UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN, MENCIÓN CONTROL DE PROCESOS

Tema: "CONTROLADOR PARA ALIMENTACIÓN DE PECES EMPLEANDO DEEP LEARNING"

Trabajo de Titulación previo a la obtención del Grado Académico de Magister en Electrónica y Automatización, mención Control de Procesos.

Modalidad de titulación: "Proyecto de Desarrollo"

Autor: Ing. Willam Patricio Cañar Yumbolema

Director: Ing. Eddie Egberto Galarza Zambrano, Mg.

Ambato – Ecuador

2022

APROBACIÓN DEL TRABAJO DE TITULACIÓN

A la Unidad Académica de Titulación de la Facultad de Ingeniería en Sistemas Electrónica e Industrial

El Tribunal receptor de la Defensa del Trabajo de Titulación presidido por la Ingeniera Elsa Pilar Urrutia Urrutia, Magíster e integrado por los señores: Ingeniero Carlos Diego Gordon Gallegos, Ph.D e Ingeniero Mario Geovanny García Carrillo, Magister designados por la Unidad Académica de Titulación de Postgrado de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, para receptar el Trabajo de Titulación con el tema: " Controlador para alimentación de peces empleando Deep Learning", elaborado y presentado por el señor, Ingeniero Willam Patricio Cañar Yumbolema, para optar por el Grado Académico de Magister en Electrónica y Automatización Mención Control de Procesos; una vez escuchada la defensa oral del Trabajo de Titulación el Tribunal aprueba y remite el trabajo para uso y custodia en las bibliotecas de la Universidad Técnica de Ambato.

Firmado electrónicamente por:
ELSA PILAR URRUTIA

Ing. Elsa Pilar Urrutia Urrutia, Mg
Presidente y Miembro del Tribunal de Defensa

Firmado electrónicamente por:
CARLOS DIEGO GORDON GALLEGOS

Ing. Carlos Diego Gordón Gallegos, Ph.D
Miembro del Tribunal de Defensa

Firmado electrónicamente por:
MARIO GEOVANNY GARCIA CARRILLO

Ing. Mario Geovanny García Carrillo, Mg
Miembro del Tribunal de Defensa

AUTORÍA DEL TRABAJO DE TITULACIÓN

La responsabilidad de las opiniones, comentarios y críticas emitidas en el Trabajo de Titulación presentado con el tema: **"CONTROLADOR PARA ALIMENTACIÓN DE PECES EMPLEANDO DEEP LEARNING"**, le corresponde exclusivamente a: Ingeniero Willam Patricio Cañar Yumbolema, Autor bajo la Dirección de Ingeniero Eddie Egberto Galarza Zambrano Magister, Director del Trabajo de Titulación; y el patrimonio intelectual a la Universidad Técnica de Ambato.

Firmado electrónicamente por:
WILLAM PATRICIO
CAÑAR YUMBOLEMA

Ing. Willam Patricio Cañar Yumbolema

AUTOR

Firmado
digitalmente por
EDDIE EGBERTO
GALARZA
ZAMBRANO

Ing. Eddie Egberto Galarza Zambrano, Mg.

DIRECTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que el Trabajo de Titulación, sirva como un documento disponible para su lectura, consulta y procesos de investigación, según las normas de la Institución.

Cedo los Derechos de mi Trabajo de Titulación, con fines de difusión pública, además apruebo la reproducción de este, dentro de las regulaciones de la Universidad Técnica de Ambato.

Firmado electrónicamente por:
WILLAM PATRICIO CAÑAR YUMBOLEMA

Ing. Willam Patricio Cañar Yumbolema

CC. 0202046779-3

ÍNDICE GENERAL

CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

AGRADECIMIENTO

A Dios por prestarme salud, vida e inteligencia para continuar sirviendo como un buen ser humano en este mundo.

A mi mamá Rosa, a mi padre Miguel por darme sus sabias palabras y consejos en mi desarrollo profesional y humano integral, desde luego por confiar siempre en mi capacidad y habilidad.

A Jessenia por darme ese aliento, esa motivación con mucho amor para levantar la cabeza en ocasiones de dificultad y avanzar en lo propuesto.

A mis hermanos, siempre fueron ese pilar fundamental con su ejemplo, de esfuerzo y dedicación para obtener un logro profesional.

A mis hijos Aaron, Willam, Kevin por su existencia, permitirme motivar cada paso de mi vida, siendo mi razón de progresar en la vida.

A la Universidad Técnica de Ambato por permitirme alimentar de conocimientos teóricos y prácticos, en especial a mis maestros quienes con esfuerzo y vocación impartieron su sabiduría la cual será plasmada en cada uno de mis actos a lo largo de mi vida profesional.

Al Mg. Patricio Córdova quien estuvo al frente de los maestrantes con su responsabilidad y liderazgo para apoyar y guiar oportunamente en los diferentes procesos académicos.

Una fraterna gratitud al Mg. Eddie Galarza quien con su paciencia y conocimiento experimentado supo aportar y guiar en mi trabajo de titulación.

Willam Patricio

DEDICATORIA

Este trabajo plasmado, fruto de mi dedicación, disciplina y perseverancia lo dedicado con todo el sentir de mi alma:
A mis hijos, son quien me han impulsado en las derrotas, en los bajones, en la crisis, permitiendo levantarme y avanzar con ánimo para terminar mi trabajo de titulación por ellos es este logro académico alcanzado.

Willam Patricio

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN, MENCIÓN CONTROL DE PROCESOS

TEMA:

"CONTROLADOR PARA ALIMENTACIÓN DE PECES EMPLEANDO DEEP LEARNING"

AUTOR: Ing. Willam Patricio Cañar Yumbolema

DIRECTOR: Ing. Eddie Egberto Galarza Zambrano Mg.

LÍNEA DE INVESTIGACIÓN:

- Tecnología de la información y sistemas de control.

