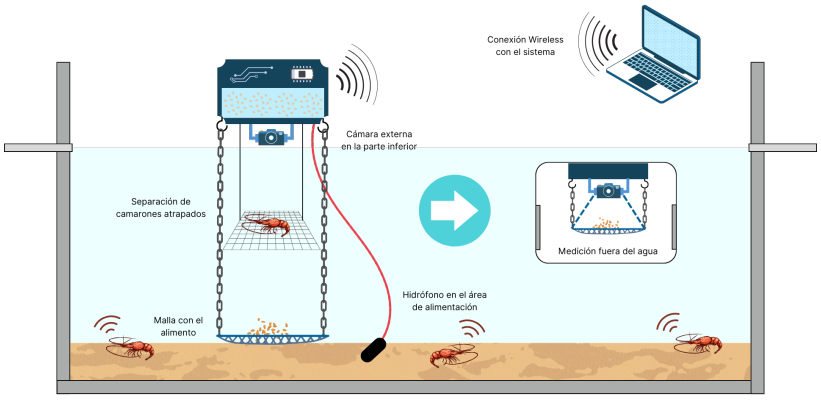


Figura 3: Representación del sistema de plataforma flotante compacta con alimentación automatizada, hidrófono y cámara

La tercera solución reutilizará el descenso de plataforma con alimento y el uso de la malla para retirar los camarones. No obstante, integrará el funcionamiento de una cámara fija en la plataforma flotante que no hace contacto con el agua y un hidrófono. La cámara capturará imágenes de bandeja con comida sin camarones, las enviará a la computadora remota y se les aplicará técnicas de procesamiento para cuantificar la cantidad de alimento consumido. Por otra parte, el hidrófono permitirá sondear áreas donde los camarones tengan hambre y también determinará si los camarones están comiendo el alimento depositado. De esta manera, se podrá determinar lugar donde se requiere comida y el tiempo de su consumo para así ajustar su cantidad y frecuencia de suministro.

3.4. Análisis de factibilidad

Análisis de factibilidad		
Costo	**Riesgos técnicos**	**Aceptabilidad**
Propuesta de solución 1		
Medio (\$105**).** El casco del barco puede ser fabricado sin demasiados requerimientos, ya que solamente debe alcanzar los puntos de las bandejas de alimentación para capturar las fotografías. El mayor de los costos está en la batería que requiere este sistema, ya que el consumo energético sí es un punto crítico.	**(1) Consumo energético.** Se trata de un solo vehículo autónomo que debe recorrer todos los puntos de alimentación, lo que implica mayor consumo de baterías. **(2) Claridad de imágenes.** Al capturar imágenes sin sacar las bandejas del agua, se debe lidiar con el grado de turbidez que esta presenta al procesar la información.	**Alta.** La propuesta cuenta con una aproximación en gran medida amigable para los usuarios finales de la solución, ya que conceptualmente el sistema realiza su misma tarea, solo que recopila la información de manera diferente y les proporciona resultados más precisos para considerar.
Propuesta de solución 2		
Medio (\$100). Debido a que existen materiales impermeables de bajo costo que pueden ser utilizados en la carcasa; además de que solo cuenta con una forma de medición del alimento consumido, el cual es la balanza.	**(1) Variación de la masa del alimento debido a la absorción de agua.** Esto podría generar que la data obtenida no sea confiable.**(2) Los camarones podrían quedarse atrapados en la malla**	**Alta**, debido a la gran semejanza con el método de comederos y mediciones completamente manuales que han estado llevando hasta ahora. Además, simplifica el trabajo de echar alimento en la piscina.
Propuesta de solución 3		
Medio $\sim$ **Alto (**\$120) Los materiales impermeable siguen siendo de bajo costo para la construcción de la base flotante. El incremento de gastos en la cámara es acompañado por el de comprar un microprocesador más potente. El hidrófono tiene un rango de precios amplio.	**(1) Se repiten riesgos técnicos de propuesta 2** (2) **Problemas de claridad** en las imágenes por falta de luz (3) **La tensión del cable del hidrófono** podría jalar toda la plataforma si otro animal tira de este.	**Alta**: El método de comederos y medición manual tendría mucha familiaridad con el propuesto. Igualmente, simplifica el trabajo de echar alimento en la piscina. La inclusión de nuevos sensores podrá presentar una barrera tecnológica; no obstante, estos datos les darán resultados más precisos en el monitoreo.

Cuadro 1: Comparación de las soluciones propuestas

3.5. Fundamento de la elección de la propuesta final

Evaluando las opciones presentadas, se considera mantener el sistema de monitoreo mostrado en las propuestas 1 y 3, y se evalúa cambiar la balanza utilizada como plato en la solución 2, por una bandeja simple. Por otra parte, se implementará un sistema para remover los camarones de la superficie de medición. Este requerirá que la estructura flotante sea más alargada a fin de utilizar una malla más larga que la bandeja. Luego, cuando se requiera retirar el comedero, se deberá subir, tanto la bandeja, como la malla fuera del agua. Se deberá levantar la malla una distancia mayor que la bandeja, de tal manera que esta pueda ser girada mediante el sistema de poleas, como se muestra en la figura 4.

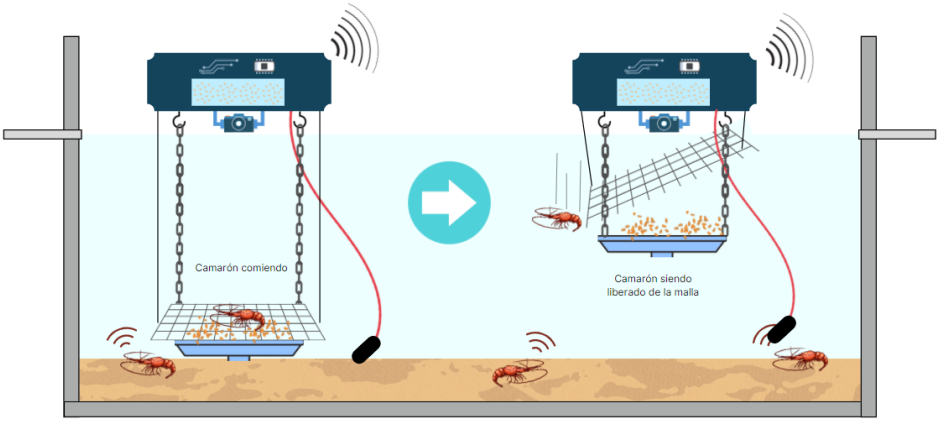


Figura 4: Representación del sistema de la propuesta final

4. Planificación y tiempos de ejecución

Para la implementación del sistema escogido se propone el siguiente plan de trabajo. Este se encuentra dividido en cinco etapas:

1. **Revisión bibliográfica y análisis situacional**: Se realiza la selección de fuentes relevantes para obtener una visión más clara del problema. Se filtran resultados relacionados con *automated shrimp farming* dentro de motores de búsqueda como IEEE Xplore y Science Direct Además, se evalúa la factibilidad de las soluciones actuales para la alimentación de camarones. Los criterios de selección tomarán especial atención al presupuesto de la solución, complejidad de solución y reivindicación a las técnicas tradicionales que se han estado utilizando por los usuarios finales. De esta manera, se elige una propuesta de solución óptima para su futura implementación.

2. **Selección de materiales y componentes electrónicos**: Una vez teniendo en cuenta un posible sistema y su funcionamiento, se determina el tipo de alimentación y comunicación de este mismo. La distancia y velocidad de transmisión serán factores importantes para la selección del tipo de comunicación. También, es necesario elegir el microcontrolador, material, sensores y actuadores más adecuados teniendo en cuenta el presupuesto y uso que se les dará a cada uno. Se buscará mantener un consumo lo más bajo posible para que la necesidad de reponer el sistema sea a mayor plazo donde una semana es el deseable.

3. **Diseño electrónico**: Se definen las conexiones entre los componentes seleccionados, a través de un diagrama esquemático y una placa de circuito impreso para optimizar el espacio del sistema electrónico. El producto final de la PCB tendrá que estar preparado para las condiciones del ambiente al que será expuesta. Asimismo, se realizará la programación para controlar los periféricos con el microcontrolador.

4. **Diseño mecánico**: Se construye un prototipo físico del sistema mecánico, analizando el movimiento que tiene cada parte y teniendo en cuenta el medio acuático en el que estará la solución. Es necesario controlar la frecuencia con la que se realizan las

medidas, es decir, evitar que un movimiento en específico sea muy recurrente con lo que se puede reducir el desgaste de las piezas mecánicas.

5. **Integración del sistema completo**: Por último, unir el diseño electrónico en el diseño mecánico y realizar las pruebas necesarias en un entorno controlado. Esta etapa tiene un flujo recursivo donde después de las pruebas se realizarán ajustes y se volverá a probar el rendimiento en los entornos propuestos. Con esto se podrá asegurar el funcionamiento adecuado de la solución implementada.

A continuación, en la Figura 5 se muestra un diagrama de Gantt en el que se definen las fechas y tiempos de ejecución para cada etapa y subproceso. Se consideran las semanas académicas del semestre universitario.

Diseño de sistemas mecatrónicos

Plataforma flotante compacta con sistema de alimentación automatizado, hidrófono y cámara

Indice	Tarea	Inicio	Fin	Semana académica	Septiembre				Octubre				Noviembre				
					S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4	S5
1	**Revisión bibliográfica y análisis situacional**	**05/09/2022**	**11/09/2022**	S4													
1.1	Buscar fuentes bibliográficas	05/09/22	07/09/22														
1.2	Seleccionar fuentes relevantes	05/09/22	07/09/22														
1.3	Evaluar soluciones actuales	07/09/22	09/09/22														
1.4	Seleccionar solución óptima	09/09/22	11/09/22														
2	**Selección de materiales y componentes electrónicos**	**12/09/2022**	**25/09/2022**	S5-6													
2.1	Determinar el tipo de alimentacion y funcionamiento esperado	12/09/22	12/09/22														
2.2	Seleccionar el material más adecuado	13/09/22	15/09/22														
2.3	Seleccionar el microcontrolador requerido	15/09/22	20/09/22														
2.4	Seleccionar los sensores y actuadores requeridos	21/09/22	25/09/22														
3	**Realizar el diseño electrónico**	**26/09/2022**	**16/10/2022**	S7-9													
3.1	Definir las conexiones entre componentes	26/09/22	30/09/22														
3.2	Hacer el diseño del PCB	01/10/22	06/10/22														
3.3	Realizar la programación del sistema	07/10/22	16/10/22														
4	**Realizar el diseño mecánico**	**17/10/2022**	**06/11/2022**	S10-12													
4.1	Obtención de materiales	17/10/22	23/10/22														
4.2	Construcción del prototipo físico	24/10/22	06/11/22														
5	**Integración del sistema completo**	**07/11/2022**	**27/11/2022**	S13-15													
5.1	Integrar el diseño electrónico en el diseño mecánico	07/11/22	20/11/22														
5.2	Pruebas en un entorno controlado	21/11/22	27/11/22														
6	**Exposición final de resultados**	**28/11/2022**	**29/11/2022**	S16													

Figura 5: Diagrama de Gantt

5. Diagrama del sistema propuesto

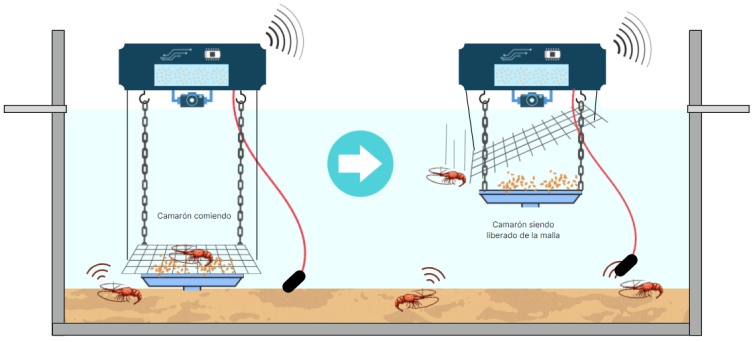


Figura 6: Ejemplo gráfico de la solución elegida en sus distintos funcionamientos

El siguiente diagrama debe contener los sistemas de actuadores y sensores elegidos para la propuesta de implementación. Estos serán divididos en diagrama mecánico y eléctrico donde se detallaran más las características del sistema. Asimismo, como parte esencial del producto final, se incluye la interfaz de usuario por la cual se podrá conocer la ubicación de cada estación y su lectura de cantidad de alimento junto a otras variables de interés.

5.1. Diagrama eléctrico

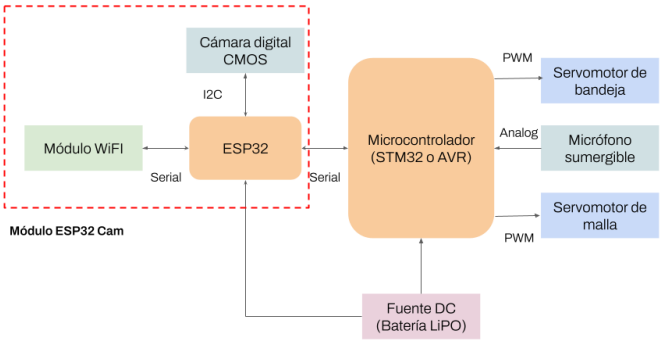


Figura 7: Diagrama eléctrico considerando componentes, sensores y uC's

Dentro del diagrama de la Figura 7 se muestran las conexiones eléctricas entre los distintos componentes utilizados. En primer lugar, se tiene un microcontrolador general que se conecta con los actuadores servomotores, uno para la bandeja con alimento y otro para la malla, el hidrófono, y el sistema de alimentación con una batería LiPo. Este se encargará de enviar las señales PWM para subir o bajar la malla y la bandeja de forma independiente. Asimismo, tendrá que leer la señal analógica del hidrófono, filtrarla y enviar externamente para mayor procesamiento.

Cómo la adquisición de imágenes es una carga considerable para un microcontrolador, una opción factible es utilizar uno que solo se encargue de este apartado. Dentro del mercado existe una opción particular conocida como ESP32 CAM que integra una cámara digital OV2640 y un módulo WiFi. Por lo tanto, la tarea de recepción y envió de imágenes se realizaría en un entorno a partir del microcontrolador general. No obstante, sí existirá una conexión entre este y el ESP32 CAM para recibir interrupciones que deseen ser enviadas desde la computadora por WiFi.

5.2. Diagrama mecánico

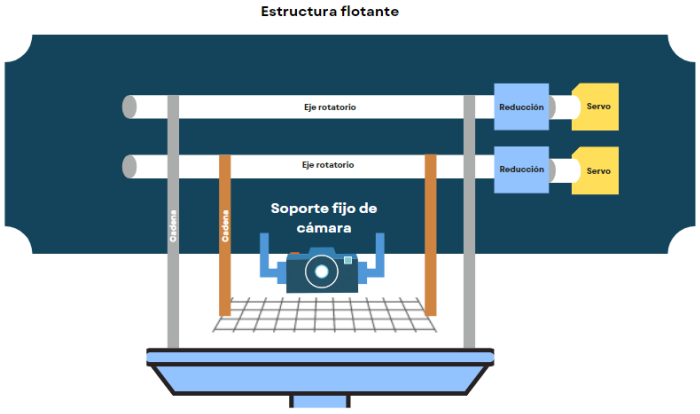


Figura 8: Diagrama mecánico

El diagrama mecánico muestra la interconexión de los actuadores mecánicos con el resto de la estructura flotante. Cómo los servomotores no tiene un torque muy elevado y la velocidad de rotación es crítica para esta aplicación, se puede acoplar una caja de reducción adicional externa a los servomotores para que se logre mover tanto la bandeja como la red en vista que los servomotores ya cuentan con una caja de reducción interna. Por otro lado, la cámara se encontrará acoplada a la estructura flotante por un soporte fijo, donde el lente de la cámara estará en vista aérea hacia la bandeja.

6. Diseño de máquina de estados

Para el diseño de la máquina de estados, se requiere plantear los estados del sistema, los eventos y acciones que están involucradas en su operación. De esta manera, es viable establecer la simulación de la lógica de su comportamiento por medio del uso del software Stateflow en MATLAB. Como base, se están considerando las propuestas evaluadas en el trabajo de Darodes [4].

6.1. Identificación de estados

- **Esperando.** Se trata de un estado de inactividad del sistema con la búsqueda de ahorro energético. En estos periodos no se está procesando, ni enviando, ni recogiendo información, por lo que no se requiere de la activación de sus componentes.
- **Procesando audio.** Es un estado en el que el sistema procesa las muestras de audio que son captadas con el hidrófono instalado en los puntos de alimentación del criadero.
- **Tomando imagen.** En este estado, la cámara es empleada para realizar la captura correspondiente de la bandeja de alimentación para llevar a cabo el procesamiento de imágenes que determine la cantidad de comida restante.

- **Procesando imagen.** Se trata del estado en el que el sistema lleva a cabo la lectura de la información procedente de la imagen capturada respecto a la cantidad de comida presente en la bandeja de alimentación.
- **Subiendo bandeja.** Mientras el sistema está en este estado, la bandeja de alimentación está ascendiendo hasta su posición fuera del agua para capturar la fotografía.
- **Bajando bandeja.** Mientras el sistema está en este estado, la bandeja de alimentación está descendiendo tras haberse obtenido los resultados que deben ser reportados al usuario.
- **Inclinando malla.** Mientras el sistema está en este estado, la malla se inclina para remover a los camarones de ella.
- **Inclinando malla 2.** Mientras el sistema está en este estado, la malla vuelve a su posición anterior a la inclinación.
- **Enviando información.** En este estado, el sistema realiza la comunicación inalámbrica para enviar el reporte de resultados a los encargados humanos del monitoreo del criadero de camarones.

6.2. Identificación de eventos

Los eventos están asociados a mediciones del entorno o cambios en variables internas que permiten la transición entre estados.

- **Audio recolectado.** En este evento, se verifica que se haya recolectado el audio del micrófono.
- **Audio procesado y camarones no tienen hambre.** En este evento, se verifica que se haya procesado el audio del micrófono y si los camarones no tienen hambre.
- **Audio procesado y camarones tienen hambre.** En este evento, se verifica que se haya procesado el audio del micrófono y si los camarones tienen hambre.
- **Bandeja arriba.** En este evento, se verifica que la bandeja se haya subido.
- **No hay camarones en bandeja y bandeja arriba.** En este evento, se verifica que la bandeja se haya subido y que no haya camarones en la bandeja.
- **Malla está inclinada.** En este evento, se verifica que la malla esté completamente inclinada.
- **Malla no está inclinada.** En este evento, se verifica que la malla esté en su posición horizontal anterior a su inclinación.
- **Bajando bandeja.** En este evento, se verifica que la bandeja se haya bajado.
- **Se procesa la imagen, pero el resultado no es apropiado.** En este evento, se procesa y se verifica la calidad de la foto tomada y si esta no es lo suficientemente buena.

- **Se procesa la imagen, pero el resultado es apropiado.** En este evento, se procesa y se verifica la calidad de la foto tomada y si es lo suficientemente buena.
- **Espera un tiempo para la bandeja se desplace verticalmente.** En este evento, el sistema espera un tiempo a que el motor logre desplazar completamente la bandeja y malla.
- **Enviando información.** En este evento, el sistema realiza la comunicación inalámbrica para enviar el reporte de resultados a los encargados humanos del monitoreo del criadero de camarones.

6.3. Identificación de acciones

Las acciones están íntimamente relacionadas con los actuadores como los servomotores, pero también pueden accionar algún periférico.

- **Motor.** En esta acción, se cambia el valor asociado al estado del motor, el cual es '1' para encendido y '2' para apagado.
- **Camarones en bandeja.** En esta acción, se coloca en '1' el valor de la variable asociada al led que indica si hay camarones en la bandeja. El led se enciende.
- **Estado de imagen.** En esta acción, se cambia el valor de la variable asociada a los leds que indican el estado de la imagen. Este puede ser recolectada.º "procesada", y, según sea el caso, se encenderá uno u otro led.
- **Tiempo_foto.** En esta acción, se espera un tiempo antes de tomar la foto.

7. Simulación en StateFlow (MATLAB)

Con base en lo planteado previamente, se procede a mostrar la implementación de las identificaciones realizadas por medio del software de simulación MATLAB en su entorno para máquinas de estados finitos, Stateflow.

7.1. Dashboard

A continuación, se muestra la totalidad del Dashboard que ha sido implementado para el manejo de datos por parte del usuario. Como se observa en la figura 9, se tienen dos secciones claramente diferenciadas a la izquierda y derecha. Consisten en los sectores de entradas y salidas de la simulación propuesta respectivamente.