FECHA: 14 marzo 2022

RESUMEN EJECUTIVO

El presente proyecto de titulación consiste en el diseño e implementación de un controlador para alimentación de peces empleado Deep Learning para lo cual se utilizó un miniordenador Raspberry y una cámara Web de bajo costo económico, esto redujo notablemente la inversión para desarrollo del proyecto. El objetivo principal consiste en la creación de un Dataset (conjunto de imágenes) etiquetado manualmente de varios peces tipo cebra (Danio Rerio) ubicados dentro de una pecera iluminada uniformemente mediante luz led blanca. En este caso se decidió utilizar dos videos con la presencia de 4 y 6 peces, respectivamente. Mediante la utilización de un algoritmo computacional se obtuvieron la secuencia de imágenes de los peces donde se puede identificar sus movimientos. Esta información es utilizada para entrenar una red convolucional mediante el algoritmo de detección de objetos en imágenes ACF (Características Agregadas de Canal). Una vez determinada la ubicación de los peces

dentro de la pecera se procede a identificar el sentimiento del cardumen mediante la implementación de tres zonas, es decir, el algoritmo desarrollado permitirá saber si los peces están en una zona de satisfacción, zona normal o una zona de alimentación. Finalmente, los índices FuzzySN, FuzzySH y FuzzyNH contienen el sentimiento de los peces y son las entradas de un controlador difuso (fuzzy) que a su vez contiene las reglas de alimentación basado en el comportamiento natural de los peces; de esta manera el sistema desarrollado es capaz de alimentar a los peces de forma automática. El error mínimo de identificación alcanzado fue de 29.5% pero la identificación del comportamiento del cardumen de peces tuvo un acierto del 100%. La prueba realizada para validar el algoritmo se dio para un caso de alimentación manual por parte de un operario, donde el sistema logro identificar correctamente el sentimiento del cardumen como satisfecho.

Descriptores: ACF, Algoritmo, Cámara web, Difuso, Deep Learning, Darnio Rerio, Dataset, Fuzzy, Procesamiento de imágenes, Matlab, Peces, Raspberry, Visión Artificial.

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

MAESTRÍA EN ELECTRÓNICA Y AUTOMATIZACIÓN, MENCIÓN CONTROL DE PROCESOS

THEME:

"CONTROLADOR PARA ALIMENTACIÓN DE PECES EMPLEANDO DEEP LEARNING"

AUTHOR: Ing. Willam Patricio Cañar Yumbolema

DIRECTED BY: Ing. Eddie Egberto Galarza Zambrano Mg.

LINE OF RESEARCH:

- Tecnología de la información y sistemas de control.

DATE: March 14th, 2022

EXECUTIVE SUMMARY

This degree project consists of the design and implementation of a controller for fish feeding using Deep Learning for which a Raspberry minicomputer and a low-cost Web camera were used, this significantly reduced the investment for project development. The main objective consists in the creation of a manually labeled dataset (set of images) of several zebra-type fish (Danio Rerio) located inside a fish tank uniformly illuminated by white LED light. In this case, it was decided to use two videos with the presence of 4 and 6 fish, respectively. Through the use of a computational algorithm, the sequence of images of the fish where their movements can be identified were obtained. This information is used to train a convolutional network using the ACF (Aggregated Channel Characteristics) image object detection algorithm. Once the location of the fish inside the fish tank is determined, the feeling of the school is identified through the implementation of three zones, that is, the

developed algorithm will allow knowing if the fish are in a satisfaction zone, a normal zone or a normal zone. feeding. Finally, the FuzzySN, FuzzySH and FuzzyNH indices contain the feeling of the fish and are the inputs of a fuzzy controller which in turn contains the feeding rules based on the natural behavior of the fish; In this way, the developed system is capable of feeding the fish automatically. The minimum identification error reached was 29.5%, but the identification of the behavior of the school of fish had a success rate of 100%. The test carried out to validate the algorithm was given for a case of manual feeding by an operator, where the system was able to correctly identify the feeling of the school as satisfied.

Keywords: Descriptors: ACF, Algorithm, Web camera, Fuzzy, Deep Learning, Darnio Rerio, Dataset, Fuzzy, Image processing, Matlab, Fish, Raspberry, Artificial Vision.

INTRODUCCIÓN

Actualmente muchos de los procesos industriales se pueden automatizar conforme al avance tecnológico, prediciendo a acciones a realizar en la planta, el cual permita aprovechar tiempos de producción y reducción gastos incensarios, aportando a la mejora continua de los sistemas de producción en específico crianza de peces.

El siguiente proyecto está dirigido al control inteligente para la alimentación de peces aplicando técnicas de inteligencia artificial, visión artificial electrónica de potencia y sistemas embebidos (Galán et al., 2000). La metodología de investigación aplicada en este proyecto de tesis fue el hipotético deductivo que se basa en el método científico y consiste en un análisis empírico del comportamiento de los peces aplicando las técnicas investigativas de entrevista informal a personas capacitadas en la crianza y venta de peces. Las técnicas de observación directa, observación de campo y experimentación fueron aplicadas para determinar el comportamiento de los peces cuando estos necesitan alimentarse, también se realiza una investigación sobre el comportamiento de los peces antes, durante y después de la alimentación. Cabe recalcar que en esta fase intervienen varios factores biológicos y fisiológicos como: producción de feromonas, estímulos externos, tamaño de cardumen, balance de dieta y localización de alimento. Mediante la revisión bibliográfica se pudo determinar que los peces más utilizados en proyectos de experimentación científica son los peces Cebra o también llamado Danio Rerio (Cayuel Fuentes et al., 2012; Palomino Echeverría, 2018; Vargas-Vargas, 2017).

El controlador desarrollado en el presente proyecto permitió identificar características específicas del comportamiento que presentan los peces antes, durante y después de un proceso de alimentación con el objetivo de aprender alimentarlos inteligentemente según su comportamiento y garantizar una alimentación óptima y por ende puedan desarrollarse de forma correcta. Por otra parte, el entrenamiento de la red neuronal fue realizado con aproximadamente dos mil imágenes en distintas posiciones de los peces cebra y otras dos mil imágenes de la pecera, pero sin la presencia de los peces. Por último, ese documento presenta una comparación del algoritmo Deep

Learning desarrollado con la plataforma VIAME Web utilizada para el análisis de superficies marinas (Dawkins et al., 2017; Pedersen et al., 2019)

Este documento se encuentra dividido de la siguiente manera: el capítulo I se enfoca en la introducción y la justificación de esta investigación considerando aspectos científicos teóricos y experimentales. Además, se describen los objetivos planteados tanto el general como los específicos.

El capítulo II contiene información sobre los antecedentes investigativos donde se describen cinco investigaciones con temas relevantes y relacionados con la temática de la cual se obtiene valiosa información tanto de materiales, metodologías como resultados esperados de esta investigación. El capítulo III contiene información sobre el marco metodológico aplicado en esta investigación donde se puede encontrar aspectos como ubicación, materiales y equipos, técnicas de investigación y el análisis de los resultados.

El capítulo IV contiene los resultados obtenidos del procesamiento de imágenes, reconocimiento de los peces a través visión artificial utilizando OpenCv en el software Python, entrenamiento de un sistema Deep Learning para el reconocimiento del comportamiento de los peces. Por último, se describen las conclusiones y recomendaciones obtenidas durante el desarrollo del presente tema de investigación.

CAPITULO I

EL PROBLEMA DE INVESTIGACION

1.1 Tema de investigación

Controlador para alimentación de peces empleado Deep learning

1.2 Planteamiento del problema

1.2.1 Contextualización

La edición de 2020 del estado mundial de la pesca y la acuicultura sigue demostrando el papel significativo y creciente que desempeñan la pesca y la acuicultura en la provisión de alimentos, nutrición y empleo. También demuestra los principales desafíos que deben abordarse a pesar de los progresos realizados en varios frentes. encaminados a la sostenibilidad de la pesca y los ecosistemas como única solución que garantizará que las pesquerías de todo el mundo sean sostenibles y puedan ser automatizados empleando la tecnología a la vanguardia.

En Ecuador según el artículo 16 de la Ley Orgánica del Régimen de la Soberanía Alimentaria, establece que: "El Estado fomentará la producción pesquera y acuícola sustentable, y establecerá las normas de protección de los ecosistemas; asimismo señala que el Estado protegerá a todos los pescadores incluyendo a los industriales, artesanales, recolectores comunitarios y estimulará la adopción de prácticas sustentables de reproducción en cautiverio de las especies de mar, río y manglar, así como la implementación de sistemas inteligentes en las piscinas de crianza y producción de peces de acuerdo a la zona de producción y localidad de la piscicultura siendo este en región: Costa, Sierra, Oriente, insular o Galápagos.

A nivel local en Tungurahua existe un reducido número personas dedicadas a la crianza de peces para fines productivos por diversos factores entre ellos climáticos, sin embargo existen lugares de comercialización de peces a nivel doméstico el cual requieren de sistemas que empleen tecnología y aplicaciones de inteligencia artificial para alimentación con visión a futuro aportando al desarrollo y calidad de vida de los

peces por ende a la economía del pueblo siempre con un enfoque claro de facilitar las cosas al hombre.

Es importante contar con un controlador inteligente para alimentación de peces empleando Deep Learning que logre predecir y tomar decisiones asertivas cuando el grupo de peces necesita ser alimentado, tomando en cuenta su comportamiento dependiendo su estadía en la zona de control.

1.2.2 Análisis Critico

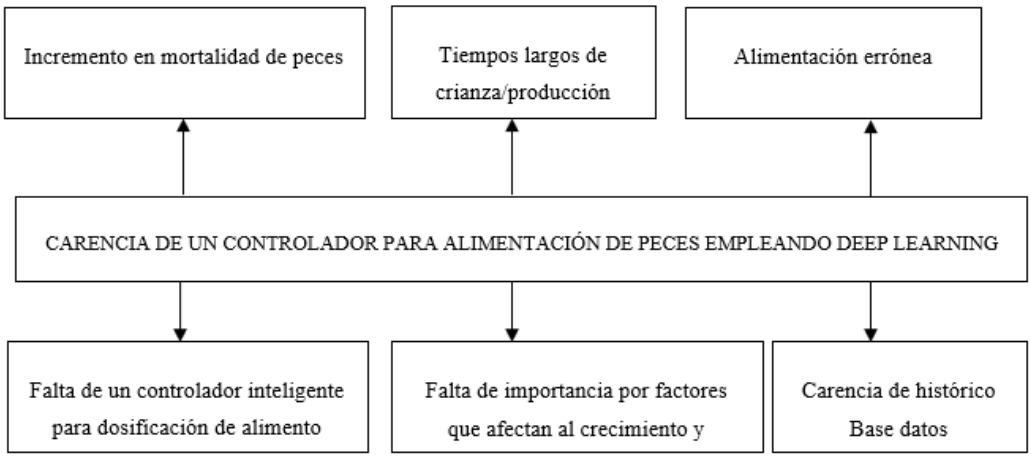


Figura 1. Diagrama de Árbol de Problema

Elaborado por: Ing. Willam Cañar

Como se muestra en el grafico anterior Figura 1, el problema identificado es la carencia de un controlador para alimentación de peces empleando Deep Learning, aun existiendo lugares de crianza de peces en su mayoría alimentación manual.

Entre las principales causas y consecuencias analizadas en cuanto al problema identificado se encuentran las siguientes:

La falta de un controlador inteligente para dosificación de alimento para los peces es el incremento de la mortalidad o disminución paulatina de la calidad de vida los peces.

También la falta de importancia por los factores que afectan al crecimiento y desarrollo de los peces su consecuencia es ocasionar tiempos largos de crianza y producción.

Finalmente la carencia de históricos o una base de datos de comportamientos de los peces cuya consecuencia se deriva en un tipo de alimentación erróneo.

1.2.3 Prognosis

Uno de los problemas mas comunes recae en la producción de peces, afectando a la economía de personas que se dedican a la crianza de peces, donde se ha venido realizando los registros de forma manual según la observación directa y/o experiencia del operario, donde se cree que el horario y la cantidad de alimentación es la adecuada, sin tomar en cuenta su comportamiento antes durante y después de ser alimentados.

Razón por la cual el desarrollo del proyecto propuesto tiene como finalidad ayudar a la anticipación y toma de decisiones ya que este estaría dosificando la alimentación de manera automática de acuerdo a su comportamiento, es decir según el número de peces que se ubican en una misma zona o espacio de control de esta amanera estaría dosificando la cantidad exacta de alimento garantizando un desarrollo optimo.

En tal virtud, este tipo de sistemas inteligentes podría usarse en pisciculturas o lugares que dedican a crianza de peces domestico sin embargo cabe recalcar se ha realizado el estudio en específico para tipo de pez cebra, como también existen estudios relacionados de crianza de peces cebra para fines médicos, considerando que se efectuara el sistema en software libre y sistema embebido donde se reduce en parte el costo del mismo.

1.2.4 Formulación del problema

¿Cómo incide el controlador empleando Deep learning para la alimentación de peces?

1.2.5 Interrogantes de la investigación

¿De que forma se puede controlar el comportamiento de los peces para ser alimentados?

¿Cuál es el algoritmo más adecuado para identificar características útiles y efectuar el procesamiento de imágenes?

¿Cómo se podría evaluar la solución brindada al problema presentado?