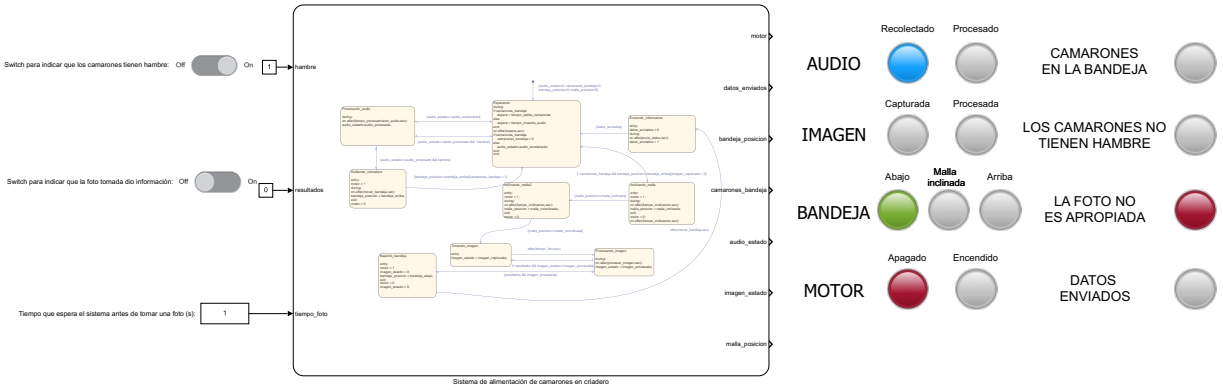


Figura 9: Dashboard de la máquina de estados finitos del sistema propuesto en Stateflow

Por otro lado, en la figura 10, se muestran las variables utilizadas en el Stateflow, tanto para las entradas y salidas, como de manera local.

TYPE	NAME	VALUE	PORT
	hambre		1
	resultados		2
	tiempo_foto		3
	motor	0	1
	datos_enviados	0	2
	bandeja_posicion		3
	camarones_bandeja		4
	audio_estado		5
	imagen_estado		6
	malla_posicion		7

(A) Variables de entrada/salida

TYPE	NAME	VALUE	PORT
	espera		
	tiempo_salida_camarones	5	
	tiempo_procesamiento_audio	3	
	tiempo_muestra_audio	3	
	mover_bandeja	4	
	tiempo_inclinacion	4	
	imagen_capturada	1	
	imagen_procesada	2	
	procesar_imagen	3	
	envio_datos	3	
	bandeja_arriba	1	
	bandeja_abajo	0	
	audio_recolectado	1	
	audio_procesado	2	
	malla_inclinada	1	
	malla_noinclinada	0	

(B) Variables locales

Figura 10: Variables de entrada, salida y locales del Stateflow

En la figura 11, se muestra un detalle de la sección de entradas del sistema simulado. En ella, se tiene el ingreso del tiempo que el sistema espera para la captura de una fotografía. Además, presenta dos switches para emular dos escenarios particulares, el hambre de los camarones, que se reflejaría en la muestra de audio tomada en el sistema real, y la calidad de la fotografía, la cual puede ser rechazada por el sistema si no se obtienen resultados concluyentes.

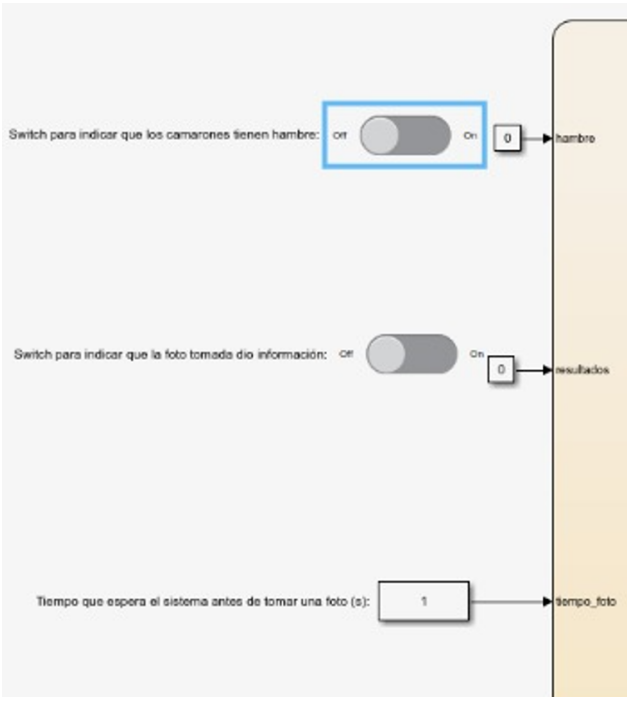


Figura 11: Entradas de la máquina de estados en Stateflow

Luego, en la figura 12, se muestra el detalle de la sección de salidas del sistema simulado. Aquí, se han colocado indicadores de lámpara para volver mucho más visual el comportamiento del cambio entre estados. Se tienen los pasos entre recolectar y procesar tanto audio como imágenes, así como la posición de la bandeja, el estado actual del motor que la desplaza y el reflejo de las entradas con switches que ya fueron descritas previamente.

Figura 12: Salidas de la máquina de estados en Stateflow

7.2. Funcionamiento

Para describir el funcionamiento, es pertinente mostrar los estados como tales implementados en Stateflow. Como se muestra en la figura 13, todas las condiciones de cambio de estado han sido marcadas de acuerdo con los eventos que fueron definidos en 6.2.

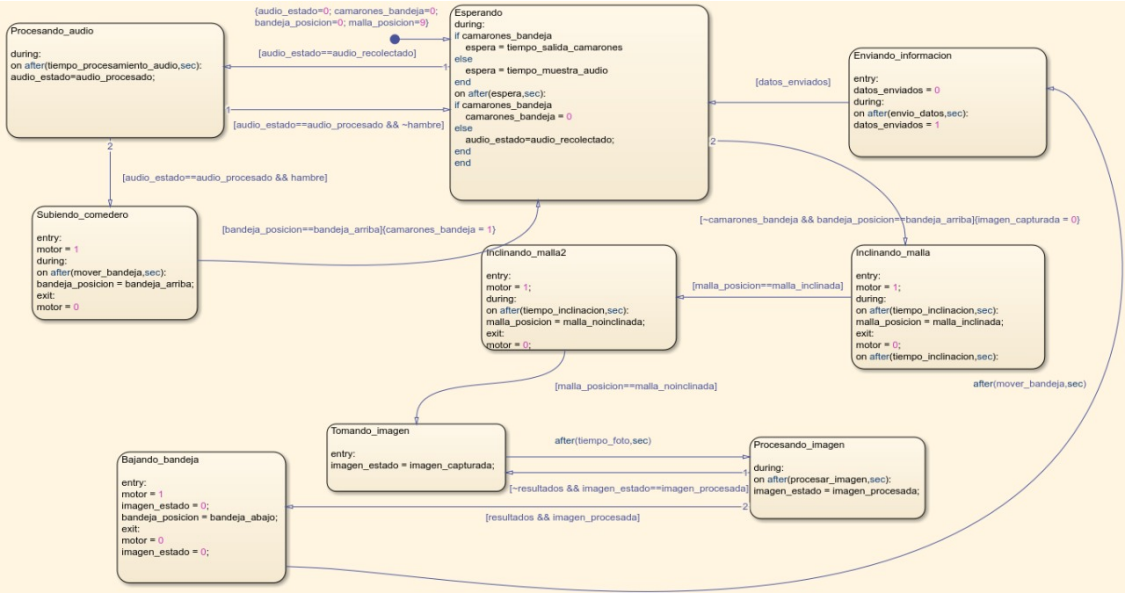


Figura 13: Estados de la máquina de estados en Stateflow

8. Módulos o subsistemas

La propuesta ha considerado varios aspectos sobre el modelo original cómo el alto costo de los micrófonos sumergibles, la necesidad de controlar precisamente la posición de la bandeja y demás aspectos. Por lo trato, la propuesta revisada se encuentra compuesta por los módulos de:

1. Balanza conformado por el módulo HX711, una celda de carga y un Arduino Pro Mini para recolectar las lecturas.
2. Actuación de motores DC con caja reductora para incrementar su torque conformado por los motores mismos, su driver y un Arduino Pro Mini para su control.
3. Módulo de visión compuesto por cámara Raspberry
4. Comunicaciones compuesto por el módulo WiFi embebido dentro del Raspberry Pi 3

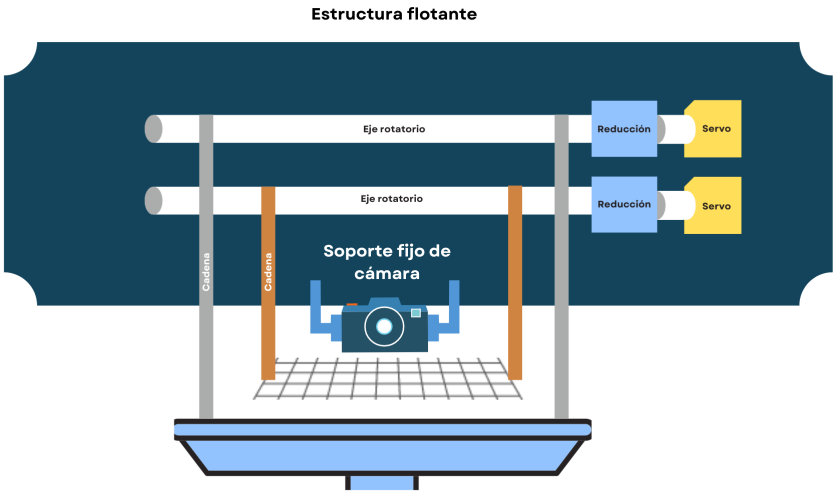


Figura 14: Esquema general de la propuesta

Estos subsistemas se puede apreciar en mayor detalle en el siguiente diagrama de bloques.

8.1. Diagramas de bloques

Dentro del diagrama se muestra las conexiones mediante protocolos de comunicaciones o conexiones físicas como alimentación que se tiene entre los distintos subsistemas. Se eligió una tarjeta Raspberry Pi 3 por su capacidad de procesamiento de imágenes, tener un módulo WiFi integrado que facilita el apartado de comunicaciones y por su fácil disponibilidad. Por otro lado, la tarjeta Arduino Pro Mini reduce la carga que podría realizar la Raspberry respecto a los sensores como las galgas y los encoders aparte que se encarga del control de los motores DC de manera independiente. Esto también aporta robustez sobre el sistema porque si un sistema deja de funcionar, la tarjeta restante mantendrá su operación. Sobre todo, si el control de motores falla con el Arduino, el Raspberry podría enviar una señal de mantenimiento urgente.

Todos los sistemas interconectados son alimentados por una batería LiPo como fuente DC que necesita tener como salidas 5V para las tarjetas y 12V luego de pasar por un conversor para alimentar a los motores DC.

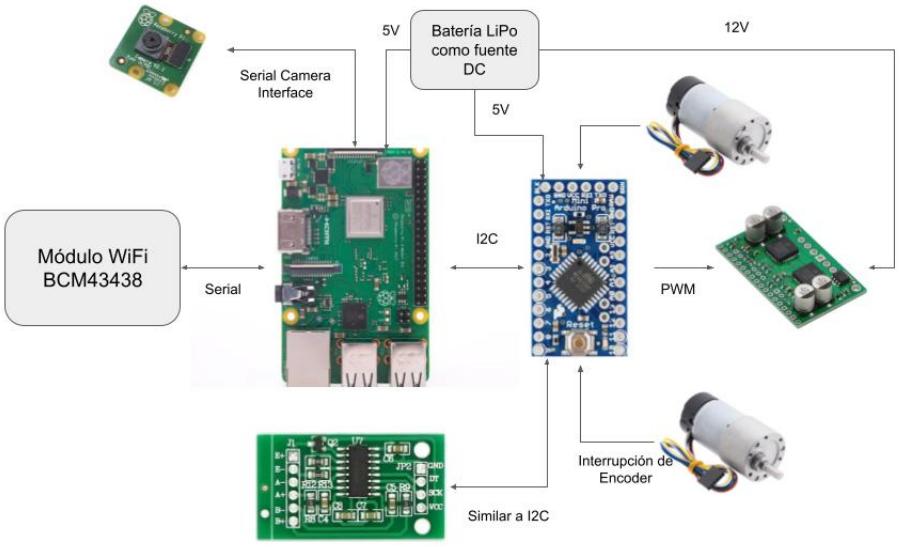


Figura 15: Módulos interconectados de la propuesta

9. Sensores y actuadores

Cada sensor en conjunto con un actuador o de forma separada se encuentra integrado en los subsistemas de la propuesta. La lista original de sensores fue reducida en vista que ya no se utilizará el hidrófono por la extensión de tiempo disponible para la realización del proyecto. Asimismo, se ha optado utilizar una tarjeta Raspberry PI 3 en vez del ESP32 CAM por la disponibilidad del equipo en el laboratorio. Esto conlleva a que se debe considerar dentro de los componentes una cámara externa compatible con Raspberry PI. Este mismo criterio de disponibilidad fue aplicado al resto de sensores e inclusive actuadores.

9.1. Sensores

1. **Cámara v2 Raspberry Pi 8 Mpx**: Esta cámara cuenta con las especificaciones necesarias para realizar segmentación de objetos por espacio de color, una técnica de procesamiento de imágenes que se debe aplicar para detectar cuanto alimentó se consumió. La resolución base de la cámara es de 8 Mpx 1080p a 30 FPS y puede reducirse si se desea una mayor cantidad de FPS. Para esta aplicación, es más importante la resolución que la cantidad de cuadros porque la toma de información de la bandeja no requiere una velocidad alta. De todas maneras, en todas las configuraciones, la tarjeta Raspberry PI no tendrá problemas con la carga computacional que conlleva. La cámara cuenta con su propio conector CSI por donde se ejecuta un protocolo de comunicación especializado (Camera Serial Interface) y se envía su alimentación.

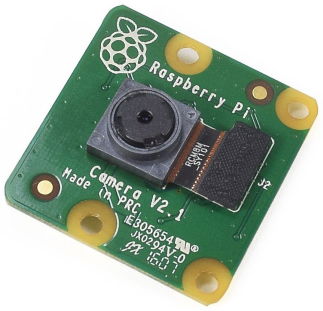


Figura 16: Cámara v2 Raspberry PI

2. **Módulo WiFi BCM43438**: Este módulo embebido en la tarjeta Raspberry permite el envío y recepción de datos a través de conexión WiFi. La tarjeta ya cuenta con todos los drivers necesarios para el envío de tramas por este medio. Cabe resaltarse el alcance de este sensor que puede llegar a comunicarse a 20 m del hotspot a 28 Mbit/s como se comprobó en la siguiente serie de test. Esto también indica que se puede forzar mayores distancias bajo el compromiso de velocidad de data. Este requerimiento debe evaluarse porque el envío de imágenes por WiFi necesita velocidades considerables, pero la frecuencia de envío será una vez cada 3-4 horas.

Figura 17: Tarjeta Raspberry PI 3 con su módulo WiFi ubicado en el centro

3. **Encoder magnético de doble canal** Este encoder permite la lectura de posición del motor DC con una resolución de 64 cuentas por revolución y ya se encuentra acoplado al actuador. Utiliza dos sensores Hall para detectar el campo magnético generado por el motor. Mediante dos interrupciones asociadas a cada sensor se identifica si se completó una vuelta en sentido horario, antihorario, o sí el motor está de parada. Se debe considerar que el encoder mide la cantidad de cuentas antes de la etapa de reducción, así que se debe escalar sus mediciones.

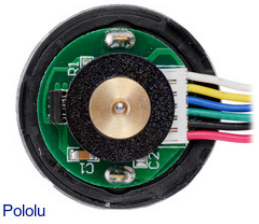


Figura 18: Encoder de doble canal por sensor Hall de resolución 64 C/R

4. **Módulo HX711 + Galgas extensiométricas**: El módulo permite la lectura de una galga extensiométrica sometida a tensión, utilizada comúnmente en balanzas. Contiene dentro de sus componentes un conversor ADC Delta Sigma con una resolución de 24 bits y que puede llegar a generar lecturas a 80 Hz. También posee una etapa de ganancia de 32, 64 o 128 sobre la señal analógica original en vista que esta es muy débil para convertirla. Para su funcionamiento se requiere alimentar el módulo con 5V digital. Por otro lado, su método de comunicación es similar a I2C donde se tiene un pin CLK por donde se envían los pulsos que establecen el valor de ganancia y coordina la generación de muestras, y el pin DT por donde sale el valor digital convertido.

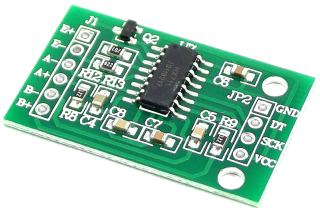


Figura 19: Módulo HX711

Aparte del módulo, ya se cuenta con las galgas colocadas en una celda de carga y conectadas a un puente de Wheatstone por doble puente para la conexión inmediata. Por lo tanto, sobre las entradas del módulo HX711 irán las conexiones de la celda de carga, como se realizaron durante el laboratorio 4 del curso.

Figura 20: Celda de carga con galgas alineadas bajo resina

9.2. Actuadores

1. **Motores Polulu DC de 12V 37Dx73L con reducción 100:1 + Driver MC33926**

   Este fiable motor DC alimentado por 12V puede llegar a velocidades de 100 rpm sin carga y generar torques de 3.3N. Es necesario utilizar una caja de reducción adicional para aumentar el torque generado para levantar la balanza y las mayas sumergidas. Este motor consume una corriente de 0.2A sin carga y puede llegar hasta 5.5A bajo la máxima exigencia. Así mismo, cuenta con una etapa de reducción de 100:1, lo que significa que a la salida del motor deben leerse 6533 cuentas por revolución. Varios miembros de este equipo han realizado pruebas utilizando estos motores e identificaron que gracias a la caja de reducción que posee y su tamaño reducido, el control de posición y velocidad obtiene mejores resultados que para motores más grandes. Es por ello, que se ha optado por este motor y tener la caja de reducción fuera del actuador.

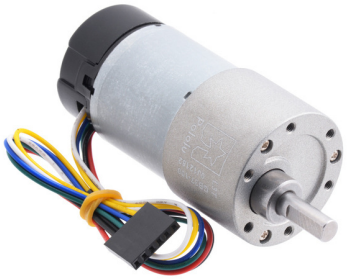


Figura 21: Motor Polulu DC de 12V con encoder incluido

Para asegurar el funcionamiento y un control más preciso del motor DC, es necesario utilizar un driver, en este caso se seleccionó el MDD10A de Cytron. Se encarga de transmitir las señales PWM desde el microcontrolador hasta los motores, definir el sentido de giro, a la par que los alimenta con 12 V y los protege de sobre corrientes cuando se someten bajo cargas muy exigentes o se detiene bruscamente. Puede trabajar sobre 10A con un pico máximo de 30A, lo que brinda un margen bastante amplio para las pruebas, y maneja señales PWM de hasta 20kHz. Además, cuenta con botones para realizar pruebas manuales sobre el motor sin necesidad de programarlo, lo cual resulta útil durante el proceso de prototipado.

Figura 22: Driver Dual Channel 10A 5V-30V de Cytron

10. Diseño mecánico

10.1. Selección de materiales

En lo que respecta a la selección de un material para la protección del sistema electrónico a implementar, existen diversas aproximaciones al tratamiento de este tipo de casos. En primer lugar, cabe resaltar que la tecnología de electrónica marítima ha comenzado a hacer énfasis en que no es necesario proteger al 100 % el sistema de la humedad para conseguir una operación apropiada por un tiempo prudente [9]. En ese sentido, el enfoque se centra en encontrar la verdadera causa del mal funcionamiento de un circuito, que por lo general se vincula a la corrosión de los componentes y sus conexiones [10]. Así, resulta más importante instalar el sistema electrónico de manera que mecánicamente esté separado de la fuente de agua. En este caso, del hábitat de los camarones como tal.

Se debe considerar que el modelo de alimentador planteado consiste en una especie de boya que se instala en un punto específico del criadero. De esta manera, el material del cual se encuentre construida debe permitir que se consiga la forma cilíndrica deseada, así como no contar con un excesivo peso y que sin deteriorarse pueda albergar al resto de componentes electrónicos que permiten la operación del sistema de acuerdo con los principios que ya fueron explicados en el planteamiento del diseño, así como en la sección 11 correspondiente a la metodología. En ese sentido, se describen algunos aspectos en torno a los materiales que frecuentemente se observan en la elaboración de boyas.

- **Polietileno de alta densidad (HDPE).** Suele ser más observado en la elaboración de envases plásticos descartables. Esto lo vuelve accesible y de poco peso. Asimismo, presenta un alto grado de flexibilidad y resistencia tanto térmica como química. Además de la industria alimenticia, también se le emplea para tuberías de distribución de agua. Presenta facilidad de personalización sobre su superficie y resulta apropiado para el reciclaje.