1.2.6 Delimitación del Objeto de Investigación

Campo: Alimentaria de peces

Área: Aplicación algoritmo Deep learning para alimentación de peces

Aspecto: Reconocimiento estimulo del cardumen.

Delimitación espacial: Pisciculturas y crianza de peces domestico particulares

1.3 Justificación

Actualmente nuestro país y todo el mundo está atravesando una problemática común como es la pandemia Covid19, ocasionado problemas socioeconómicos y de salud, debido a este acontecimiento la población ha visto la necesidad de buscar diferentes fuentes de ingresos, pese a la falta de recursos, muchas personas han tomado la decisión de crear pequeños emprendimientos. Además se ha visto que empresarios de magnitud grande, medianos, pequeños, y nuevos emprendedores apuestan su inversión en el campo agronómico y pecuario, crianza y productores de animales entre otros, siendo estos campos menos afectados por el hecho de ser indispensables para el desarrollo sostenible del país en cuestiones socio económicas y alimentarias de la población, los cuales últimamente han tenido una gran demanda de materiales, maquinaria y recursos para abastecer con normalidad en línea productiva.

La economía ecuatoriana se ha caracterizado por ser proveedora de materias primas en el mercado internacional y al mismo tiempo importadora de bienes y servicios de mayor valor agregado. Los constantes e imprevistos cambios en los

precios internacionales de las materias primas, así como su creciente diferencia frente a los precios de los productos de mayor valor agregado y alta tecnología, han colocado a la economía ecuatoriana en una situación de intercambio desigual sujeta a los vaivenes del mercado mundial. (SENPLADES, 2012). Es por esta razón que se pretende realizar el siguiente proyecto, para contribuir a la transformación de la matriz productiva, teniendo como recurso principal el proveer de un controlador óptimo para la crianza de peces.

En el mercado ecuatoriano no existen controladores digitales para el monitoreo y alimentación inteligente y que disminuya los índices de mortalidad de peces (Mecánica, 2015). De igual forma se añade que los módulos controlados también han ido evolucionando en países desarrollados hasta un escenario donde la aplicación de inteligencia artificial se ha convertido en un elemento fundamental dentro del proceso productivo (Puig, 2011), ya que la producción de peces depende en gran medida de un buen sistema de alimentación según el comportamiento del pez. El sistema desarrollado será capaz de identificar el comportamiento de los peces para activar un mecanismo de alimentación, minimizando el índice de mortalidad de los peces. El diseño del sistema de control se adaptará eficazmente a procesos de crianza de peces en sintonía con las nuevas tecnologías, es decir a través de técnicas de inteligencia artificial. En este proyecto se obtendrá dicho objetivo a través de un modelamiento del prototipo de un estanque modular para peces y así obtener su respuesta a la acción de control de acuerdo a su diseño (De O. B. Neto et al., 2010). De igual forma, este proyecto aportará de manera directa a pequeños y medianos productores acuícolas y así obtener mejores resultados, ya que un buen control de alimentación inteligente, reducirá el índice de mortalidad y mejorará la calidad del pez, según el estudio de (Oviedo, 2012) y (Machado Barrera, 2019), lo que garantizará una sostenibilidad económica del productor.

1.4 Objetivos General y específicos:

1.4.1 General:

Desarrollar un controlador para alimentación de peces empleado Deep Learning.

1.4.2 Específicos:

- Investigar los fundamentos teóricos, aplicaciones, técnicas y algoritmos de inteligencia artificial con enfoque en acuicultura.
- Diseñar el controlador empleando Deep Learning para la alimentación de peces.
- Implementar el controlador y respectivo sistema embebido.
- Realizar pruebas comparativas y validarlas mediante técnicas estadísticas.

CAPÍTULO II

ANTECEDENTES INVESTIGATIVOS

2.1 Fundamentación teórica

Para desarrollar el presente proyecto, es importante recabar información y argumentos sólidos que abarquen a temáticas sobre el ámbito de crianza de peces, para fines alimenticios, reproductivos, médicos y fines de estudio comportamental de los peces para una alimentación inteligente, así como la implementación de sistemas de visión e inteligencia artificial.

El autor Guio Rodríguez elaboró el proyecto de tesis titulado "DESARROLLO DE UN SISTEMA DE SEGUIMIENTO DE LA NAVEGACIÓN DE PECES CEBRA EN 3 DIMENSIONES USANDO APRENDIZAJE PROFUNDO" en la Universidad Santo Tomás. La problemática de este proyecto de tesis está enmarcada en el campo de la medicina, debido a que se utiliza los peces cebra como elementos de experimentación en el tratamiento del Alzheimer (AD-Alzheimer Disease). Los científicos encargados de este proyecto requieren de un sistema que pueda identificar y realizar un seguimiento de los peces dentro de una pecera. Los materiales utilizados en este proyecto son: una cámara web, computadora de alto rendimiento, iluminación, pecera y varios peces tipo cebra. Este proyecto consiste en la creación de un dataset en base a videos capturados previamente en otro proyecto de investigación, a partir de estas dos bases de datos se da inicio a un procesamiento de imágenes que consistió en extraer frames cuando el pez cebra se pueda observar claramente. Una vez creado esta información se procede entrenar una red convolucional utilizando la configuración YoloV3 y YoloV3-tiny, cabe recalcar que esta arquitectura requiere de parámetros adicionales sobre la ubicación relativa del pez en cada una de los frames del dataset. El autor concluye que las redes convolucionales ofrecen buenas características para dar solución a los problemas de identificación de objetos dentro de una imagen, es decir, a medida que se aumentan las imágenes del dataset los resultados de identificación mejoran de la misma manera. (Guio Rodríguez, 2021)