- **Policloruro de vinilo (PVC).** Se caracteriza por una considerable resistencia mecánica, tanto a impactos como esfuerzo. Sin embargo, puede llegar a ser flexible y moldeable sin necesidad de aplicarle elevadas temperaturas. Es resistente al agua y particularmente empleado para fines de aislamiento y protección. A todo lo anterior, se añade un bajo costo de instalación dada su frecuencia de uso y prolongado tiempo de vida.

- **Poliestireno de alto impacto (HIPS).** Se trata de un material usado en impresión 3D y para artículos desechables que no se encuentran vinculados directamente con la industria alimentaria en la mayoría de los casos. Es altamente moldeable, ya que tiene la facilidad de replicar detalles de una geometría asignada por medio de un molde particular. De este modo, resulta apropiado para generar varias copias de una estructura que conforme.

En función de lo brevemente descrito acerca de los materiales más frecuentes observados en la fabricación de boyas según la National Oceanic and Atmospheric Administration (NOOA) en el Marine Debris Program, se tiene que cada uno de los materiales presenta alguna ventaja particular sobre el otro. El HDPE permite un fácil reciclaje, el PVC tiene un prolongado tiempo de vida y el HIPS replica moldes fielmente. Sin embargo, para fines de

duración de la implementación y una resistencia mecánica óptima para el diseño del sistema se opta por el PVC. Además, este es frecuentemente utilizado para aislamiento de cableado eléctrico, con lo que se vincula al propósito de protección del sistema electrónico frente a la humedad del entorno. Asimismo, aporta con su elevada resistencia a la corrosión para el tipo de ambiente en el que se instala el sistema de alimentación.

10.2. Proceso de moldeado

Para el diseño propuesto, se está optando por un material que requiere un proceso de moldeado para plásticos dado que se trata de PVC. Existen varias formas de hacer que se adapte a la forma de un molde en particular. Sin embargo, el procedimiento más frecuente resulta ser el moldeo por inyección. Este es empleado en piezas que requieren una alta velocidad de fabricación, lo que implica que podrían estructurarse varios sistemas en un intervalo corto de tiempo.

Si bien este procedimiento incrementa en cierta medida los requerimientos de fabricación, ya que se necesita acero inoxidable para asegurar la calidad de los productos luego de pasar por los moldes, permite un ajuste idóneo para el tipo de geometría deseado para la cápsula física en la que se instala el resto de componentes del sistema de alimentación propuesto.

11. Metodología

Este apartado se encarga de profundizar en cómo cada subsistema se encarga de desempeñar su labor. Se incluirá tanto su fundamento matemático cómo comentarios sobre la implementación real que puedan mejorar el rendimiento de la tarea.

11.1. Métodos utilizados

- **Control PID de posición para motores DC**
  Al enviar una señal PWM con cierto Duty Cycle hacia los motores, se espera un movimiento limpio a cierta velocidad proporcional al PWM. No obstante, la realidad nos muestra que enviar comandos directos sobre los actuadores sin retroalimentación tiene pobres resultados. Aparte, en caso de disturbios, los actuadores no responderán adecuadamente. En consecuencia, se debe aplicar un control PID de posición sobre los motores DC para lograr elevar las mallas y balanza en un tiempo razonable hacia una altura deseada.

  Este control se basa en utilizar la señal retro alimentada de posición y calcular que tan alejada está de la deseada, denominado a partir de la altura que se busca recorrer. A esta diferencia se le conoce como error $e = \text{referencia} - \text{posición medida}$. Esta misma variable es multiplicada por una ganancia proporcional y se obtiene una primera ley de control denominada control **P**.

  $$u_P = K_p e \tag{1}$$

  Para su implementación, se debe especificar el rango de valores que puede tomar para que sean reescaladas a -255 a 255 según lo requiere la señal PWM y el manejo de

dirección dentro del Driver para cada motor.

Una desventaja de este control es que solo se puede asegurar que el motor llegue a una posición estable más no que siga la referencia. En sí, el control **P** suele tener un error en estado estacionario. Para reducirlo, se propone utilizar un término integral que va acumulando el error y lleva la respuesta a seguir la referencia con el menor error posible. Cómo las transformadas continuas como la integral no puede calcularse en un microcontrolador, se debe discretizar la operación para ser implementado. Esto se consigue mediante la integración de Euler que aproxima la integral por una suma acumulada de muestras pasadas con la muestra actual multiplicada por un tiempo de muestreo. Entonces, se define la integral discretizada del error tal que:

$$\int e\ dt\ \approx I = I + e(dT) \tag{2}$$

Considerando tanto la parte proporcional como integral, la ley de control se expande a:

$$u = P + I = K_p e + K_i(\sum e + e(dT)) \tag{3}$$

Este control ya puede generar resultados satisfactorios para el seguimiento de posición en motores DC. Adicionando una parte derivativa se puede conseguir reducir las oscilaciones en la respuesta o los sobre impulsos que aparecen cuando la posición trata de seguir la referencia de forma abrupta. No obstante, se suele optar por utilizar un filtro digital pasa bajos para filtrar ruido y vibraciones frente al control derivativo porque se evita tener que ajustar una ganancia más.

Los filtros digitales puede ser implementados fácilmente cómo la suma ponderada de muestras actuales, pasadas y muestras filtradas. Cada coeficiente son calculados con base en la respuesta de frecuencia que se desea obtener, que en este caso es una curva que permite pasar todas las frecuencias por debajo de cierto umbral (frecuencia de corte). La respuesta de un filtro pasa bajos en transformada de Laplace se define tal que:

$$H(s) = \frac{Y_{filt}(s)}{X_{raw}(s)} = \frac{\omega_0}{s + \omega_0}$$

donde ω_0 está relacionado con la frecuencia de corte y es ajustable.

Para ser implementable en un microcontrolador se puede despejar ambas partes de la ecuación para obtener derivadas de la señal cruda y filtrada, y luego discretizar las derivadas con diferencias finitas para obtener una expresión a partir de muestras actuales y pasadas.

$$\dot{y_{filt}} + \omega_0 y_{filt} = \omega_0 x_{raw}$$

$$(1 + \omega_0)y[n] - \ y[n-1] \ = \ \omega_0 x[n]$$

$$y[n] \;=\; \frac{\omega_0}{(1+\omega_0)}x[n] - \frac{1}{(1+\omega_0)}\; y[n-1]$$

Esta última expresión ya podría codificarse en un microcontrolador. Al considerarse más términos, se puede mejorar la respuesta del filtro. En el siguiente video se propone un filtro pasa bajos de mayor orden para el control PI del mismo tipo de motores con frecuencia de corte de 25 Hz. La siguiente expresión ya tiene la forma discreta del filtro con los coeficientes calculados y fue testeada en una plataforma Arduino.

$$v_{filt}[n+1] \;=\; 0{,}854 * v_{filt}[n] \;+\; 0{,}0728 * v_{raw}[n] \;+\; 0{,}0728 * v_{raw}[n-1] \qquad (4)$$

Esta misma función de un filtro digital podría realizar con un análogo con el compromiso que los valores nominales de los componentes pasivos no varíen mucho durante el tiempo de operación. También ocurren casos donde las propiedades dinámicas del sistema ya ofrecen cierta robustez ante oscilaciones cómo ocurre cuando hay una gran carga inercial en el motor.

- **Lectura de galgas extensiométricas**

  Las galgas al ser elemento piezoeléctricos que cambian su resistencia dependiendo del esfuerzo al que se encuentre sometidas, permiten hacer lecturas sobre cuanta carga o peso se les está aplicando. La variación de la resistencia (ΔR) de la galga depende tanto de la deformación, el valor actual de la resistencia (R), y el factor de galga (GF) que proviene de la fabricación y el material de la galga.

$$GF = \frac{\Delta R/R}{\epsilon}$$

La disposición de galgas que se encuentra en la celda de carga utilizada se denomina half-bridge, donde una galga se comprime y la otra se expande al aplicar una carga en su extremo. Estas galgas son conectadas a un puente de Wheastone para la medición precisa de la variación de su resistencia.

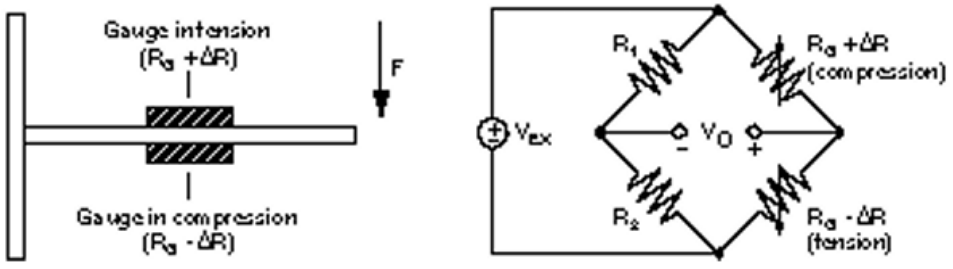


Figura 23: Configuración de las galgas para half-bridge en celda de carga y puente de Wheatstone [11]

Para encontrar una relación entre el voltaje medido del puente y la deformación, solo se debe encontrar en que caso el puente se encuentra balanceado y para que valor de

resistencia de las galgas corresponde. Al final, este resultado se resume en la siguiente fórmula:

$$\frac{V_{medido}}{V_{cc}} = \frac{-GF\epsilon}{2} \tag{5}$$

Esta aplicación se concretará más en la medición de pesos que puede conseguir sin necesidad de utilizar las ecuaciones a partir de medir pesos conocidos y crear un factor de escalamiento entre la lectura del HX711 y el peso real. No obstante, si se desea interpretar las lecturas en términos de deformación de las galgas si es necesario conocer la ecuación anteriormente mencionada.

- **Segmentación por espacio de colores**

  La información que aportan las imágenes es una combinación de formas geométricas y diferentes gamas de colores. El ojo identifica objetos o instancias al observar un cambio de colores en su delimitación o borde. De forma natural, el ojo genera colores por la suma de las distintas frecuencias de la luz incidente. En particular, tiene foto receptores que detecta luz que corresponde a las bandas rojo(R), verde(G) y azul(B). Es de esta inspiración biológica que se crea el espacio de colores RGB donde cada color es generado por la suma ponderada de los tres componentes. Las cámaras también obtienen fotos utilizando este mismo espacio de colores. Para extraer objetos en particular dentro de la imagen se podría hacer un filtro que permita pasar solo una intensidad de color. A este proceso se le conoce como segmentación de color.

  No obstante, esta técnica en RGB muchas veces falla porque los objetos que buscamos segmentar no están compuestos de un único color, sino más bien de una textura que contiene varias componentes muy cercanas a un color particular. Es por eso que existen, otros espacios de colores que crean los colores a partir de la resta de sus propios componentes como Cyan, Magneta y Yellow. También se pueden crear espacios de colores donde la saturación sea un parámetro para el color cómo ocurre con HSV(Hue, Saturation y Value) o HSL(Hue, Saturation y Lightliness). Es justamente este último espacio de color que ha mostrado resultados prometedores para la segmentación de color de la comida de los camarones como se observa en [6].

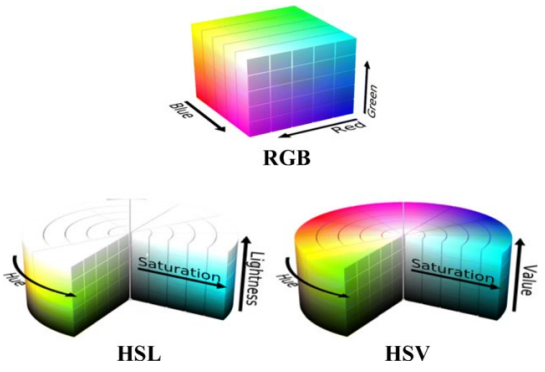


Figura 24: Representación de los espacios de colores RGB, HSL y HSV

Para conseguir pasar el espacio de color a RGB a HSV se necesita hacer una serie de transformaciones condicionales que dependen que cuál canal (R, G, B) tiene el mayor valor y el mínimo valor.

$$H = \begin{cases} 0 & if\ max = min \\ 60 \times \frac{G-B}{max - min} & if\ max = R \\ 60 \times \frac{B-R}{max - min} + 120 & if\ max = G \\ 60 \times \frac{R-G}{max - min} + 240 & if\ max = B \end{cases} \tag{6}$$

$$L = \frac{1}{2}(max + min) \tag{7}$$

$$S = \begin{cases} 0 & if\ max = min \\ \frac{max - min}{max + min} & if\ L \leq \frac{1}{2} \\ \frac{max - min}{2 - (max + min)} & if\ L > \frac{1}{2} \end{cases} \tag{8}$$

Una vez hecha la transformación entre espacios, se puede proceder a segmentar la comida de los camarones. En [6] se obtuvieron buenos resultados al segmentar con una máscara en el canal Hue que solo aceptara valores entre $0 \leq H \leq 60$ y $109 \leq H \leq 180$. Esta máscara implementada simplemente sería una condicional if aplicada a elemento del array que corresponde al canal Hue.

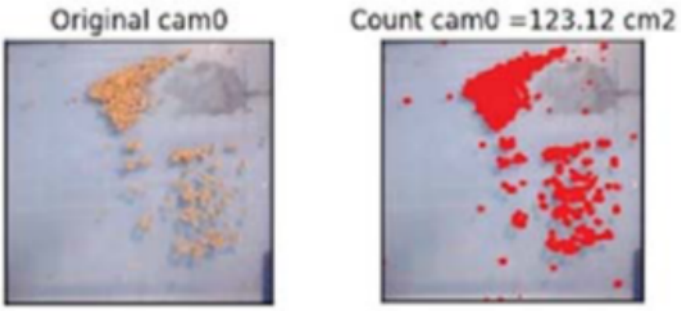


Figura 25: Resultados de la segmentación de color en la comida de camarón obtenidos en [6]

Tanto la transformación de espacio de color de RGB a HSV sobre la imagen capturada y la segmentación del color de la comida del camarón se pueden realizar utilizando las librerías OpenCV mediante un script de Python que se carga en la tarjeta Raspberry Pi 3.

11.2. Vínculo con el diseño propuesto

El diseño propuesto aprovecha las capacidades de la segmentación de color para identificar cuanta comida ha sido consumida por los camarones y contrasta esta información con el peso medido por el sistema de balanzas. Finalmente, el correcto posicionamiento del comedero tanto para capturar fotos con suficiente claridad de la escena y levantar la balanza dependen netamente del control PI de posición aplicado sobre los motores DC

12. Diseño CAD e interfaz de usuario

12.1. Celda de carga

El modelo de celda de sensor de celda de carga utilizado fue WSS-10kg. A continuación, se muestra el modelo CAD en inventor el cual fue extraído del sitio con este hipervínculo.

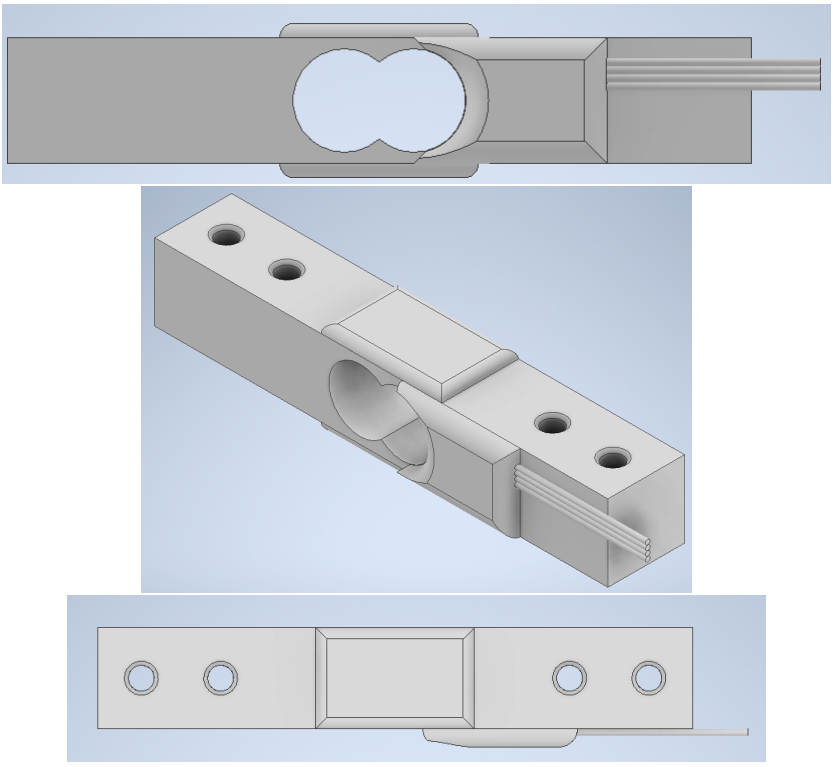


Figura 26: Vista frontal, isométrica y superior del modelo CAD del sensor de celda de carga

De acuerdo con la hoja técnica de la celda de carga de código de parte OBUG-10kg-0.4-000, si bien no es el modelo exacto utilizado en el laboratorio, se pudo tener una mejor idea de las dimensiones para el sensor. A continuación se muestra el dibujo técnico del modelo, acorde la norma ISO.

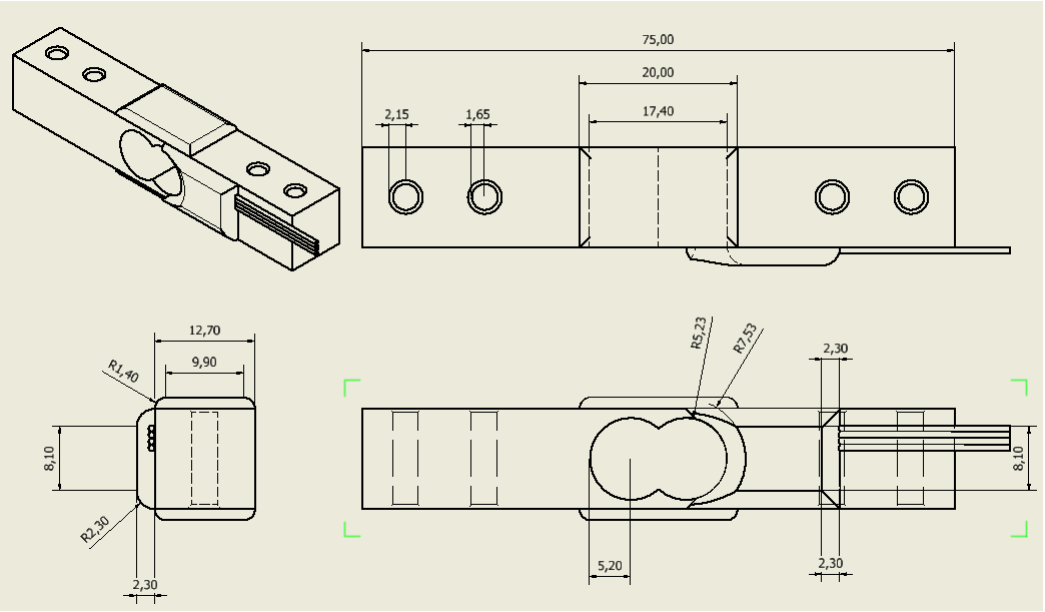


Figura 27: Dimensiones para la celda de carga de acuerdo a su capacidad de medición (Elaboración propia)

12.2. Módulo HX711

Para el diseño del módulo HX711 se utilizó el programa Fusion 360 dada su facilidad e integración a EAGLE, que fue en donde se realizó la placa de circuito impreso. De esta manera, se incluyeron las piezas 3D de los resistores, capacitores, transistor NPN y los pinhead; sin embargo, no se encontró el archivo .STEP del microcontrolador del HX711 de manera gratuita en línea, por lo que solo se consideró el espacio necesario para este componente con lo cual se tiene una idea del tamaño que ocupa cada componente en este módulo. Para una mejor visualización del diseño se puede observar en el siguiente hipervínculo.

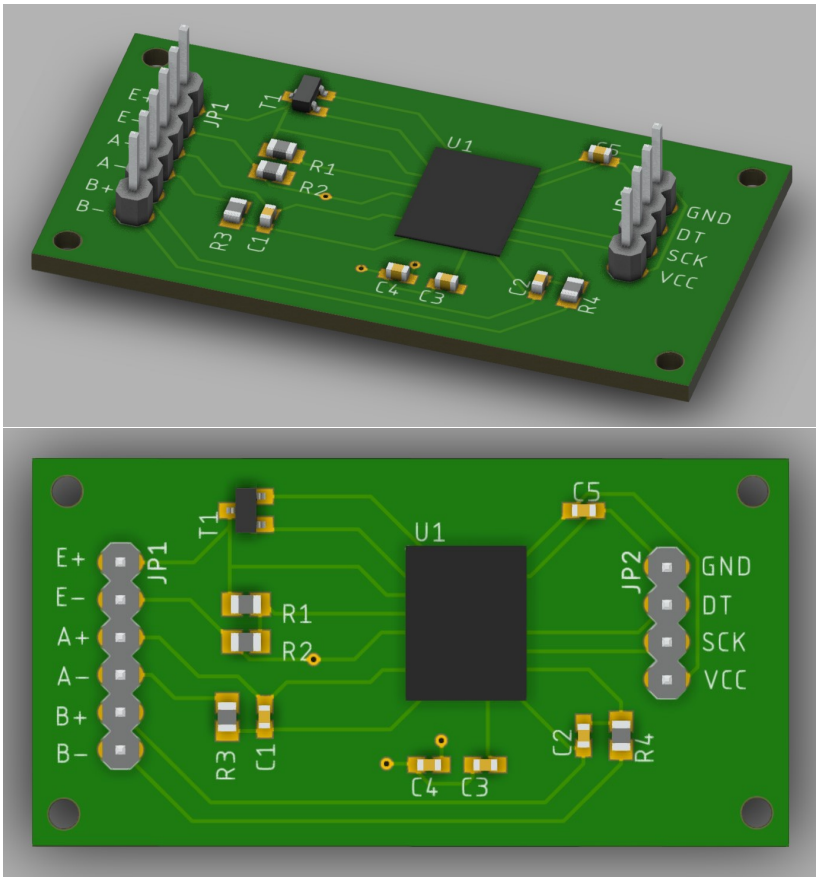


Figura 28: Diseño CAD para el HX711 en Fusion 360

12.3. PCB para el módulo HX711

En el caso del diseño del printed circuit board, este se realizó en el software Autodesk EAGLE. Para ello, se empezó definiendo el esquemático para lo cual se siguió como modelo la estructura del PCB físico que fue entregado al grupo, ya que el esquemático del datasheet contaba con un menor número de componentes y de pines. De esta manera, se utilizaron etiquetas para poder realizar un diseño más organizado y conectar todos los componentes. Cabe resaltar que se consideraron conectores de tipo pin y resistores/capacitores/transistor con montura superficial, tal y como se cuenta en el módulo HX711 físico.

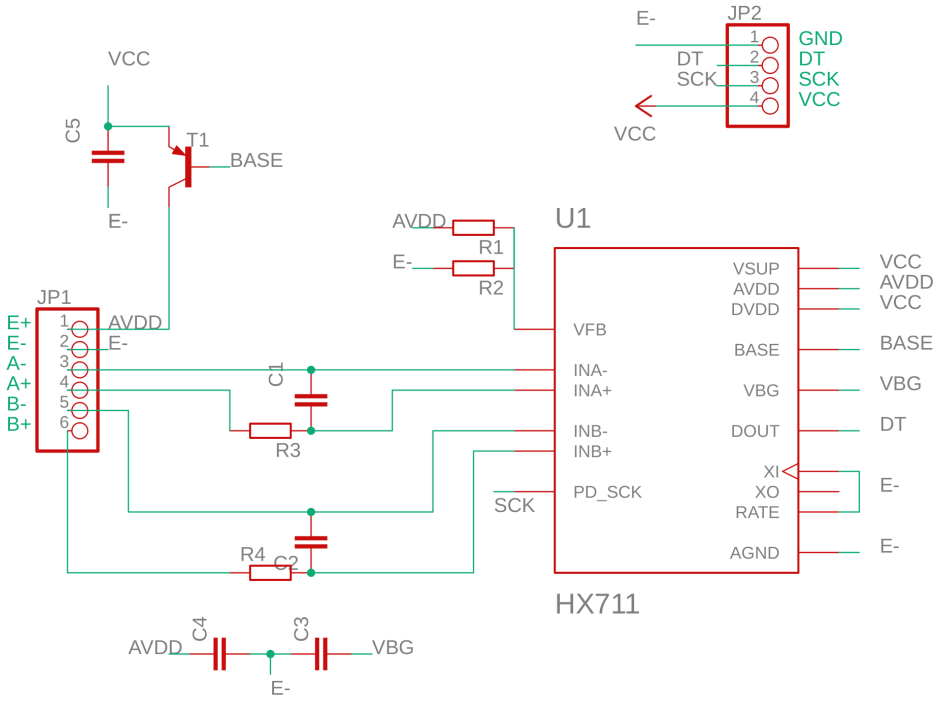


Figura 29: Esquemático para el HX711

Una vez obtenido el esquemático con todos los componentes conectados, se pasó a la vista de Board en EAGLE. Se ubicaron los componentes de la manera más óptima posible y se realizaron las conexiones en la capa superior (top) de la placa. Se consideró utilizar ambas caras de la PCB, unidas a través de vías, para evitar cruces entre las pistas o cercanías innecesarias entre las rutas; y se añadieron orificios en los extremos para poder fijar la tarjeta en caso sea necesario.

Es necesario resaltar que para definir el ancho de las pistas se consideró una calculadora online considerando una temperatura ambiental de 25°C y un rango máximo de ±5°, así como la corriente indicada en el datasheet del módulo (1.5 mA en operación normal), y un espesor de $1oz/ft^2$.

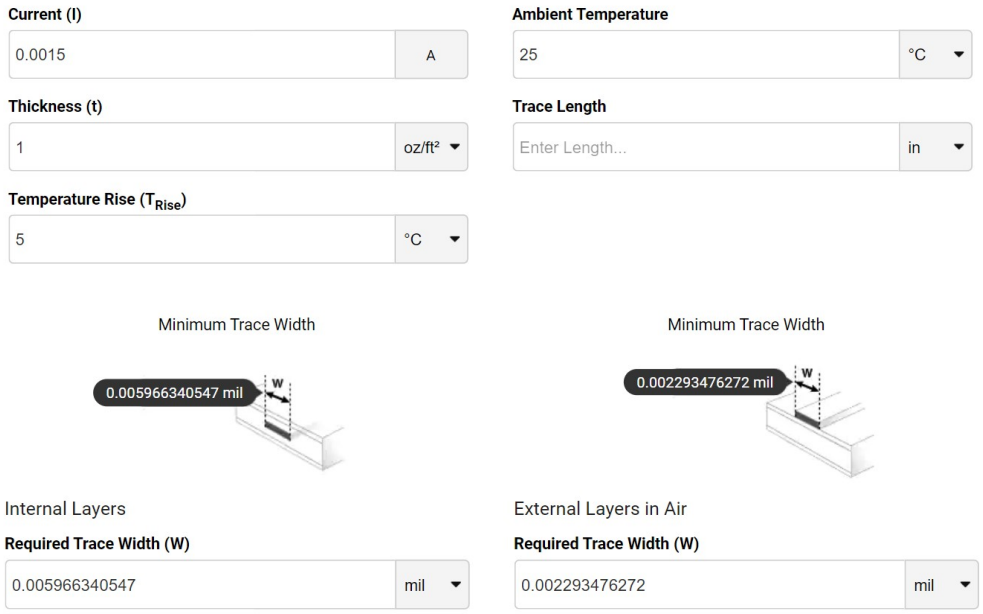


Figura 30: Cálculo del ancho de pistas

Así, el ancho requerido es considerablemente pequeño, de aproximadamente 0.006 mm. Esto se debe a que al no ser un circuito de potencia, no disipa mucho calor, por lo que se tomó en cuenta un valor mayor para el ancho de pistas de 6 mm. También, se evitaron los ángulos de 90° al momento de doblar alguna de las pistas y una separación considerable entre estas mismas siempre que fuera posible.

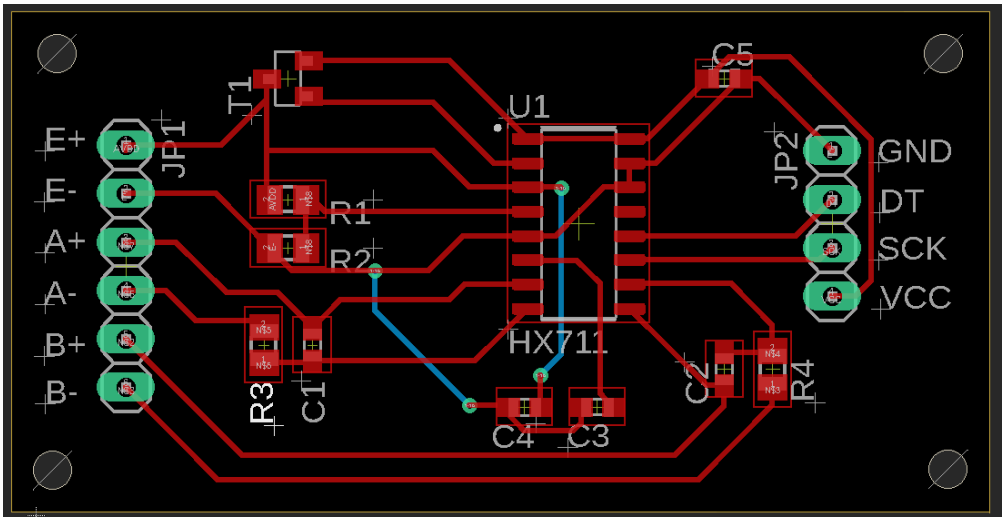


Figura 31: Diseño del printed circuit board para el HX711

Además, se muestra el diseño de manufacturing, en donde se puede visualizar una vista previa de cómo quedarían las pistas impresas sobre la placa.

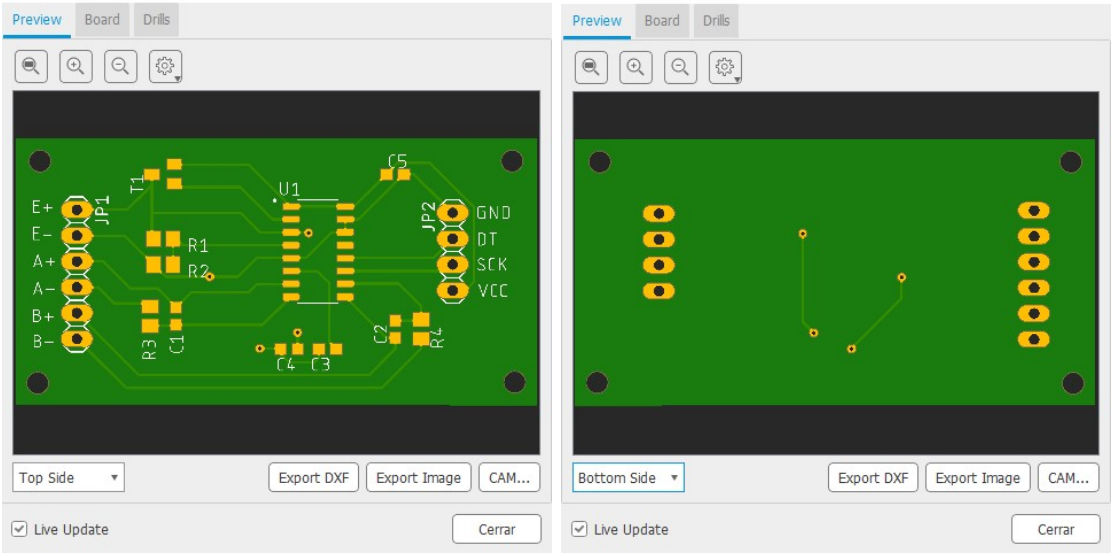


Figura 32: Manufacturing del PCB para el módulo HX711

12.4. Interfaz de usuario

Se diseñó una interfaz de usuario amigable e intuitiva, la cual se puede ser visualizada en este hipervínculo, en la que se facilita el acceso a los datos más relevantes de cada sistema.

En la pantalla de inicio (Figura 33A), se permite acceder con un correo electrónico y contraseña. Luego, se redirige al usuario a una pantalla con los sistemas de monitoreo (Figura 33B) en los que, al pasar el ratón por encima de cada uno, se da a conocer las mediciones más recientes según cada ubicación a modo de ventana emergente. Asimismo, esta ventana permite ampliar los datos históricos (Figura 33C) para cada sensor con gráficas de cada parámetro y la última fotografía capturada.

Cabe resaltar que en la segunda pantalla también se reportan los sistemas caídos o que no envían datos en cierto tiempo para alertar al usuario de la falla y proceder con el mantenimiento del mismo.

(A) Pantalla de inicio *(B) Sistemas de monitoreo* *(C) Datos históricos*

Figura 33: Cuadros de las vistas de la interfaz de usuario

13. Resultados de experimentación con sensores

13.1. Principio de transducción de una celda de carga

Una celda de carga permite la recopilación de información de deformación del metal que la conforma con base en el uso de galgas extensiométricas. Estas se constituyen como resistencias variables que se adhieren a una superficie deformable que posteriormente será sometida a alguna fuerza externa que provoque estrés mecánico por medio de tensión o compresión.

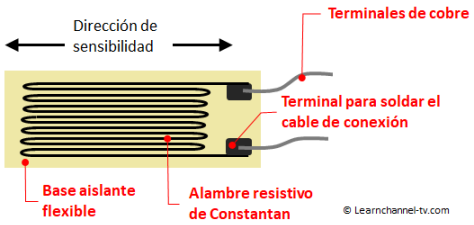


Figura 34: Diagrama de una galga extensiométrica

Como se observa en la figura 34, el sensor de deformación presenta una grilla de material resistivo que se deforma junto con la superficie a la cual se encuentra adherida. En ese sentido, se sigue un principio de transducción piezorresistivo para la galga solamente, ya que su valor de resistencia se incrementa al someterla a tracción en su dirección de sensibilidad, mientras que se reduce al someterla a comprensión en esa misma dirección.

Sin embargo, la medición de su valor de resistencia no es suficiente como para contar con una precisión apropiada. Es por esto que la celda de carga configura una conexión de puente de Wheatstone con las galgas que presenta, de manera que se aproveche la transducción piezoeléctrica y sea viable obtener una lectura de diferencia de potencial a la salida del puente, como se muestra en la figura 36.

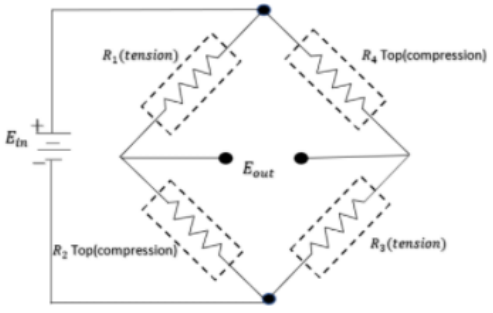


Figura 35: Diagrama de puente de Wheatstone completo

La denominación del puente depende de la cantidad de galgas que se encuentren involucradas en su configuración, siendo simple cuando solamente es una de las 4 resistencias, medio cuando se tienen 2 galgas extensiométricas y completo cuando la totalidad del puente está compuesto por estos sensores. De este modo, la variación de resistencia de la galga provoca una variación en la tensión de salida E_{out}. Así, la diferencia de potencial variable constituye el principio de transducción piezoléctrico una vez que se ha conformado el puente.

13.2. Funcionamiento del HX711

El módulo de transmisión de celda de carga HX711, de acuerdo con su hoja de datos [12], cuenta con una entrada SCK, por donde se reciben pulsos desde el microcontrolador, y una salida DT, por la cual se envían los datos de manera serial. De esta manera, se establece una comunicación que, como se mencionó previamente, es muy similar al I2C. Por un lado, el pin DT indicará si la data está lista para ser extraída cuando su valor está en alto, caso contrario, se encuentra en periodo de muestreo. Por otro lado, la cantidad de pulsos recibidos por el pin SCK determinará la ganancia utilizada para amplificar la señal analógica original de la manera en que se muestra en el cuadro inferior. Cada vez que se recibe un pulso, la data recibida por DT es desplazada un bit, empezando por el más significativo, y acabando con el menos significativo hasta completar la secuencia. Así mismo, el último pulso de cada secuencia colocará el pin DT en alto nuevamente.

PD_SCK Pulses	Input channel	Gain
25	A	128
26	B	32
27	A	64

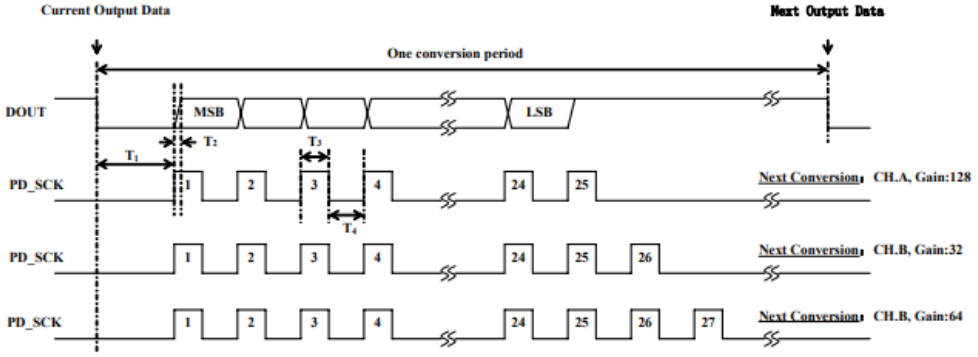


Figura 36: Secuencia de funcionamiento para pines SCK y DT

En cuanto al uso del canal A o B, dependerá del cableado realizado, dado que el sensor se carga cuenta con 4 cables distribuidos de la manera en la que se muestra en la imagen a continuación.

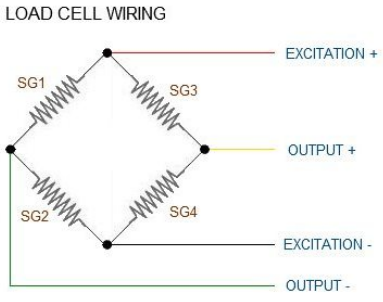


Figura 37: Conexiones del sensor al puente de Wheatstone interno

De acuerdo con esto, las conexiones entre los cables del sensor deben realizarse como se indica en la tabla si se quisiese utilizar el canal A o B. Cabe mencionar que para el propósito de nuestro proyecto, utilizamos una ganancia de 128, por lo que realizamos las conexiones para el canal A.

Cable del sensor	**Señal**	**Pin del HX711**
Rojo	Voltaje de excitación+	E+,VCC
Negro	Voltaje de excitación-	E-,GND
Verde	Amplificador-	A- o B-
Blanco	Amplificador+	A+ o B+

14. Diseño y construcción de la estructura mecánica del sistema

Para la construcción de la estructura mecánica del sistema, se están empleando las piezas disponibles en el kit Tetrix hechas de Aluminio 7005 con un espesor de 2 mm. A manera de región estructural base, se han acoplado 4 perfiles C en formación cuadrangular de modo que refuercen la estabilidad de la estructura ante la adición posterior del peso de los componentes electrónicos. Su disposición se observa mediante la figura 38.

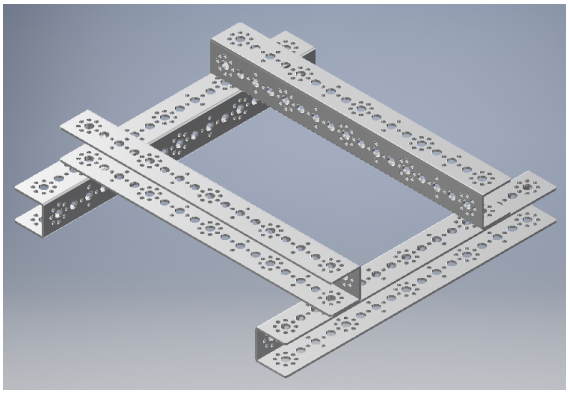


Figura 38: Disposición de 4 perfiles C a manera de base estructural

El sistema electrónico requería de una base adicional que pudiera encontrarse suspendida para disponer de un mayor espacio en torno a la instalación de los motores y demás componentes que conforman la transmisión de movimiento. Por tanto, se recurrió a la unión de dos canales de 32 mm (como se muestra en la figura 39) para constituir un total de 4 perfiles S que pudieran soportar esta plataforma para las plataformas de desarrollo y el puente H a ser utilizados.

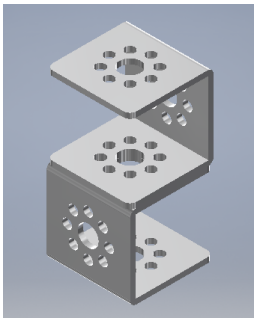


Figura 39: Disposición de perfiles S con base en unión de canales de 32 mm

Finalmente, se adicionaron dos placas planas de construcción de 64 mm x 192 mm. Entonces el resultado final del acoplamiento de todas las piezas resulta en el conjunto mostrado por medio de la figura 40.