Siguiendo con la investigación, el autor Martínez Peiró ha desarrollado el proyecto titulado “TÉCNICAS AVANZADAS DE VISIÓN POR COMPUTADOR (VC) BASADAS EN DEEP LEARNING (DL) APLICADAS A LA MONITORIZACIÓN DE ESPECIES MARINAS (BLUEFIN TUNA)” en la Universidad Politécnica de Valencia. La problemática de este proyecto se origina cuando el autor identifica que no existe material científico orientado a explorar las arquitecturas, técnicas y metodologías de las redes neuronales convolucionales aplicadas en el campo detección y segmentación de objetos en imágenes. Los materiales utilizados únicamente consisten en una computadora de alto rendimiento. Este proyecto utiliza redes convolucionales (CNN) y la técnica Transfer Leraning para utilizar redes entrenadas previamente con una gran cantidad de datos (dataset). Adicionalmente se realiza una comparación de las detecciones obtenidas a través de los modelos Faster R-CNN, Mas R-CNN, Yolov5 y PointRed, cabe recalcar que todas estas técnicas de detección y segmentación dentro de una imagen fueron aplicadas para monitorizar automáticamente la biomasa de la especie marítima llamada Atún Rojo (Thunnus Thynnus). El autor concluye que las técnicas de CNN y Deep Learning (DL), el conjunto de datos Ground Truth desarrollado manual y específicamente para este proyecto con un control minucioso de las etapas de anotación y etiquetado, presentan excelentes resultados en identificación y segmentación de los peces. Los modelos con mejores resultados en detección de objetos en imágenes son Faster R-CNN, Mask R-CNN y Yolov5, por otro lado, para segmentación de objetos en imágenes los mejores son Mask R-CNN y PointRend. (Martínez Peiró, 2021)

El proyecto desarrollado por el autor Álvaro Vidaor con el título “DESARROLLO DE UN SISTEMA DE DETECCIÓN DE PECES” en la UNIVERSIDAD POLITÉCNICA DE CATALUÑA. Este proyecto se origina debido a la ausencia de un sistema de detección automático para identificar octópodos utilizando visión artificial. Los materiales utilizados en este proyecto de tesis es una computadora de alto rendimiento. El funcionamiento de este proyecto inicia con la creación de una base de datos de los octópodos (pulpos) la misma que tiene tres etapas que son: entrenamiento, validación y test. El reconocimiento de este tipo de animal se realiza mediante la utilización de una red neuronal llamada Keras-Retinanet implementada de dos formas diferentes. Los detectores desarrollados también se dividen en la fase entrenamiento y la fase de pruebas, de esta manera se puede

determinar la configuración con mejores resultados a la hora de utilizar nuevas imágenes de los pulpos. El autor concluye que los resultados obtenidos demuestran que las dos implementaciones generan una identificación de los pulpos por encima del 80%. Los entrenadores desarrollados no generan Overfitting, es decir, se puede aumentar la base de datos inicial sin problemas. Finalmente, el autor enuncia que el método Keras Retinanet no es tan eficiente como el Keras-Mask RCNNN, debido a que esta última no genera tantos falsos positivos. (Vidaor Maristany, 2020)

Los autores Shilpi Prerana y Brejesh Lall han desarrollado el proyecto titulado "DFTNET: DEEP FISH TRACKER WITH ATTENTION MECHANISM IN UNCONSTRAINED MARINE ENVIRONMENTS" publicado en la revisa IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT. La problemática de este proyecto está enfocada en la Administración Nacional Oceánica y Atmosférica donde se establece que el Océano cubre una superficie del 70% de la Tierra y, es el encargado de regular la temperatura, el clima y permitiendo la vida de todos los seres vivos. Sin embargo, el medio marítimo ha sido ampliamente afectado por los seres humanos, llegando incluso a extinguir determinadas especies marítimas; por este motivo es muy importante el desarrollo de mecanismo de observación automática de ecosistemas marítimos para realizar un seguimiento de los peces. Los materiales utilizados en este proyecto de tesis son una computadora de alto rendimiento y software de programación. El funcionamiento de este proyecto consiste en el desarrollo de una red de seguimiento de peces denominada (DFTNet), que tiene una operabilidad definida por los autores como Siamesa, es decir, combina la similitud de la apariencia de los peces con la atención de la red de memoria a corto plazo y de esta manera capturar los movimientos de los peces en los siguientes fotogramas. Los autores realizan una serie de pruebas de funcionamiento del DFTNet y los datos son comparados con otro proyecto de identificación de peces denominado Fish4Knowledge, llegando a obtener una reducción significativa del 60.9% en los interruptores de identificación (ID). Finalmente, los autores realizan una conclusión sobre el buen desempeño de la herramienta DFTNet para realizar el seguimiento de múltiples peces en la vida real y en entornos marinos difíciles.(Gupta et al., 2021)

Otro proyecto relacionado al tema de investigación es el desarrollado por los autores Nawaf Farhan, Abdullah Zawawi y Mohd Azam con el título "IMPROVED

DEEP LEARNING FRAMEWORK FOR FISH SEGMENTATION IN UNDERWATER VIDEOS" publicado en la revista internacional Ecological Informatics. La problemática de este proyecto de investigación está enfocada en la necesidad que tiene la comunidad científica por el desarrollo de métodos efectivos para reconocer peces y sus movimientos dentro de espacios marítimos; estos sistemas son muy importantes cuando se requiere medir las restricciones aplicas en áreas donde los peces se encuentran en peligro de extinción debido alteraciones de su habitad y sobre todo a las amenazas industriales. Los materiales utilizados en este proyecto de investigación son una computadora de alto rendimiento y videos de los peces en un entorno marítimo. El funcionamiento de este proyecto de investigación consiste en un método mejorado de reconocimiento de objetos en imágenes, el mismo que cuenta con cuatro etapas muy importantes: 1) Pre procesamiento de imágenes, esta etapa es aplicada con el objetivo de eliminar factores externos de los videos como iluminación innecesaria, brillo, ruido, entre otros. 2) Se aplica un método de Deep Learning usando RESENT para la detección de peces. 3) Se extiende el funcionamiento a múltiples peces mediante la utilización de la arquitectura Regional Proposal Network (RPN). 4) Esta etapa consiste en la aplicación de un método de segmentación dinámica. Finalmente, los autores concluyen que los resultados del algoritmo desarrollado en este artículo tienen una mejor capacidad de rendimiento con respecto a otros modelos de segmentación en instancias de múltiples peces. (Alshdaifat et al., 2020)

Los autores Fathima Syreen y Merriliance desarrollaron una investigación titulada "DEEP CONVOLUTIONAL NETWORKS FOR UNDERWATER FISH LOCALIZATION AND SPECIES CLASSIFICATION" publicada en la revista International Journal of Advanced Research in Engineering and Technology (IJARET). El problema planteado en esta investigación está enfocado en la dificultad que existe para identificar peces dentro de un ambiente marítimo con obstáculos como piedras, flora excesiva, sombras, distintos tipos de iluminación, factores que se encuentran en un entorno real de los peces. Los materiales utilizados en esta investigación son una computadora y una base de datos de videos en entornos marítimos. El funcionamiento de este proyecto inicia por una revisión bibliográfica donde los autores argumentan que el reconocimiento de peces vivos es una tarea difícil debido a las diferentes características que presenta el mar abierto, por lo que deciden realizar su investigación en un ambiente limitado. La técnica propuesta se denomina