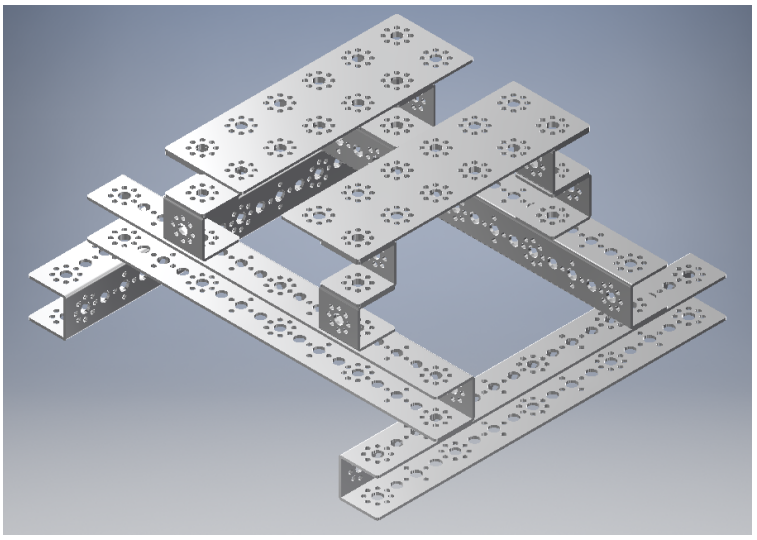


Figura 40: Conjunto completo de piezas para la base de la estructura

Luego de haber colocado las piezas anteriores, se añaden los tres soportes destinados al motor DC y los dos servomotores para el sistema de transmisión del sistema que se describirá más adelante en la sección 15. Se muestra su ubicación por medio de la figura 41.

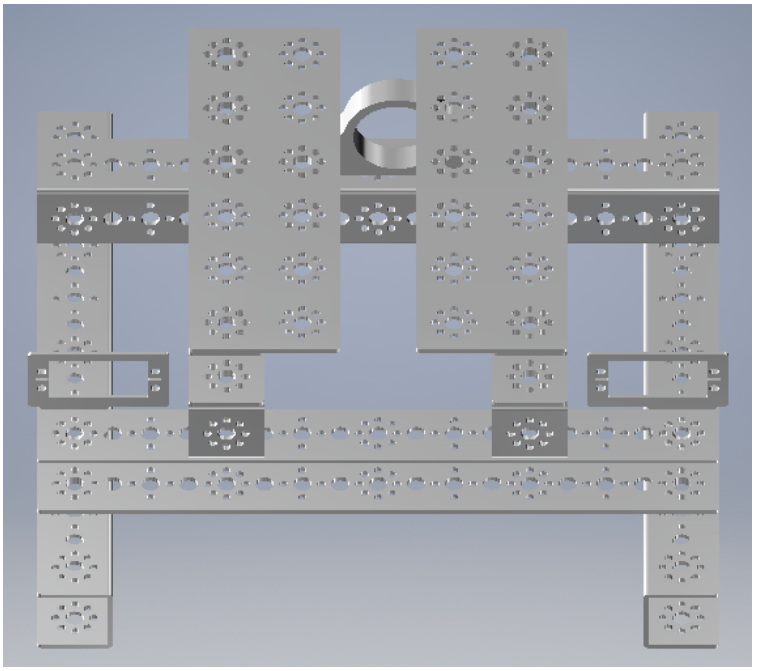


Figura 41: Disposición estructural de los soportes para los motores del sistema

Posterior a la disposición de las piezas, se asignan los correspondientes tornillos y tuercas de sujeción para proporcionar una mayor completitud al modelo 3D de la estructura del sistema trabajado. De esta manera, se obtiene la siguiente vista isométrica del modelo CAD completo para la base principal de la estructura mecánica (ver figura 42) a la cual luego se acoplará la transmisión de movimiento.

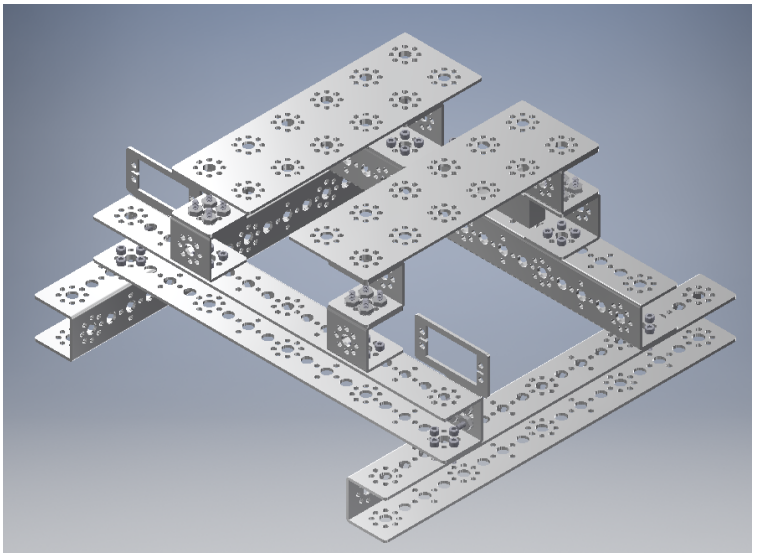


Figura 42: Base de la estructura mecánica con tornillos y tuercas posicionados

Cabe resaltar que no se han agregado tornillos en las placas superiores debido a que es necesaria la instalación de un elemento aislante que cubra el metal y así evite la interferencia

con el conexionado del sistema electrónico que se encontrará ubicado en su parte superior. A continuación, se observa el plano de la estructura mecánica (ver figura 43):

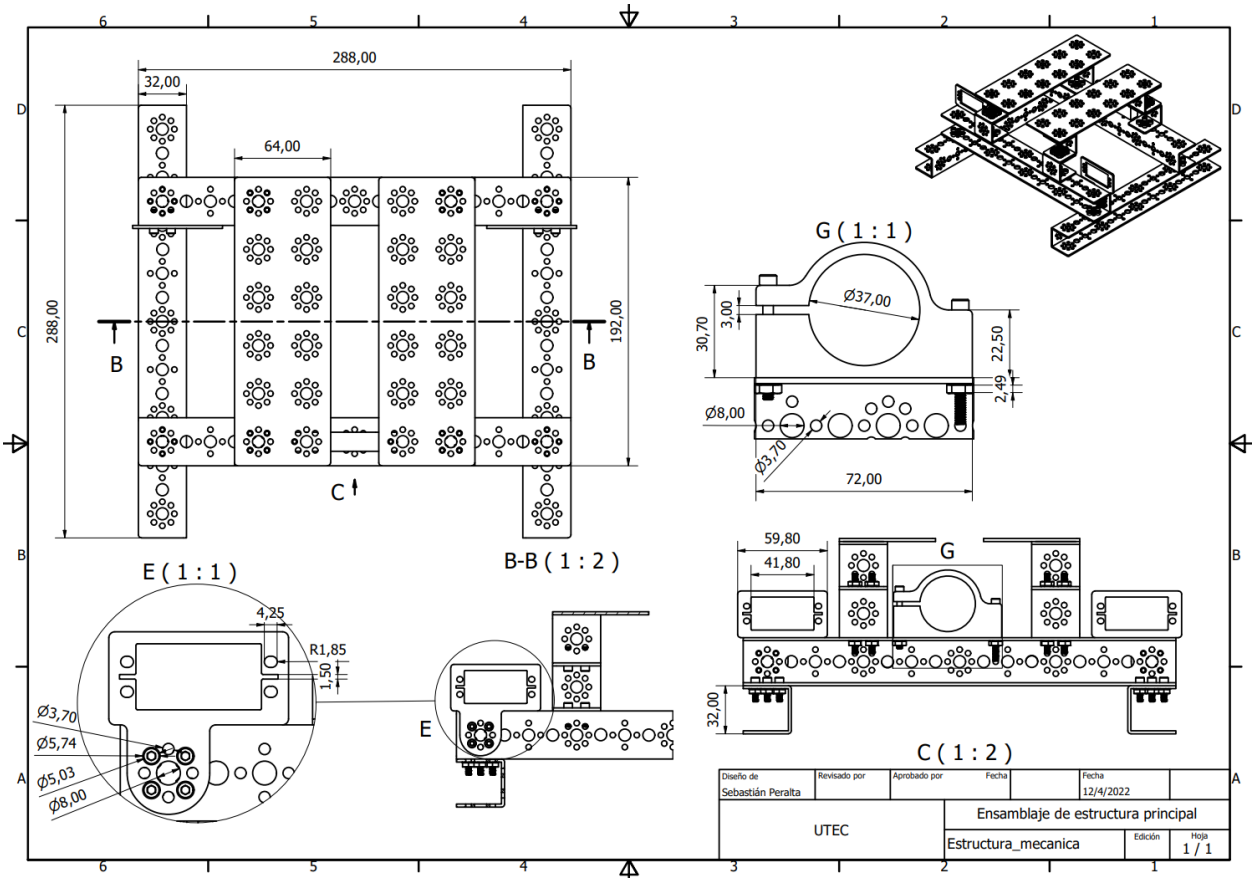


Figura 43: Plano de la estructura mecánica

En cuanto a la estructura que soporta la celda de carga, está conformada principalmente por dos capas de soportes que mantienen a la celda en una misma ubicación permitiendo su deformación de modo que las galgas extensiométricas puedan arrojar una lectura apropiada a manera de voltaje en el puente de Wheatstone. La primera capa de soporte la componen dos piezas planas de mayor longitud. Esta servirá para colocar los demás elementos requeridos para el soporte sobre ellas. Encima, se sitúa la segunda capa, la cual consta de 3 piezas planas de menos longitud colocadas ortogonalmente sobre las piezas de la primera capa. Sobre ambas piezas extremas de la segunda capa (ver figura 44).

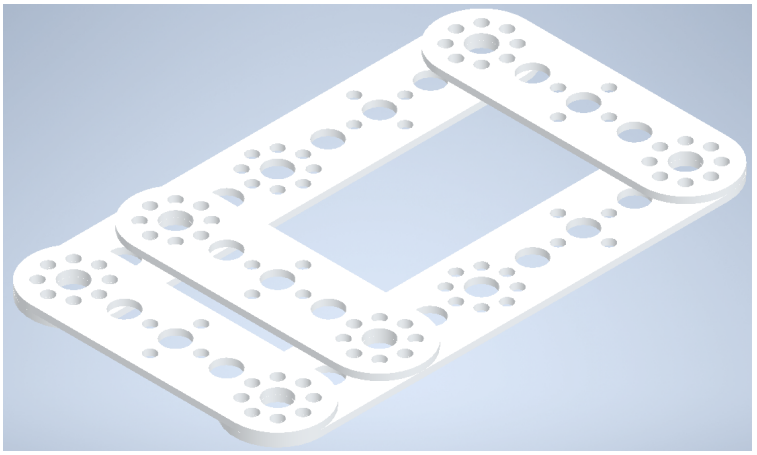


Figura 44: Capas iniciales del soporte para la celda de carga

Además, se incluyen dos piezas en forma de Ç.ªtornilladas a los laterales. Estas piezas cuentan con orificios en el centro, los cuales permitirán que una cuerda enlazada con el motor los sostenga y permita regular el nivel de la estructura.

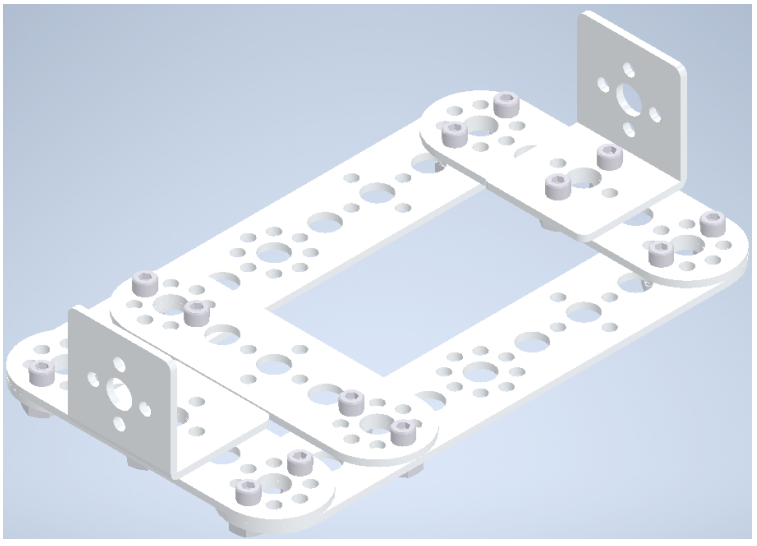


Figura 45: Soporte para la celda de carga con laterales para la suspensión del sistema

Finalmente, se implementó en la estructura el modelo CAD de la celda de carga de 10 kg. A continuación se muestra la estructura de soporte de la celda de carga con todos sus elementos:

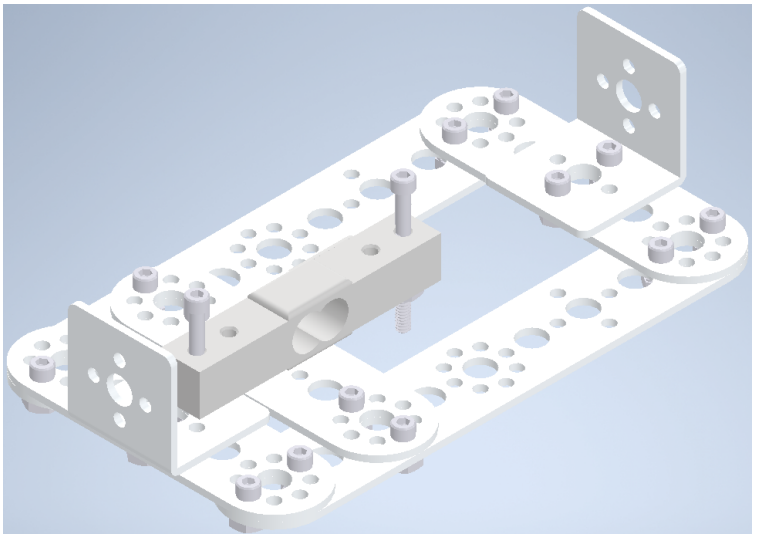


Figura 46: Vista isométrica de la estructura de soporte para la celda de carga

En la siguiente imagen se pueden observar los nudos utilizados para asegurar los tornillos acoplados a la estructura.

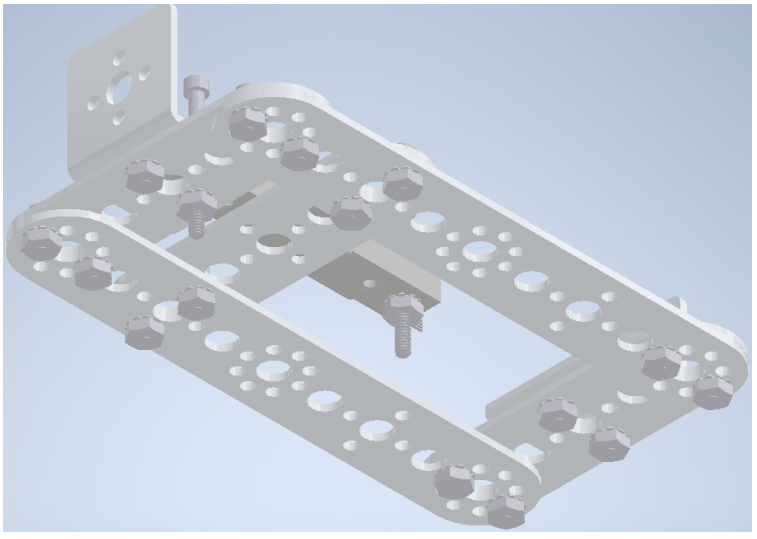


Figura 47: Vista de tornillos para la estructura de soporte para la celda de carga

Finalmente, se muestra el plano realizado a partir del modelo CAD de la estructura de soporte para la celda de carga.

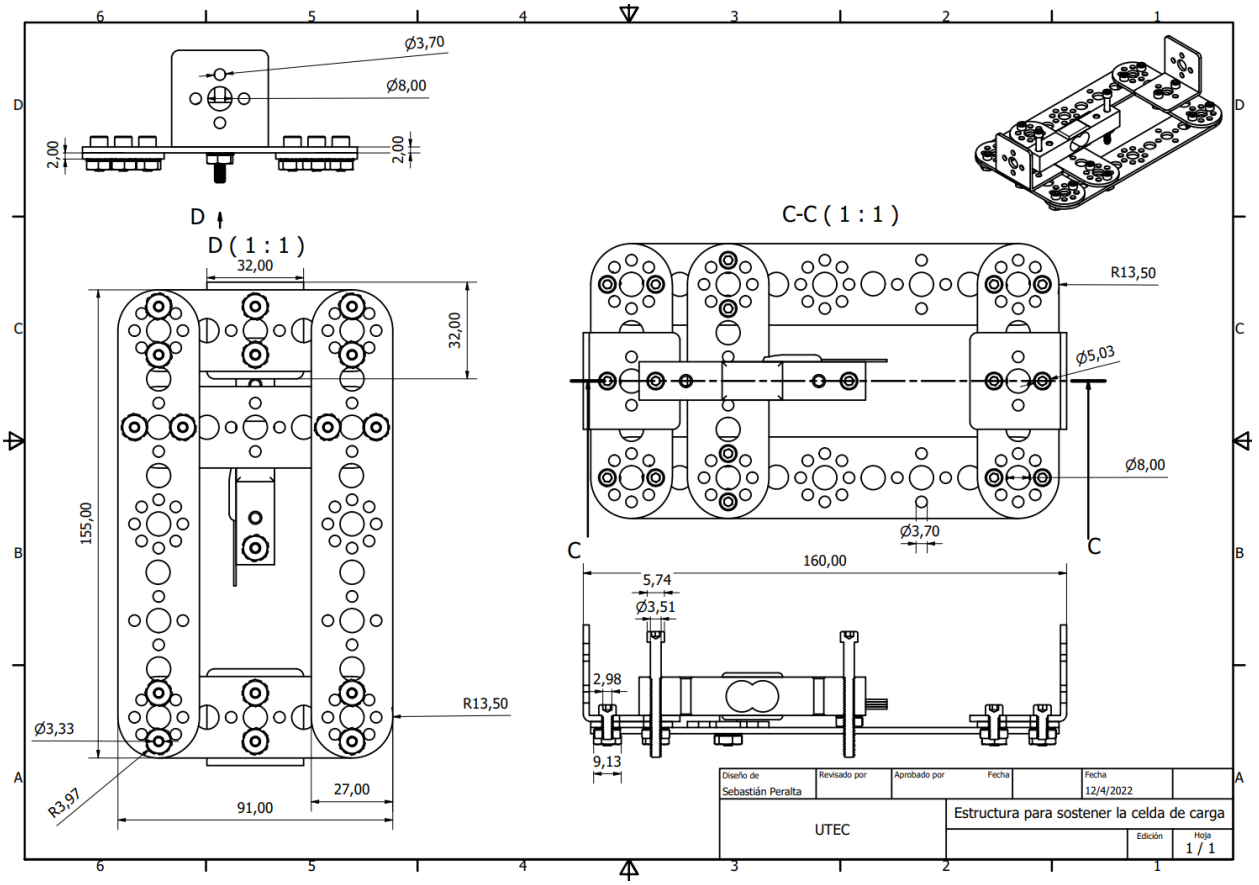
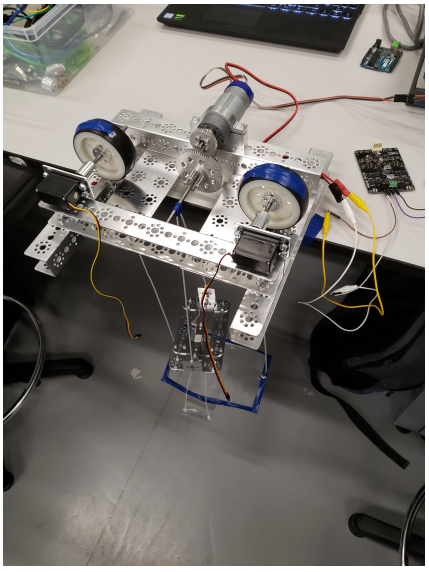
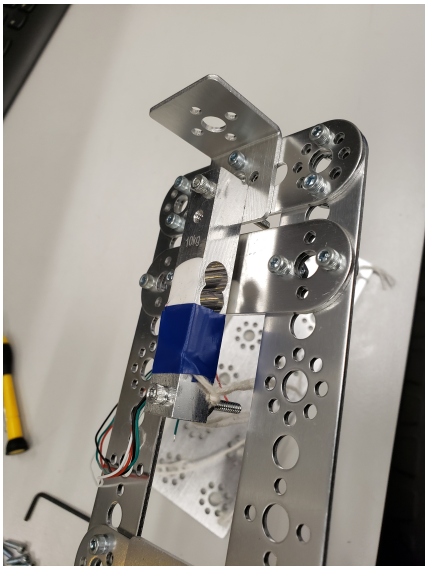


Figura 48: Plano de la estructura de soporte para la celda de carga

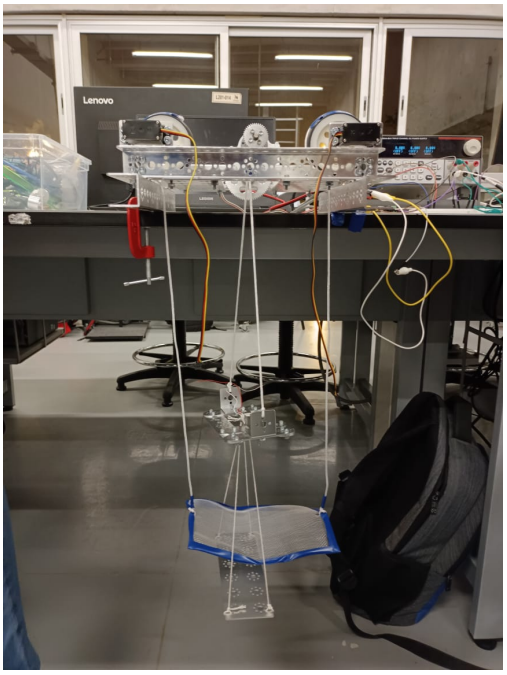
Los planos previos fueron establecidos en función de la versión final del diseño mecánico con el que fueron realizadas las pruebas del funcionamiento del prototipo. Sin embargo, también se realizó una elaboración física previa para poner a pruebas distintas configuraciones de las piezas con las que se podía trabajar. El resultado de esta construcción es el que se observa a continuación en la figura 49.



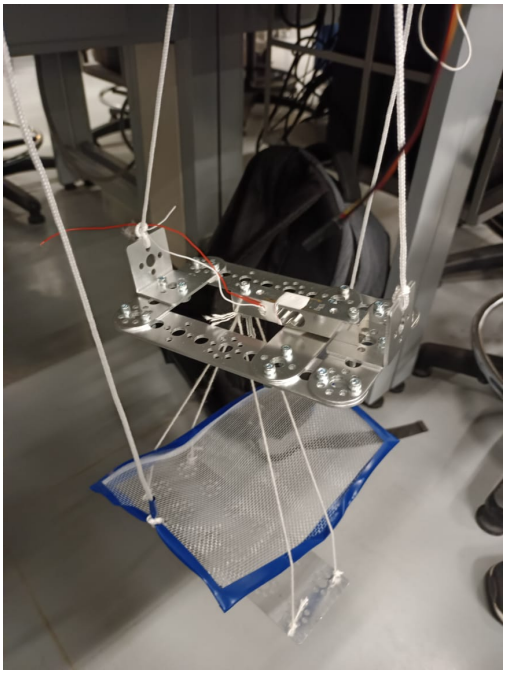
(a) Vista isométrica



(b) Sistema de medición de masa



(c) Vista frontal



(d) Sistema de remoción de camarones

Figura 49: Sistema de solución completo

Posteriormente, se agregaron las modificaciones que conllevaron la adición del segundo nivel ya mencionado previamente y con esto una tabla de madera a manera de elemento aislante para los componentes electrónicos en la parte superior. También, se sustituyó la placa metálica utilizada como simulación de recipiente por un borde plástico al que se adhirió una bolsa que se convirtió en el comedero del prototipo. Estas descripciones se ven concretadas en la figura 50.

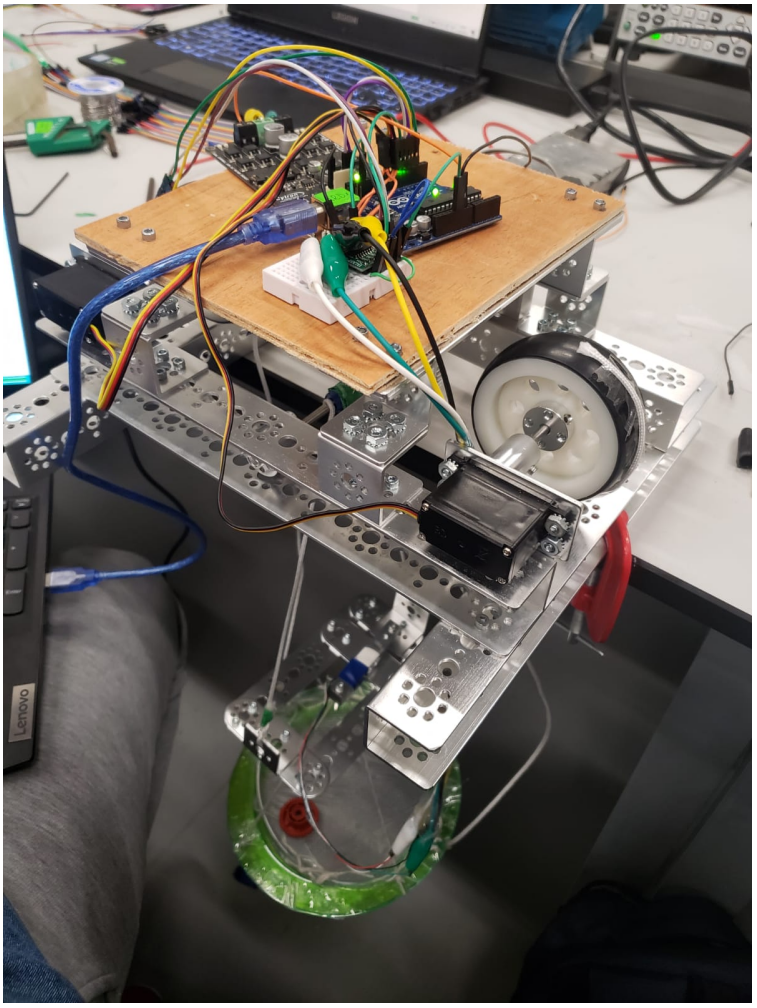


Figura 50: Construcción final del prototipo del sistema

15. Diseño y construcción del sistema de transmisión de movimiento

En torno a la transmisión de movimiento buscada para el sistema, se sabe con lo presentado por medio de la sección 14 que se cuenta con tres actuadores para componer la totalidad de desplazamientos necesarios para la operación del sistema. Los tres soportes que fueron agregados permiten que los actuadores sean acondicionados adecuadamente en la estructura mecánica como se muestra mediante la figura 51.

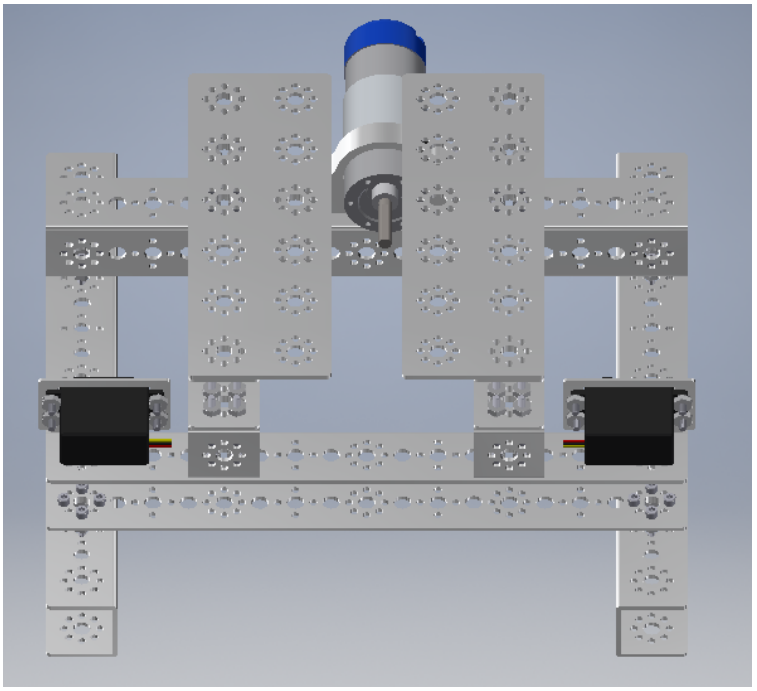


Figura 51: Modelo CAD con actuadores agregados

Es necesario que el motor desplace la carga a una velocidad que no desestabilice el sistema. Esto se debe a que un movimiento demasiado acelerado podría provocar daños en la estructura del prototipo. En ese sentido, no se requiere que el torque generado se aproveche sobre la carga, ya que el propósito del mecanismo es hacer que el desplazamiento sea lento para evitar cualquier posible perturbación debida a un movimiento rápido del comedero. Así, se establece la relación de engranajes que se muestra por medio de la figura 52.

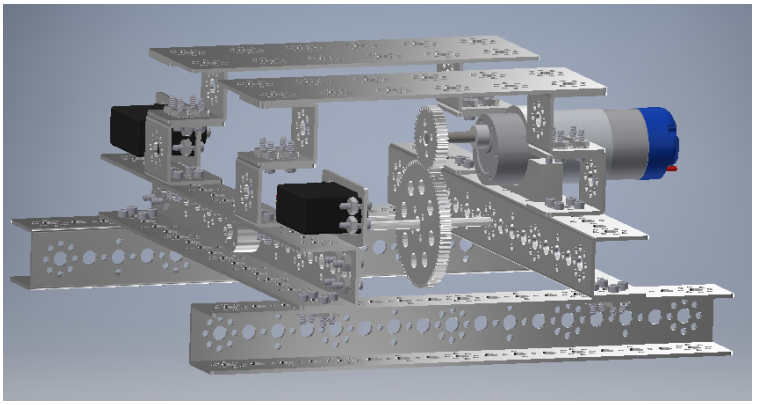


Figura 52: Modelo CAD que muestra la relación de engranajes a ser implementada en la transmisión del sistema

Con base en la primera construcción que fue elaborada para el prototipo de funcionamiento del sistema, se tiene una mejor visualización de la colocación tanto de los engranajes en sí mismos como de los topes que fueron colocados para restringir el desplazamiento axial del eje en el que se encuentran montados. Dicha construcción se evidencia por medio de la figura 53.

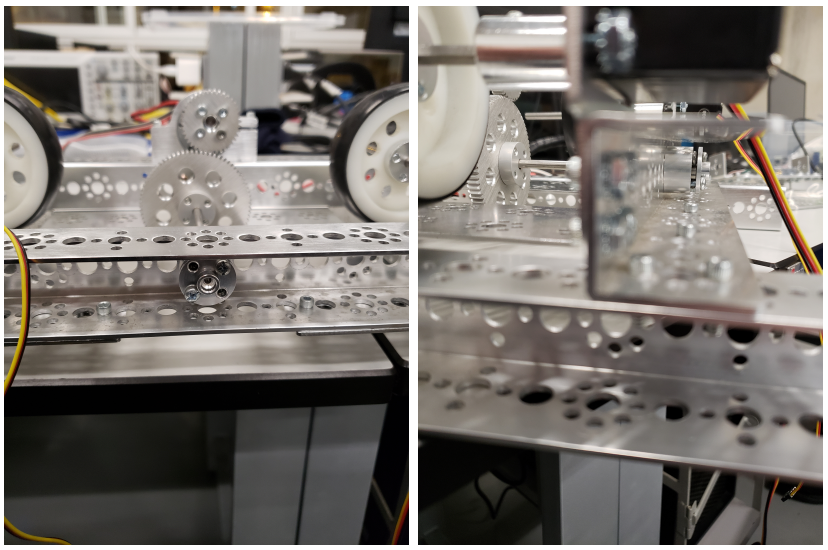


Figura 53: Visualización del sistema de transmisión para el motor DC

De manera semejante a como fue mostrado para la estructura mecánica general en la sección 14, la construcción inicial del sistema para la realización de pruebas del mecanismo es la que se ha mostrado ya, mientras que el resultado final es el que se muestra por medio de la figura 54.

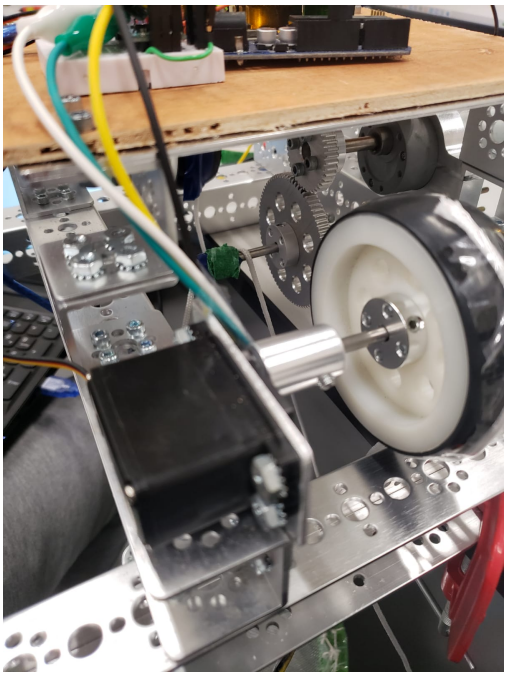


Figura 54: Construcción final de la transmisión para el motor DC en el prototipo

En lo que respecta a los servomotores del sistema, la consideración crítica en torno a ellos es su desplazamiento reducido respecto al motor DC. Por ello, requieren un incremento en su desplazamiento angular para que sea efectivo el movimiento de remoción de camarones en el sistema. Con tal finalidad, se han agregado ruedas del kit Tetrix para incrementar el

radio disponible y así conseguir el desplazamiento de agitación para remoción de camarones. El resultado del prototipado en las construcción inicial es lo que se aprecia en la figura 55.

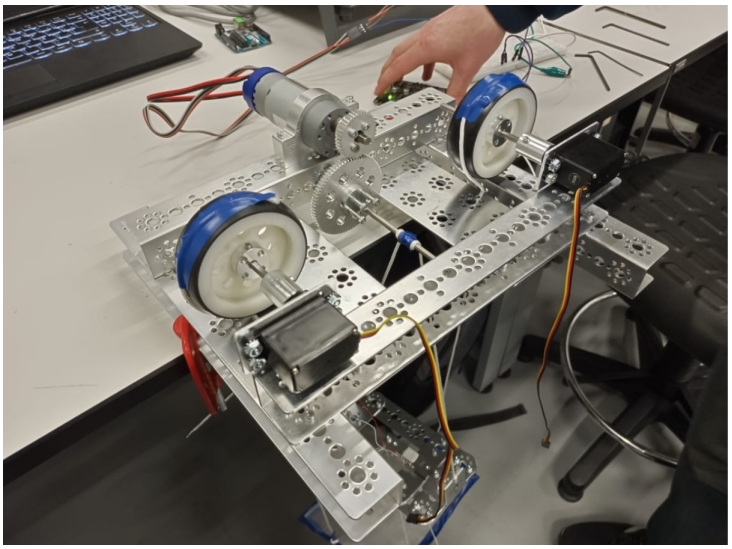


Figura 55: Adición de ruedas para incremento de desplazamiento angular de servomotores

16. Diseño y construcción del sistema electrónico

En el siguiente diagrama (ver figura 56) se puede observar las conexiones finales realizadas para el sistema electrónico.

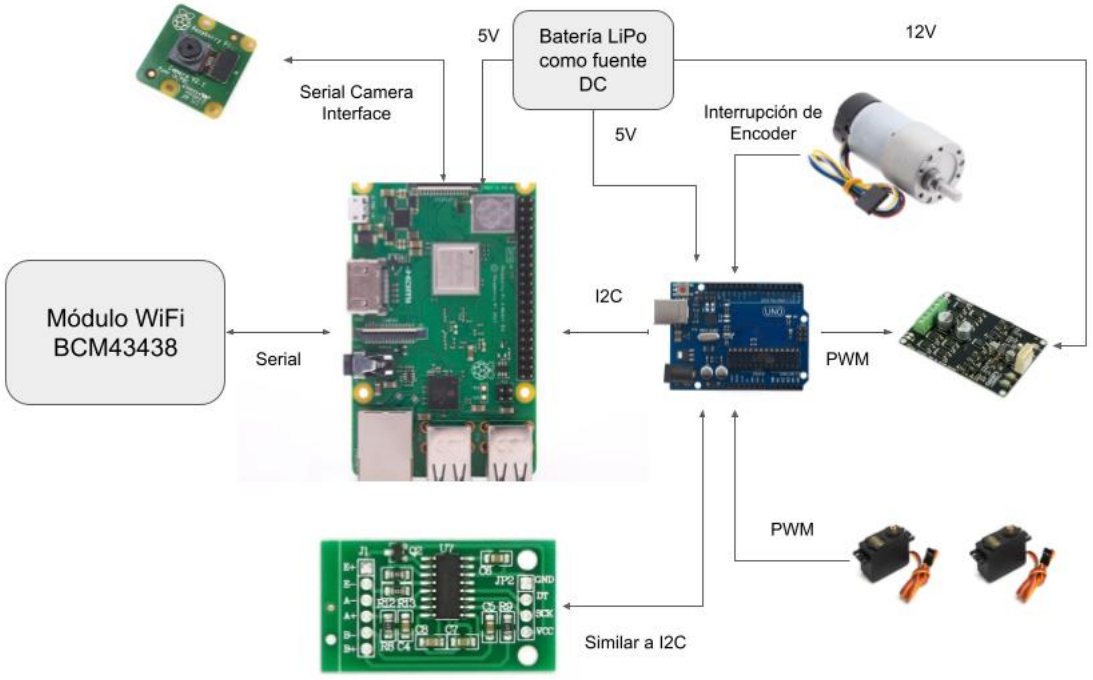


Figura 56: Módulos interconectados del prototipo implementado

A diferencia de la propuesta inicial, se utilizó la tarjeta Arduino UNO para el control del motor DC (movimiento de la bandeja de alimento) y del servomotor (movimiento de la malla para el sistema de remoción de camarones) así como para la medición de masa con el HX711 y la celda de carga. Por otro lado, el Raspberry Pi se encarga de procesar las imágenes recibidas por la cámara, recibir las mediciones del Arduino, enviar señales para realizar las mediciones periódicamente desde el Arduino, y mandar toda la información hacia la interfaz de usuario.

16.1. Sistema de medición de masa (HX711 + celda de carga)

El sistema de medición de masa se observa en la Figura 49b está compuesto por las siguientes partes:

- Arduino: recibe y procesa los datos provenientes del HX711. Se encuentra fuera de la Figura 49b, debido a que irá colocado en una plataforma al centro de la estructura principal.
- HX711: transmisor de celda de carga. Lee las señales de la celda de carga mediante un ADC y envía los datos al Arduino. Se encuentra fuera de la imagen por el mismo motivo que el Arduino.
- Celda de carga: transductor que contiene galgas extensiométricas configuradas en puente de Wheatstone que permiten convertir la energía mecánica; producida por cargas de compresión, tensión o flexión, a energía eléctrica medible. Como se explicó en la actividad 1, se encuentra atornillado a una estructura de 4 placas.
- Plato: representado por una placa cuadrangular, mediante cuerdas de nylon en sus 4 esquinas, se amarró al extremo de la celda de carga en tensión.