VGG-16 con arquitectura DeppFish para mejorar su rendimiento y la extracción de características. Este modelo consta de dos fases: localización de los peces a través de la red VGG-16 y posteriormente se aplica una clasificación de los peces mediante DeepFish. Luego, el sistema aplica un proceso de vectorización apoyado en un clasificador forestal aleatorio que le permite clasificar de mejor manera los peces encontrados. Finalmente, los autores concluyen que los resultados experimentales obtenidos demuestran que el modelo VGG-16 con arquitectura DeepFish tienen una precisión máxima de 99.47% para detectar peces. (Fathima Syreen & Merriliance, 2020)

Otro proyecto relación con Deep Learning para reconocimiento de peces es el desarrollado por los autores Wenbo Zhang, Chaoyi Wu y Zhenshan Bao con el título "DPANET: DUAL POOLING-AGGREGATED ATTENTION NETWORK FOR FISH SEGMENTATION" publicado en The Institution of Engineering and Technology. La situación problemática trata en este proyecto de investigación está relacionada con la necesidad de contar con sistemas de seguimiento de los hábitats de peces con el objetivo de proteger la sostenibilidad de las pesquerías comerciales en entornos marinos, de esta manera se podrá evitar daños a los ecosistemas de los peces. Los materiales utilizados son una computadora de alto rendimiento y una base de datos de videos con peces en entornos marinos. Este proyecto consiste en el desarrollo de una red de atención agregada de agrupación dual (DPANet) que sirve para capturar de forma adaptativa dependencias de largo alcance a través de una manera eficiente y amigable con la computación, generando un costo computacional muy bajo. DPANet contiene dos módulos diferentes; el de posición de agrupación-agregado y un módulo de atención de canal de agrupación-agregado, estos dos módulos sirven para generar contextos de dimensión espacial y dimensión de canal, respectivamente. Mediante el desarrollo de pruebas exhaustivas los autores validan el funcionamiento de DPANet utilizando el conjunto de datos de imágenes de peces de DeepFish, así como también las imágenes subacuáticas de SUIM, logrando una puntuación de IOU (métrica para determinar la superposición entre la máscara objetivo y la predicción deseada) de 91.08% y 85.39%, respectivamente. (Zhang et al., 2021)

Los autores Hongwei Qin et al., han desarrollado un proyecto de investigación titulado "DEEPFISH: ACCURATE UNDERWATER LIVE FISH RECOGNITION

WITH A DEEP ARCHITECTURE" y fue publicado en la revista de Neurocomputing the ScienceDirect. La problemática de este proyecto se centra en la dificultad que existe para detectar peces bajo el agua y la demando que tienen este tipo de algoritmos. Sin embargo, se debe considerar que el entorno natural sin restricciones es muy complejo de analizar. El objetivo general de este proyecto consiste en el desarrollo de un algoritmo capaz de reconocer peces a partir de una serie de videos obtenidos mediante cámaras submarinas desplegadas en el océano. El procesamiento de imágenes aplicado en este proyecto consiste en la descomposición matricial escasa y de bajo rango de las imágenes a través de la Deep Learning, posteriormente se aplica un análisis de componentes principales (PCA) mediante dos capas convolucionales seguido de un hash binario en la capa no lineal e histogramas de bloques en la capa de agrupación de características. Luego se aplica SPP (agrupación de pirámide espacial) y SVM (clasificador lineal). Finalmente, el algoritmo es puesto a prueba con un dataset de reconocimiento donde se obtuvo un acierto del 98.64% (Qin et al., 2016).

Continuando con la revisión bibliográfica, los autores Ishaq, Sajith & Wahlby han desarrollado un artículo investigativo con el título "DEEP FISH: DEEP LEARNING–BASED CLASSIFICATION OF ZEBRAFISH DEFORMATION FOR HIGH-THROUGHPUT SCREENIN" publicado en la revista SLAS Discovery. Este proyecto consiste en el desarrollo de un algoritmo capaz de clasificar mediane imágenes y técnicas de Deep Learning al pez cebra (Danio Rerio). Este algoritmo de alto rendimiento tiene la capacidad de analizar la estructura física del pez obteniendo las deformaciones del pez de cuerpo entero y las placas de micro pocillos de otros peces. Este algoritmo es capaz de realizar un aprendizaje solamente con 84 imágenes con una precisión de 92.8% de detección y clasificación del pez cebra con respecto a otros peces. Finalmente, con un aumento del dataset los autores pudieron alcanzar un máximo de detección del 95% según las características especificadas por el usuario (Ishaq et al., 2017).

Los autores Suxia, et al., también han desarrollado un proyecto de investigación titulado "FISH DETECTION USING DEEP LEARNING" publicado en la revista Hindawi. La problemática de este proyecto consiste en la necesidad que tienen los Vehículos Submarinos Autónomos (AUV por sus siglas en inglés Autonomous Underwater Vehicle) para actualizarse y detectar peces de forma más

eficiente. El objetivo principal de este proyecto consiste en el desarrollo de un sistema integrado de detección de peces para ser incorporado en un vehículo AUV, esto permitirá adquirir información circundante de los sensores y tomar decisiones de forma automática. El procedimiento de procesamiento puede imitar las rutinas de aprendizaje del ser humano. Un sistema avanzado con más poder de cómputo puede facilitar la función de aprendizaje profundo, que explota muchos algoritmos de redes neuronales para simular cerebros humanos. En este artículo, se propuso un método de detección de peces basado en una red neuronal convolucional (CNN). El conjunto de datos de entrenamiento fue recolectado del Golfo de México por una cámara digital (Cui et al., 2020).

CAPÍTULO III

MARCO METODOLÓGICO

3.1 Ubicación

La revisión bibliografía de antecedentes investigativos ha permitido identificar que una de las mejores opciones para desarrollar un algoritmo de detección de objetos dentro de imagen utilizando seres vivos, como en estos casos peces; debe iniciar con una etapa de prototipo en un ambiente controlado. Por ese motivo, en este proyecto se decidió utilizar una pecera de 50 x 40 x 35 centímetros de largo, alto y ancho, respectivamente. Adicionalmente se acondiciono un espacio físico de 4 x 5 metros cuadros aproximadamente, donde se puede acceder a una buena iluminación tanto natural como artificial.