A continuación se muestran los códigos de Arduino comentados, utilizados para el sistema de medición:

```
1  //*****************************************
2  //************DEFINICIONES*****************
3  //*****************************************
4
5  #include <Arduino.h>
6  #define DOUT_MODE INPUT
7
8  #if FAST_CPU
9  uint8_t shiftInSlow(uint8_t dataPin, uint8_t clockPin, uint8_t bitOrder) {
10     uint8_t value = 0;
11     uint8_t i;
12
13     for(i = 0; i < 8; ++i) {
14         digitalWrite(clockPin, HIGH);
15         delayMicroseconds(1);
16         if(bitOrder == LSBFIRST)
17             value |= digitalRead(dataPin) << i;
18         else
19             value |= digitalRead(dataPin) << (7 - i);
```

```
20          digitalWrite(clockPin, LOW);
21          delayMicroseconds(1);
22      }
23      return value;
24  }
25  #define SHIFTIN_WITH_SPEED_SUPPORT(data,clock,order) shiftInSlow(data,clock,
        order)
26  #else
27  #define SHIFTIN_WITH_SPEED_SUPPORT(data,clock,order) shiftIn(data,clock,order)
28  #endif
29
30  //*****************************************
31  //************FUNCIONES**********************
32  //*****************************************
33
34  // Funcion para inicializar
35  void begin(byte dout, byte pd_sck);
36
37  // Funciones de timming
38  bool is_ready();
39  void wait_ready(unsigned long delay_ms = 0);
40
41  // Lectura y promedio de lectura
42  long read();
43  long read_average(byte times);
44
45  // Funciones getters
46  double get_value(byte times);
47  float get_units(byte times);
48
49  // Obtenci n de tara o error inicial
50  void tare(byte times);
```

Listing 1: Código archivo .h para la lectura del HX711

```
1  #include "DSM_HX711.h"
2  #include <Arduino.h>
3  #define DOUT 4
4  #define PD_SCK 5
5
6  byte GAIN = 1;
7  long OFFSET = 0;
8
9  void begin(byte dout, byte pd_sck){
10   // Inicializa el modulo HX711 utilizando el canal A con ganancia de 128
11
12   pinMode(PD_SCK, OUTPUT);
13   pinMode(DOUT, INPUT);
14 }
15
16
17 bool is_ready() {
18   // Espera un flanco de bajada de DOUT
19
20   return digitalRead(DOUT) == LOW;
```

```
21 }
22
23
24 void wait_ready(unsigned long delay_ms) {
25   // Espera el flanco de bajada luego de un delay de X ms
26
27   while (!is_ready()) {
28     delay(delay_ms);
29   }
30 }
31
32
33 long read(){
34
35     // Espera que la muestra este lista seg n el flanco de
36     // bajada de DOUT
37
38     wait_ready();
39
40     // Estructuras de datos:
41     // value : Almacena el valor final de lectura
42     // data  : Almacena el valor leido del ADC
43     // filler: Un conjunto de ceros para llenar el n mero
44     //         y tener un valor de 32 bits
45
46     unsigned long value = 0;
47     uint8_t data[3] = { 0 };
48     uint8_t filler = 0x00;
49
50     noInterrupts();
51
52     // Genera 24 pulsos en PD_SCK para leer la lectura
53     data[2] = SHIFTIN_WITH_SPEED_SUPPORT(DOUT, PD_SCK, MSBFIRST);
54     data[1] = SHIFTIN_WITH_SPEED_SUPPORT(DOUT, PD_SCK, MSBFIRST);
55     data[0] = SHIFTIN_WITH_SPEED_SUPPORT(DOUT, PD_SCK, MSBFIRST);
56
57     // Se define el canal y la ganancia utilizada para la sig. lectura
58     for (unsigned int i = 0; i < GAIN; i++) {
59       digitalWrite(PD_SCK, HIGH);
60       digitalWrite(PD_SCK, LOW);
61
62       interrupts();
63
64         // Es evaluada la data y se le concatena ceros para tener
65         // un registro de 32 bits
66       if (data[2] & 0x80) {
67         filler = 0xFF;
68       } else {
69         filler = 0x00;
70       }
71
72       // Se construye el valor entero de 32 bits final
73       value = ( static_cast<unsigned long>(filler) << 24
74           | static_cast<unsigned long>(data[2]) << 16
75           | static_cast<unsigned long>(data[1]) << 8
```

```
76            | static_cast<unsigned long>(data[0]) );
77
78        return static_cast<long>(value) - OFFSET;
79
80    }
81 }
82
83
84 long read_average(byte times) {
85   // Toma "times" muestras y las promedia
86   // para filtrar ruido
87
88   long sum = 0;
89   for (byte i = 0; i < times; i++) {
90     sum += read();
91     delay(0);
92   }
93   return sum / times;
94 }
95
96
97 double get_value(byte times) {
98   // Obtiene el valor promediado
99   return read_average(times);
100 }
101
102
103 float get_units(byte times) {
104   // Obtiene el valor promediado y convertido a kilos
105   // seg n un factor de escalamiento f sico considerando
106   // la resoluci n y las propiedades de la galga
107
108   return get_value(times) / 439430.25*2.23;
109 }
110
111
112 void tare(byte times) {
113   // Calcula la tara o error que debe sustraerse
114   // a la balanza sin carga
115
116   double sum = read_average(times);
117   OFFSET = sum;
118 }
```

Listing 2: Código archivo .h para la lectura del HX711

Principalmente, la función *get_units* convierte un valor promediado convertido a kilogramos. Este valor es obtenido en función a cierta cantidad solicitada (en el programa principal se definió este valor como 25) de muestras a promediar. Asimismo, la función *read*() permite enviar 24 pulsos por el puerto PD_SCK para poder obtener la lectura requerida en un valor entero de 32 bits.

Por otro lado, como se observó en experimentación previa con la celda de carga, existe un offset que debe ser compensado de manera que se pueda tener una lectura comprensible para cualquier usuario sin problema. Particularmente, para el caso de una balanza el pro-

cedimiento es conocido y se trata de la aplicación de una tara. Esto compensa el offset y establece un nuevo cero para la medición de peso que es registrado por el componente de manera que queda adaptado a la implementación realizada. Esto se corrigió con la función *tare*, que calcula el error a sustraerse cuando la balanza no cuenta con una carga.

```
#include "DSM_HX711.h"
#include <Arduino.h>
#define DOUT 4
#define PD_SCK 5

byte GAIN = 1;
long OFFSET = 0;

void begin(byte dout, byte pd_sck){
  // Inicializa el modulo HX711 utilizando el canal A con ganancia de 128

  pinMode(PD_SCK, OUTPUT);
  pinMode(DOUT, INPUT);
}


bool is_ready() {
  // Espera un flanco de bajada de DOUT

  return digitalRead(DOUT) == LOW;
}


void wait_ready(unsigned long delay_ms) {
  // Espera el flanco de bajada luego de un delay de X ms

  while (!is_ready()) {
    delay(delay_ms);
  }
}


long read(){

    // Espera que la muestra este lista seg n el flanco de
    // bajada de DOUT

    wait_ready();

    // Estructuras de datos:
    // value : Almacena el valor final de lectura
    // data  : Almacena el valor leido del ADC
    // filler: Un conjunto de ceros para llenar el numero
    //         y tener un valor de 32 bits

    unsigned long value = 0;
    uint8_t data[3] = { 0 };
    uint8_t filler = 0x00;

```

```
50      noInterrupts();
51
52      // Genera 24 pulsos en PD_SCK para leer la lectura
53      data[2] = SHIFTIN_WITH_SPEED_SUPPORT(DOUT, PD_SCK, MSBFIRST);
54      data[1] = SHIFTIN_WITH_SPEED_SUPPORT(DOUT, PD_SCK, MSBFIRST);
55      data[0] = SHIFTIN_WITH_SPEED_SUPPORT(DOUT, PD_SCK, MSBFIRST);
56
57      // Se define el canal y la ganancia utilizada para la sig. lectura
58      for (unsigned int i = 0; i < GAIN; i++) {
59        digitalWrite(PD_SCK, HIGH);
60        digitalWrite(PD_SCK, LOW);
61
62        interrupts();
63
64          // Es evaluada la data y se le concatena ceros para tener
65          // un registro de 32 bits
66        if (data[2] & 0x80) {
67          filler = 0xFF;
68        } else {
69          filler = 0x00;
70        }
71
72        // Se construye el valor entero de 32 bits final
73        value = ( static_cast<unsigned long>(filler) << 24
74            | static_cast<unsigned long>(data[2]) << 16
75            | static_cast<unsigned long>(data[1]) << 8
76            | static_cast<unsigned long>(data[0]) );
77
78        return static_cast<long>(value);
79
80    }
81  }
82
83
84  long read_average(byte times) {
85    // Toma "times" muestras y las promedia
86    // para filtrar ruido
87
88    long sum = 0;
89    for (byte i = 0; i < times; i++) {
90      sum += read();
91      delay(0);
92    }
93    return sum / times;
94  }
95
96
97  double get_value(byte times) {
98    // Obtiene el valor promediado
99    return read_average(times);
100 }
101
102
103 float get_units(byte times) {
104   // Obtiene el valor promediado y convertido a kilos
```

```
105   // segun un factor de escalamiento f sico considerando
106   // la resolucion y las propiedades de la galga
107
108   return get_value(times) / 439430.25;
109 }
110
111
112 void tare(byte times) {
113   // Calcula la tara o error que debe sustraerse
114   // a la balanza sin carga
115
116   double sum = read_average(times);
117   OFFSET = sum;
118 }
```

Listing 3: Código archivo .cpp para la lectura del HX711

```
1 #include "DSM_HX711.h"
2
3 #define DOUT 4
4 #define PD_SCK 5
5
6 void setup() {
7   Serial.begin(9600);
8   begin(4, 5);
9   Serial.print("Lectura del valor del ADC:  ");
10  Serial.println(read());
11  Serial.println("No ponga ningun  objeto sobre la balanza");
12  Serial.println("Destarando...");
13  Serial.println("...");
14  tare(20);  //El peso actual es considerado Tara.
15  Serial.println("Listo para pesar");
16 }
17
18 void loop() {
19  Serial.print("Peso: ");
20  Serial.print(get_units(25),3);
21  Serial.println(" kg");
22  delay(500);
23 }
```

Listing 4: Código Arduino para la lectura del HX711

16.2. Sistema de remoción de camarones

El sistema de medición de masa se observa en la Figura 49d está compuesto por las siguientes partes:

- Arduino: controla la rotación de los servomotores. Se encuentra fuera de la Figura 49d, debido a que irá colocado en una plataforma al centro de la estructura principal.
- Servomotores (2): son controlados por el Arduino y son los que permiten la inclinación de la malla al enredar la cuerda de nylon alrededor de sus ejes durante el giro. Son

utilizados en lugar de motores debido a que al cargar únicamente la malla con los camarones, no requiere de mucha potencia.

- Ruedas (2): están colocados en los ejes de los servomotores. Su función es aumentar el desplazamiento de la cuerda de nylon, al aumentar el diámetro de aquello sobre lo que se enreda. Al aumentar este desplazamiento, se logra una mayor inclinación de la malla.

- Malla: Se coloca sobre el plato y debe ser más ancha que este para evitar que los camarones caigan en el plato al inclinarse. Debe ser porosa para la comida, pero no para los camarones. De esta manera los camarones podrán alimentarse por encima de ella, pero cuando se incline removerá a los camarones.

A continuación se muestra el código de Arduino comentado para el sistema de remoción de camarones:

```
#include <Arduino.h>
#include <Servo.h> //SERVO
//#include "DSM_servomotor.h"
//*************************\\
////      DEFINICIONES      \\\\
//*************************\\
Servo servomotor_r; //servomotor derecho
Servo servomotor_l; //servomotor izquierdo
int i=0; //cuenta las veces que se inclina

void setup() {
  Serial.begin(9600);
  //Asignacion de pines a los
  //que se conectan los servos
  servomotor_r.attach(9);
  servomotor_l.attach(6);
}

void loop() {
  //Al iniciar el programa
  //Servomotores a posicion inicial
  servomotor_r.write(0);
  servomotor_l.write(0);
  delay(1000); //espera 1 segundo
  //Inclina 3 veces la malla hacia
  //la derecha (levanta la cuerda izquierda)
  if(i<3){
    servomotor_r.write(0); //servomotor derecho en 0
    servomotor_l.write(180); //servomotor izquierdo rota
    i=i+1; //Aumenta la cuenta
    delay(1000);
  }
  //Inclina 3 veces la malla hacia
  //la izquierda (levanta la cuerda derecha)
  else if(i<6 && i>=3){
    servomotor_r.write(180); //servomotor derecho rota
    servomotor_l.write(0); //servomotor izquierdo en 0
```

```
38      i=i+1; //Aumenta la cuenta
39      delay(1000);
40    }
41    //Luego de inclinar la malla 3 veces por lado
42    //la cuenta vuelve a 0
43    else{
44      i=0;
45      delay(5000);
46    }
47  }
```

Listing 5: Código Arduino para el movimiento de la malla

El código para el servomotor realiza 3 inclinaciones para cada lado y luego reinicia la cuenta hasta que se requiera el proceso nuevamente. Estas 3 inclinaciones se hacen para maximizar la probabilidad de que los camarones retenidos caigan fuera del sistema. Este procedimiento se puede observar en el siguiente video.

16.3. Control por retroalimentación PD de actuación mecánica

Como el motor DC que se encarga de elevar el comedero debe tener un manejo de su posición angular lo más precisa posible, se ha elegido por incluir un control PD que envíe una señal PWM al motor DC mediante un drive Cytron y que es retroalimentado por el encoder a la salida del motor. Este control es ejecutado dentro del Arduino UNO y es capaz de recibir una referencia de ángulo a través de la consola Serial mediante plotea su posición actual frente a la referencia para comprobar los requerimientos de control. Las lecturas del encoder son obtenidas mediante interrupciones y se calcula el error entre la referencia y las lecturas escaladas a ángulos hexadecimales. Este error es derivado por diferencias infinitas y la ley de control resultante es la combinación ponderada del error (proporcional) y su derivada. Es necesario aplicar una saturación +12 V sobre la ley para no dañar el actuador. Finalmente, se obtuvieron de manera experimental las ganancias Kp = 0.2 y Kd = 0.05.

Se podría haber realizado una sintonización más fina de estas ganancias utilizando los requerimientos de diseño cómo error en estado estacionario máximo, la rapidez de la respuesta y el sobre impulso aceptable. No obstante, para efectos prácticos de esta aplicación, las ganancias encontradas cumplen con los resultados que buscábamos. Adicionalmente, se menciona que el uso de una ganancia integral inducía cierta inestabilidad en el sistema producto de las oscilaciones que genera a pesar del uso de un ganancia derivativa y la carga inercial que amortigua este efecto. Cómo se observó que el error en estado estacionario era mínimo, se prefirió prescindir de este término en el control de posición.

```
1 #define MOTOR_FREQUENCY                 1000    //hz
2
3 #define ENC_IN_RIGHT_A 2
4 #define ENC_IN_RIGHT_B 3
5
6 #define PWM1 5
7 #define DIR1 4
8
9 volatile int right_wheel_tick_count = 0;
```

```
volatile int last_right_wheel_tick_count = 0;

int val = 0;
int encoder_val =0;

float kp = 0.2;
float ki = 0.0;
float kd = 0.05;
float Theta, Theta_d;

int dt;

unsigned long t;
unsigned long t_prev = 0;

int val_prev =0;
float e, e_prev = 0, inte, inte_prev = 0;
float Vmax = 12;
float Vmin = -12;
float V = 0.1;

char c;
```

Listing 6: Código archivo .h para el control del motor DC

```
#include "ctrl_motor.h"

//***Motor Driver Functions*****

void WriteDriverVoltage(float V, float Vmax) {
  int PWMval = int(255 * abs(V) / Vmax);
  if (PWMval > 255) {
    PWMval = 255;
  }
  if (V > 0) {
    digitalWrite(DIR1, HIGH);
  }
  else if (V < 0) {
    digitalWrite(DIR1, LOW);
    }
  else {
    digitalWrite(DIR1, LOW);
  }
  analogWrite(PWM1, PWMval);

}


void setup() {
  Serial.begin(9600);
  pinMode(ENC_IN_RIGHT_A , INPUT_PULLUP);
  pinMode(ENC_IN_RIGHT_B , INPUT);

  digitalWrite(PWM1, LOW);        // MOTOR
  digitalWrite(DIR1, LOW);
```

```
31
32    attachInterrupt(digitalPinToInterrupt(ENC_IN_RIGHT_A), right_wheel_tick
        , RISING);
33  }
34
35
36  void loop() {
37
38    // Reading values
39    //val = analogRead(potPin);                              // Read V_out
        from Reference Pot
40    encoder_val =right_wheel_tick_count;
41    while(Serial.available()){
42      c = Serial.read();
43      if (c == 'A'){
44        val = Serial.parseInt();
45        }
46      }
47
48
49
50    // Calculo dt y referencias
51    t = millis();
52    dt = (t - t_prev);                                       // Time step
53    Theta = val;                                             // Theta= Actual
        Angular Position of the Motor
54    Theta_d = encoder_val;                                   // Theta_d= Desired
        Angular Position of the Motor
55
56    // Calculo de error y ley de control
57    e = Theta_d - Theta;                                     // Error
58    inte = inte_prev + (dt * (e + e_prev) / 2);              // Integration of
        Error
59    V = kp * e + ki * inte + (kd * (e - e_prev) / dt) ;      // Controlling
        Function
60
61    // Saturaci n
62    if (V > Vmax) {
63        V = Vmax;
64        inte = inte_prev;
65      }
66    if (V < Vmin) {
67      V = Vmin;
68      inte = inte_prev;
69      val_prev= val;
70    }
71
72    // Envio de ley de control
73    WriteDriverVoltage(V, Vmax);
74    Serial.println(Theta_d); Serial.print(" \t");
75    Serial.print(Theta); Serial.print(" \t ");
76
77    // Actualizaci n
78    t_prev = t;
79    inte_prev = inte;
```

```
80    e_prev = e;
81    delay(10);
82
83  }
84
85  void right_wheel_tick() {
86
87    // Read the value for the encoder for the right wheel
88    int val = digitalRead(ENC_IN_RIGHT_B);
89
90    if(val == LOW) {
91      right_wheel_tick_count++;
92    }
93    else {
94      right_wheel_tick_count--;
95    }
96
97  }
```

Listing 7: Código Arduino para el control del motor DC

Las evidencias de este sistema se encuentran en el siguiente video. En las evidencias se observa el excelente desempeño que tiene el control actual, y esa es la razón por la cual no se ha incluido un terminal integral, en especial, porque inducen ruido sobre la señal de control y sobre el actuador mismo. Afortunadamente, cualquier ruido del actuador o inducido por una señal del encoder contaminado es atenuado en cierta manera por la carga inercial que tiene el motor DC acompañado por la etapa de reducción.

16.4. Procesamiento de imágenes digitales

Para obtener una mayor información sobre la cantidad de alimento consumido e inclusive combinar las distintas mediciones físicas del alimento, se ha diseñado un subsistema de visión computacional que se encarga de estimar el área cubierta por el alimento en el comedero. La metodología es aplicar segmentación de color en el espacio de color HSV (Hue, Saturation Value), ya que es más fácil identificar colores y ajustar solo un parámetro (Saturation) para obtener todo el espectro cercano a ese color cómo sus versiones oscuras o más claras. El resultado de este proceso es una máscara, una imagen que contiene un objeto de nuestro interés de color blanco y el resto de la imagen de color negro, de la cual se puede extraer su contorno y calcular fácilmente su área. Para mejorar la estimación del área se aplicaron dos operaciones de procesamiento a la máscara para eliminar ruido, las cuales son erosión y dilatación.

El equipo utilizado es una tarjeta Raspberry Pi 3 con su respectiva cámara Pi que puede obtener hasta resoluciones de 1080p a 30 FPS. Así mismo, se utilizó la librería OpenCV para realizar las operaciones de cambio de espacio de color RGB a HSV, la segmentación de color, la obtención de la máscara, el cálculo de contornos y la obtención de su área. El script utilizado se presenta a continuación:

```
1 import cv2 as cv
2 import numpy as np
```

```
from picamera.array import PiRGBArray
from picamera import PiCamera

window_capture_name = 'Video Capture'
window_detection_name = 'Object Detection'

kernel_open = np.ones((5, 5), np.uint8)
kernel_close = np.ones((20,20), np.uint8)

camera = PiCamera()
camera.resolution = (640, 480)
camera.framerate = 30

rawCapture = PiRGBArray(camera, size=(640, 480))

for frame in camera.capture_continuous(rawCapture, format="bgr",
    use_video_port=True):

    frame = frame.array
    frame_HSV = cv.cvtColor(frame, cv.COLOR_BGR2HSV)
    img_mask = cv.inRange(frame_HSV, (low_H, low_S, low_V), (high_H, high_S,
    high_V))

    img_mask = cv.morphologyEx(img_mask, cv.MORPH_OPEN, kernel_open)
    img_mask = cv.morphologyEx(img_mask, cv.MORPH_CLOSE, kernel_close)

    img_final = img_mask

    const, h = cv.findContours(img_final.copy(),cv.RETR_EXTERNAL,cv.
    CHAIN_APPROX_NONE)

    total_area = 0

    for i in const:
        if cv.contourArea(i) < 100:
            continue
        else:
            total_area += cv.contourArea(i)

    cv.drawContours(frame,const,-1,(0,255,0))
    cv.putText(frame, str(total_area), (100,100), cv.FONT_HERSHEY_SIMPLEX
    ,0.65, (255, 255, 255), 2)
    cv.imshow(window_capture_name, frame)
    cv.imshow(window_detection_name, img_mask)

    key = cv.waitKey(30)
    if key == ord('q') or key == 27:
        break
```

Listing 8: comida_segmentation.py

17. Calibración del sistema de medida y muestreo de datos experimentales

17.1. Medición de masa

Para poder hallar el factor de conversión del valor leído a la medida adecuada, se utilizaron 4 masas distintas (con valores conocidos y comprobados con una balanza). De esta manera, los datos obtenidos fueron los siguientes:

Masa conocida (kg)	Lectura inicial	Lectura con la masa	Factor de conversión
0.5881	0.404	0.673	2.186
0.2081	0.404	0.493	2.3382
0.067	0.405	0.435	2.233
0.3414	0.405	0.550	2.3545

Cuadro 2: Mediciones iniciales del sistema de medición de masa

Con ello, el valor promedio de los factores de conversión es 2.2779; sin embargo, se utilizó 2.23 en el sistema de medición después de realizar pruebas adicionales con lo que tenemos un valor más cercano. Las lecturas calibradas obtenidas se muestran a continuación:

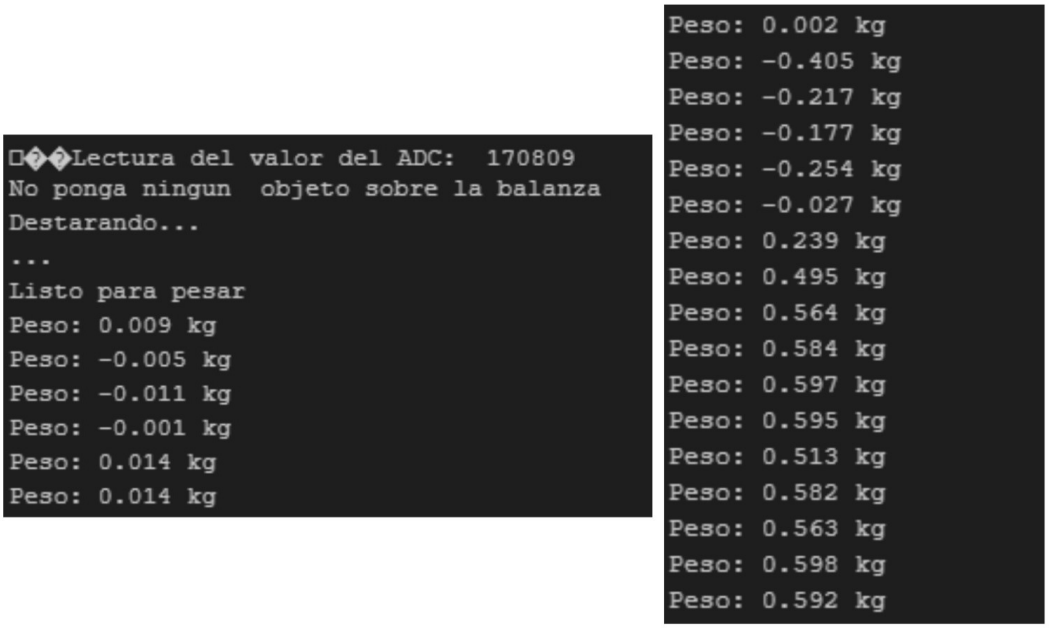


Figura 57: Mediciones de la celda de carga registradas

En la figura anterior (Figura 57) se observa que el valor inicial tras destarar es aproximadamente 0. Esto significa que el offset, el cual es equivalente a la masa del plato aproximadamente, se ha calculado muy cerca del valor real. Asimismo, se realizó la medición utilizando la masa conocida de 0.5881 kg, dando como resultado valores entre 0.584 kg y 0.598 kg. Esto evidencia la exactitud del factor de conversión tomado.