Por otro lado, la detección de peces en un entorno marítimo real requiere de una cantidad de equipos más desarrollados, por ejemplo, tarjetas que soporten inteligencia artificial y posean GPU para procesar la información. De la misma manera se podría describir los requerirnos biológicos de los peces en un entorno natural libre.

3.2 Equipos y materiales

Los materiales utilizados para el desarrollo del presente proyecto de trabajo de titulación son una Raspberry Pi 3B, cámara Web y una computadora ASUS-NotebookSKU de 16GB de Ram. Estos componentes se describen a continuación:

3.2.1 Raspberry Pi 3B

Es el controlador necesario para la implementación de un sistema de visión artificial con Deep Learning para la identificación de objetos en una imagen, cabe recalcar que este dispositivo debe ser compatible con el software *Matlab* y con los requerimientos establecidos en los objetivos de investigación. El miniordenador *Raspberry Pi 3B* cuenta con un procesador de cuatro núcleos de 64 bits que se ejecuta a 1.5 Ghz, dos puertos USB 2.0 y dos puertos USB 3.0 lo que facilita la transferencia de datos a través de USB, adicionalmente cuenta con puertos HDMI. Este

miniordenador de gran potencia, es muy utilizado para desarrollar proyectos con *Tensorflow*, *PiHole*, *Minecraft* o *Kodi*. Este miniordenador necesita una fuente de 3A, así como disipadores de calor y buena ventilación ya que tiende a calentarse rápidamente (BricoGeek, 2021).

Estas razones justifican la utilización del miniordenador Raspberry Pi debido a que presenta una mejor capacidad de procesamiento con respecto a otros controladores, se tomó en cuenta los requerimientos de corriente y ventilación para garantizar un buen funcionamiento.

Figura 2. Miniordenador Raspberry Pi 3B.

Fuente: (Viera-Maza, 2017)

3.2.2 Webcam Logitech C922

La cámara debe ser seleccionada considerando los requerimientos mínimos que determina el software *Matlab*, donde se especifica una resolución de 1080x720 pixeles y un ángulo de visión mayor a 120 grados. El mini ordenador Raspberry Pi 3B puede utilizar las placas de expansión CM3/CM3+ para incluir otro tipo de periféricos, pero en este caso se optó por la utilización de una cámara Web Logitech serie C922 con un conector USB.

Figura 3. Cámara Web Logitech C922.

Fuente: (Seguritec, 2015)

Sus especificaciones técnicas son:

- ✓ Apertura (F): 2.35
- ✓ Longitud focal: 0.124 in.
- ✓ Ángulo de visión (diagonal): 160 grados
- ✓ Proporciona salida de potencia de 5Vdc
- ✓ Dimensiones: 0.984 in x 0.945 in

3.2.3 Asus X552LA Notebook

El desarrollo del algoritmo de detección de peces requiere de una computadora para el proceso del entrenamiento e implementación de la red neuronal con Deep Learning. Las características de la laptop se muestran en la Tabla 0.1.

Tabla 0.1. Características de la laptop ASUS.

Marca	Asus
Modelo	X552LA-SX850H
Procesador	Intel Core I5-4200U 1.6 GHz, (c/TB 2.6GHz)
Memoria RAM	4GB DDR3 1600 MHz SDRAM
Disco Duro	1 TB 5400 RPM
Pantalla	LED 15.6" HD 1366x768
Unidad óptica	DVD±RW

Tarjeta de video	Intel HD Graphics
Conectividad	• Ethernet 10/100/1000 Base T • Wi-Fi 802.11b/g/n • Bluetooth 4.0
Multimedia	• Cámara web integrada • Entrada para auriculares estéreo/micrófono
Puertos y Ranuras	• COMBO audio jack • USB 3.0 port(s) • RJ45 LAN Jack for LAN insert • HDMI
S.O.	Windows 8.1
Batería	4-Cell
Peso	2.3 Kg
Adicionales y/o Condición	Regalo de cortesía

Fuente: (Asus, 2019)

3.3 Tipo de Investigación

La investigación desarrollada es del tipo experimental debido a que se realizan varias pruebas de experimentación en los procesos de alimentación de 13 peces de tipo Cebra, modificando la hora de alimentación y la cantidad de alimento. Además, se aplica la observación directa y de campo para identificar la variación en su comportamiento, también se investiga la influencia del cardumen en el comportamiento de los peces.

Por otro lado, se realiza una investigación de las diferentes técnicas y métodos para los sistemas de visión artificial basado en Deep Learning y sus aplicaciones en la identificación de objetos dentro de una imagen, también se analizan los sistemas de implementación para sistemas embebidos.

Las técnicas aplicadas para el desarrollo del algoritmo se basan en una investigación aplicada, en la que se realizan procesos y sub procesos tales como: técnicas de iluminación, adquisición de imágenes, procesamiento de la imagen, etiquetado de los objetos, entrenamiento de la red neuronal, entre otros.

3.4 Población o muestra

La población del estudio del presente proyecto de investigación se encuentra conformada por la totalidad de 13 peces ubicados en una pecera de 50 x 40 x 35 centímetros de largo, alto y ancho, respectivamente. La zona de estudio está ubicada en la ciudad de Ambato donde se ha dispuesto una habitación ampliamente iluminada donde se encuentra implementado el sistema de visión artificial para alimentación de peces basado en Deep Learning. En la Figura 4 se puede observar la pecera con los peces de tipo cebra.

Figura 4. La pecera con los 13 peces cebra.

Fuente: Autoría Propia

3.5 Recolección de información

En esta investigación científica se decidió aplicar el método hipotético deductivo para obtener información sobre el proceso de crianza y alimentación de peces, este método consiste en la obtención de información empírica para luego validar dicha información a través de cálculos matemáticos y una revisión bibliográfica de proyectos relacionados. En la Figura 5 se puede observar las técnicas utilizadas durante esta etapa inicial de investigación.

Figura 5. Etapas de investigación.

Fuente: Autoría Propia

3.5.1 Observación directa y de campo

En esta sección se aplicó la técnica de observación directa y observación de campo. La observación directa fue aplicada a una pecera de 50 x 40 x 35 centímetros; un total de 15 peces del tipo Cebra fueron ubicados en la pecera. Estos peces fueron observados durante 45 días aplicando una alimentación cada día a las 8am y otra a las 8pm, de esta manera se pudo determinar la existencia de algún tipo de comportamiento antes, durante y después de realizar el proceso de alimentación, cabe recalcar que en esta etapa se utiliza una alimentación manual por parte de un mismo operario con el objetivo de que los peces aprendan este patrón.