17.2. Segmentación del color en imágenes

Durante las pruebas se identificó que el color de segmentación de la comida se encuentra dentro de los rangos de HSV(7,18,118) y HSV(42,70,150). Estos valores pueden variar dependiendo de las condiciones de luminosidad de la escena. Así mismo, se pudo observar que la aplicación de la erosión y dilatación eliminan gran parte del ruido de la máscara original, aunque también aumentan el uso de recursos computacionales. No obstante, si solo se busca enviar imágenes con el área ya estimada y no realizar esta tarea en un vídeo enviado en tiempo real, la tarjeta Raspberry no presenta muchas complicaciones para realizarla.

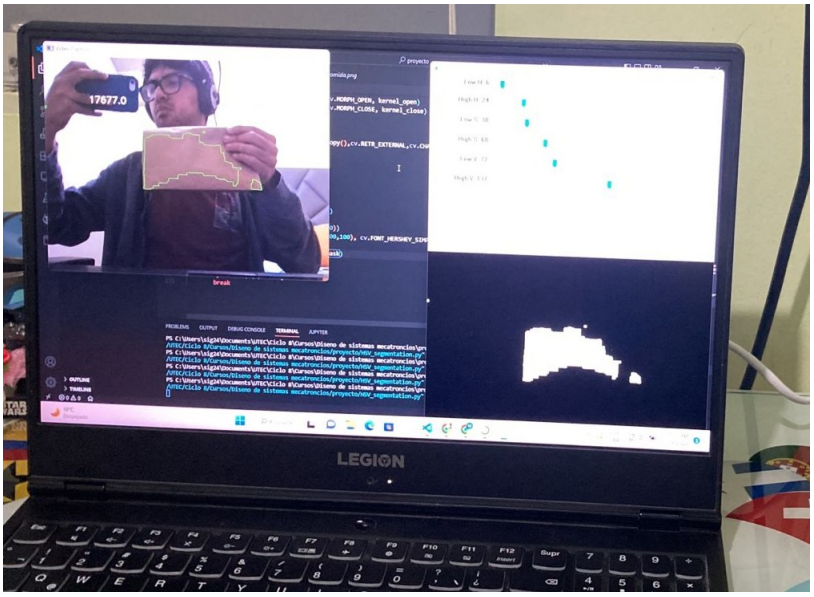


Figura 58: Registro de las imágenes con la cámara

El objetivo final sería tener una gráfica con marcas de tiempo que compara la cantidad de peso medido por la galga y la estimación de área de la cámara para conocer el consumo de los camarones que pueda ser visualizada en la interfaz de usuario por cada estación. Precisamente esto nos daría una mejor aproximación, ya que la referencia del comportamiento respecto a la alimentación sería de dos tipos (imagen y valor registrado por la galga). Así, al combinar la información tanto del peso del alimento como de la imagen del alimento restante en el comedero correspondiente nos brinda una adecuada coordinación de los resultados obtenidos.

La información del área estimada fue comprobada de forma experimental con un montículo formado por lentejas que tiene un color similar al de la comida de las camarones. Cómo la estimación del color HSV que se deseaba segmentar ya fue realizada, el nuevo resultado experimental prescinde de la ventanas de calibración HSV y el resultado de la máscara para solo mostrar el área estimada en valor numérico. Esta medición es enviada desde el Raspberry hasta un terminal laptop por conexión de SSH donde por esta misma conexión se podría desplegar la interfaz gráfica cargada dentro del Raspberry. Los resultados de este apartado se pueden observar en el siguiente vídeo

De las pruebas experimentales se pueden realizar varios comentarios pertinentes:

- La estimación de área de comida están en unidades de píxeles detectados, por lo que sería necesario tener un factor de conversión de píxeles a cm^2. En vista que la distancia

a la comida afecta la cantidad de área estimada, cómo se observa en el video al acercar la cámara, es necesario tener una referencia visual que esté estática con respecto al comedero y que se conozca con certeza su área en cm^2. Esto podría realizar con un rectángulo de color amarillo y dimensiones conocidas que es fácil de distinguir y segmentar.

- La estimación de área contiene aún muestras ruidosas a pesar del preprocesamiento realizado por lo que se podría suavizar la estimación tomando 5 a 10 fotos del comedero en su posición superior. y promediando el área detectada en cada una de ellas.

18. Resultados de validación experimental

18.1. Muestreo de datos experimentales

En esta subsección, se muestran los resultados experimentales de la masa de la comida antes y después de humedecerla. Para realizar esta prueba, se instaló el sistema de alimentación sobre un recipiente con agua. Se hizo de tal manera que al descender al nivel más bajo del plato este pudiese sumergirse (ver figura 59).

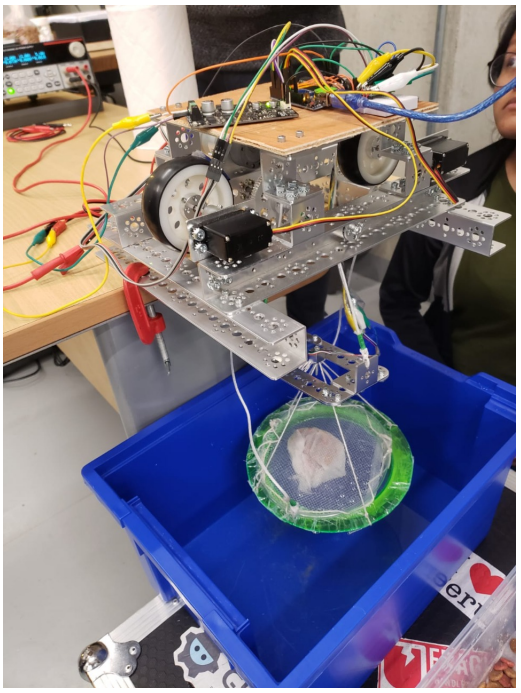


Figura 59: Simulación de funcionamiento del sistema para el muestreo de datos experimentales

A partir de los datos obtenidos se puede determinar que tan confiable es nuestro sistema es entre las dos etapas, utilizando un caso lo más cercano a la realidad y aplicar correcciones sobre el valor medido.

En la tabla 3 se presentan todos los valores números empleados en el experimento que se conforman por el valor real de la masa de comida previamente medido en una balanza de

precisión, el valor medido por la galga antes de mojarse y después de mojarse. Cabe resaltar que puede existir cierta desviación entre la medición seca y el valor real producto que la comida fue envuelta en una hoja de papel absorbente.

N° de muestra	Masa real (kg)	Masa seca (kg)	Masa mojada (kg)
Muestra 1	0.013	0.015	0.050
Muestra 2	0.016	0.017	0.047
Muestra 3	0.0204	0.022	0.048
Muestra 4	0.0294	0.033	0.057
Muestra 5	0.0342	0.038	0.067

Cuadro 3: Resultados experimentales de estimación de masa

Para comprobar si el factor de conversión muestras a kilogramos de la galga fue seleccionado correctamente en las secciones anteriores, se realizará una regresión lineal entre el valor real y el valor seco de masa donde una pendiente igual a 1 y *bias* igual a 0 indica que la medición es exacta.

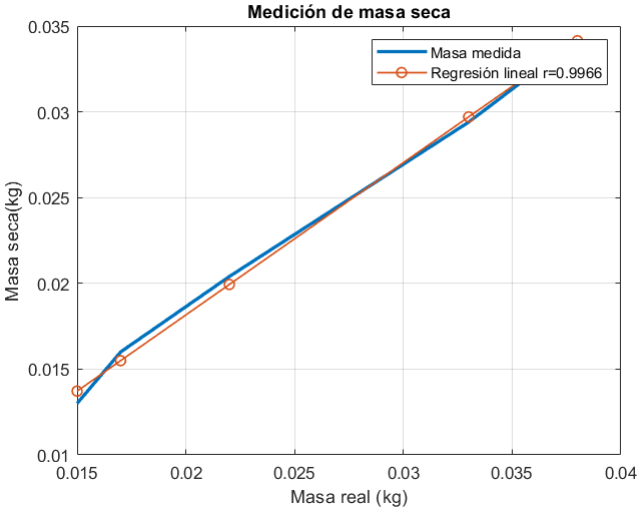


Figura 60: Resultados para masa seca

En primer lugar, a partir de 60, que las afirmaciones que se hagan a partir de la regresión lineal obtenida son confiables por el alto residuo que se consiguió, donde el valor ideal es 1. Así mismo, no hay evidencia alguna de sobre ajuste entre la regresión y la data real. En segundo punto, se obtuvo la ecuación lineal de $y = 0{,}8882x + 0{,}00039$ que indica la cercanía que tienen los datos reales frente a los medidos. Por lo tanto, podemos concluir que tanto nuestro factor de conversión cómo el proceso de tarado influyen correctamente en la medición final.

Por otra parte, se ha repetido el proceso anterior para la data de masa mojada para explorar que posibles correcciones se pueden aplicar a la medición final.

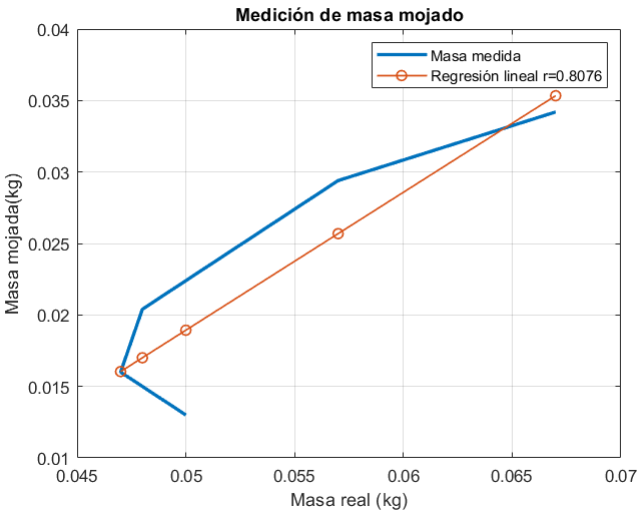


Figura 61: Resultados para masa mojada

En 61 se muestra que el residuo de la nueva regresión no es tan alto a comparación del caso para masa seca, aunque todavía se mantiene un rango aceptable. La ecuación obtenida para masa mojada fue $y = 0{,}9648x - 0{,}0293$ de la cual puede concluirse que si bien existe una fuerte correlación entre las dos variables, hay presencia de un *bias* para cada medición de masa mojada igual a -0.0293. El origen de esta desviación puede deberse a la absorción de agua en el comedero y en el papel que recubría la comida. Cómo solución a estos problemas de implementación, se podría aplicar la regresión lineal sobre la medición de masa mojada a manera de corrección para obtener la masa real de la comida con la condición que el tiempo de sumergimiento y secado de la comida sea uniforme.

19. Conclusiones y Recomendaciones

19.1. Conclusiones

- El sistema propuesto se compone tanto de una sección mecánica como electrónica. Necesita una alimentación que constituye un sistema de potencia y para establecer comunicaciones tiene también un subsistema específico que puede considerar todos los protocolos involucrados. Dado que cada uno de ellos cuenta con tareas particulares, resulta útil distribuir la carga de planificación entre los miembros del equipo para que la ejecución de cada etapa resulte apropiada y más eficiente.

- Los diagramas de bloques permiten estar al tanto de la interconexión que debe estar presente entre los componentes de cada uno de los subsistemas que conforman la implementación a llevar a cabo. El diseño se compone así de distintas secciones en las que se debe cubrir un conjunto de requerimientos específicos para su correcta operación. Asimismo, la organización visual de los componentes permite la realización de una revisión efectiva para la identificación de fallas en torno a la instalación de dispositivos o a la configuración general de cada subsistema. De manera semejante, se pueden proponer modificaciones al diseño de manera clara y los cambios se vuelven más comprensibles,

incluso sin tener el total entendimiento de cómo funcionan todos los componentes del sistema.

- Con base en la visualización física del sensor de carga, se ha establecido un modelo en tres dimensiones que representa la configuración ya descrita en torno a los principios de transducción involucrados. La forma más clara de llevar a cabo esta transmisión de información respecto al diseño consiste en plasmar un plano con las principales dimensiones con las que se está trabajando. De este modo, se puede presentar el modelo planteado y este podría ser replicado. Se trata de una manera de incrementar la repetibilidad del diseño en algún entorno, ya sea del mismo tipo o semejante, en donde pueda ser aplicada una implementación de la misma naturaleza.
- El software de Autodesk para el trabajo de PCBs, Eagle, resulta bastante versátil para la realización de un modelado claro de las líneas de conexión en un circuito impreso. Se ha replicado el módulo HX711 siguiendo las restricciones pertinentes y cumpliendo con la misma distribución de componentes de acuerdo con lo que va a ser implementado en el sistema de alimentación de camarones. Además, se pudo extraer de este mismo entorno un modelo 3D basándose en una exportación del circuito elaborado a Fusion 360. De manera semejante a los planos, las hojas técnicas son el medio para evidenciar el modo en el que se encuentran interconectados los componentes electrónicos disponibles en un circuito integrado de esta naturaleza y permiten comunicar la forma en la que deberían ser instalados.

19.2. Recomendaciones y trabajos futuros

- Sistema de medición de peso: Se debe comprobar que la corrección final aplicado al peso mojado mediante la regresión lineal obtenida permite obtener una mejor estimación.
- Integración de una interfaz gráfica por Node-RED: Esta plataforma será ejecutada dentro del Raspberry Pi 3 y puede ser visualizada desde un terminal laptop mediante conexión SSH. Desde esta interfaz con su respectivo Back end se pueden enviar señales periódicas para tomar fotos, para hacer las mediciones, graficar el historial de mediciones, interactuar con varias estaciones, enviar comandos por serial hacia el Arduino para los motores, etc. Este trabajo futuro sería clave para el despliegue del proyecto funcional y requiere tiempo para implementar cada componente de Node-RED con su respectiva verificación de funcionamiento entrelazado.
- Segmentación de color: Cómo se mencionó en apartados anteriores, existe un ruido considerable en la estimación de área de la comida que podría reducirse filtrado 5 a 10 valores de área medidos. Así mismo, se recomienda fijar un límite máximo de área en el comedero para descartar valores anormales.
- Mayor procesamiento de imagen: Se podría realizar un procesamiento de la información visual de forma más exhaustiva dentro de la Laptop para poder detectar los camarones mediante redes convolucionales cómo es el caso de la arquitectura YOLO. Esto podría ser ayuda cuando el movimiento de los servomotores no haya logrado retirar los camarones por completo y también rechazar la zona detectada cómo camarón en la segmentación de color.

- Estimación de peso a partir de imágenes: Si bien no existe muchas referencias de cómo poder convertir la estimación de área en peso restante de la comida, en [13] se consiguió esta tarea al utilizar redes neuronales de pocas capas de neuronas debido a que funcionan de forma óptima cómo aproximadoras de funciones. Cómo entradas a la red tenía el ancho y el largo de una fruta y en su salida la estimación de peso. Esto podría ser extrapolado para el área segmentada de la comida. Con dos medidas diferentes de peso, se tendría un valor mucho más confiable.

Referencias

[1] S. Peixoto, R. Soares, J. F. Silva, S. Hamilton, A. Morey, and D. A. Davis, "Acoustic activity of litopenaeus vannamei fed pelleted and extruded diets," *Aquaculture*, vol. 525, p. 735307, 2020.

[2] J. Reis, S. Peixoto, R. Soares, M. Rhodes, C. Ching, and D. A. Davis, "Passive acoustic monitoring as a tool to assess feed response and growth of shrimp in ponds and research systems," *Aquaculture*, vol. 546, p. 737326, 2022.

[3] T. Napaumpaiporn, N. Chuchird, and W. Taparhudee, "Study on the efficiency of three different feeding techniques in the culture of pacific white shrimp (litopenaeus vannamet)," *Journal of Fisheries and Environment*, vol. 37, no. 2, pp. 8–16, 2013.

[4] J.-B. Darodes de Tailly, J. Keitel, M. A. Owen, J. M. Alcaraz-Calero, M. E. Alexander, and K. A. Sloman, "Monitoring methods of feeding behaviour to answer key questions in penaeid shrimp feeding," *Reviews in Aquaculture*, vol. 13, no. 4, pp. 1828–1843, 2021.

[5] M. Wei, Y. Lin, K. Chen, W. Su, and E. Cheng, "Study on feeding activity of litopenaeus vannamei based on passive acoustic detection," *IEEE Access*, vol. 8, pp. 156654–156662, 2020.

[6] N. Chirdchoo and W. Cheunta, "Detection of shrimp feed with computer vision," *Interdisciplinary Research Review*, vol. 14, no. 5, pp. 13–17, 2019.

[7] J. Huang, C.-C. Hung, S.-R. Kuang, Y.-N. Chang, K.-Y. Huang, C.-R. Tsai, and K.-L. Feng, "The prototype of a smart underwater surveillance system for shrimp farming," in *2018 IEEE International Conference on Advanced Manufacturing (ICAM)*, pp. 177–180, IEEE, 2018.

[8] J. Capelo, E. Ruiz, V. Asanza, T. Toscano-Quiroga, N. N. Sánchez-Pozo, L. L. Lorente-Leyva, and D. H. Peluffo-Ordóñez, "Raspberry pi-based internet of things for shrimp farms real-time remote monitoring with automated system," in *2021 International Conference on Applied Electronics (AE)*, pp. 1–4, IEEE, 2021.

[9] S. E. Martin, "Protecting electrical and electronic systems from moisture damage," in *OCEANS 2009*, pp. 1–3, IEEE, 2009.

[10] R. Comizzoli, R. Frankenthal, P. Milner, and J. Sinclair, "Corrosion of electronic materials and devices," *Science*, vol. 234, no. 4774, pp. 340–345, 1986.

[11] N. Instruments, "Application note 078: Strain gauge measurement - a tutorial," *National Instruments Corporation*, 1995.

[12] Avia Semiconductors, *24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales*, 2011.

[13] J. Lee, H. Nazki, J. Baek, Y. Hong, and M. Lee, "Artificial intelligence approach for tomato detection and mass estimation in precision agriculture," *Sustainability*, vol. 12, no. 21, 2020.