La observación de campo se aplica para identificar la reacción de los peces cuando una persona diferente se acerque a la pecera, de la misma manera que el caso anterior, se busca identificar un determinado patrón de comportamiento en los peces. Adicionalmente se pretenden identificar la distribución de los peces dentro de la pecera, es decir, determinar la ubicación del cardumen en todo momento.

3.5.2 Revisión Bibliográfica

La revisión bibliográfica realizada se enfocó en la obtención de información relacionada con el comportamiento de los peces en la búsqueda y captura de alimento, para lo cual se ha considerado el trabajo realizado por Aguirre (2004), donde se investiga la interdependencia existente entre los peces y el medio acuático, condiciones jerárquicas entre los diferentes tipos de peces, presencia de depredadores y la generación de feromonas que se provoca por otras especies, movimientos en la corriente acuática generada por todos los peces que ocupan un espacio acuático. Todos estos factores influyen directamente sobre la búsqueda, localización y captura del alimento en los peces. Además, Aguirre (2004) dice que existen otros tipos de factores que influyen en la alimentación de los peces como sus hábitos de consumo, conductas de aprendizaje, entre otros factores. (Aguirre, 2004)

3.5.3 Entrevista

Se decidió aplicar la técnica de entrevista informal a la señora propietaria de un acuario en la ciudad de Ambato, donde se obtiene información sobre los procesos de crianza y alimentación de los peces de diferentes tipos. Esta información es muy importante para la realización de proyecto de investigación debido a que se cuenta con información de primera mano de una persona con amplia experiencia en la crianza de peces.

3.6 Procesamiento de la información

Los datos obtenidos se analizan de acuerdo con los siguientes procedimientos:

- Revisión crítica de la información obtenida aplicando las técnicas de investigación como: observación, revisión bibliográfica, entrevista y experimentación; de esta manera se pretende descartar información defectuosa, incompleta, contradictoria, no pertinente, etc.
- Tabulación o cuadros según la información obtenida durante la investigación donde se especifique las unidades, valores cuantitativos acorde al proceso de investigación.
- Manejo de la información, optimizando las tablas con los datos que tienen un aporte significativo al proceso de investigación.

Estudios estadísticos de datos para la presentación de los resultados.

3.7 Operación de variables

3.7.1 Variable independiente: Visión artificial con Deep Learning

Tabla 0.2. Operacionalización de la variable independiente.

CONCEPTO	DIMENSIONES	INDICADOR	ÍTEM	INSTRUMENTOS
Métodos de Deep Learning aplicados a la detección de objetos dentro de una imagen para identificar su comportamiento y accionar un mecanismo de alimentación.	Adquisición de imágenes	Numero de imágenes capturadas	¿Los equipos seleccionados están en la capacidad de adquirir imágenes?	Aplicación práctica
	Algoritmo Deep Learning	Precisión, tiempo, calidad, control	¿Cuáles son las técnicas de detección de objetos utilizadas?	Aplicación práctica
	Análisis e interpretación de los datos mediante un controlador Fuzzy	Número de imágenes procesadas	¿La información obtenida será fiable para el funcionamiento del controlador?	Aplicación práctica

Fuente: Autoría Propia

3.7.2 Variable dependiente: sistema de control para alimentación de peces

CONCEPTO	DIMENSIONES	INDICADOR	ÍTEM	INSTRUMENTOS
Ejecución de un conjunto de algoritmos que permitan determinar el comportamiento de los peces y accionar un mecanismo de	Peces	Ubicación de los peces en la pecera	¿Cuál es el comportamiento de los peces cuando están hambrientos?	Entrevista a los vendedores de peces. Revisión bibliográfica.
	Algoritmo de detección	Porcentaje de eficiencia del sistema	¿Los equipos elegidos estarán en la capacidad de procesar la	Aplicación práctica

alimentación cuando tengan hambre.			información necesaria?	
	Alimentación	Número de peces detectados	¿El sistema es capaz de identificar el comportamiento de los peces?	Aplicación práctica

Fuente: Autoría Propia

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

4.1 Selección de software

En esta sección se realiza una revisión de los programas que permiten desarrollar aplicaciones con visión artificial y Deep Learning, estos pueden ser licenciados o de código abierto. Cabe recalcar que el software seleccionado debe permitir su implementación en una Raspberry Pi 3B, con el objetivo de desarrollar un sistema embebido y se funcionamiento autónomo.

4.1.1 Matlab

Este software de procesamiento matemático dispone de un IDE propio basado en un lenguaje de programación nativo, es decir, un leguaje creado por MathWorks que solo puede ser utilizado dentro de esta plataforma. Este programa contiene dos aplicativos muy importantes que son el Simulink y GUIDE. Este programa posee la facilidad de desarrollar aplicaciones de visión artificial e implementarlas en una RaspBerry Pi de forma inalámbrica para que tenga un funcionamiento autónomo sin la necesidad de conectarse a una PC. Por otro lado, este software es licenciado, pero posee una opción de licencias para estudiantes, una opción que vuelve muy asequible de comprar.

4.1.1 LabView

Este programa fue desarrollado por la empresa Nacional Instruments y contiene unas librerías que permiten el desarrollo de aplicaciones con Visión Artificial desde una computadora. Este software está orientado a la adquisición de datos en tiempo real y consiste en una programación orientada a objetos, basado en el lenguaje de programación de C++. Visual Basic y .Net.

Este programa es licenciado y no dispone de una licencia para estudiantes económica, tiene un alto costo computacional debido a la cantidad de instrumentos virtuales y controladores que posee o están presentes durante la ejecución de un programa.

4.1.2 OpenCV

OpenCV es un programa de código abierto muy utilizado en la adquisición, procesamiento, análisis de imágenes y aplicaciones de visión artificial con Deep Learning. Una de las principales desventajas de este tipo de programas es la necesidad de contar con un procesamiento por GPU para mejorar el rendimiento de las aplicaciones desarrolladas y de esta manera reducir el costo computacional. El desarrollador de este proyecto es INTEL.

4.1.3 Costo de licencias

La selección adecuada del software para la implementación del presente proyecto de investigación presenta un verdadero desafío, debido a que cada una de las opciones ya sean programas licenciados o de código abierto presentan una serie de ventajas y desventajas para su implementación.

La utilización de un software con licencia ofrece innumerables ventajas tanto en implementación como en el tiempo de desarrollo y puesta en marcha del sistema de detección de objetos con Deep Learning. Además, este tipo de programas cuenta con un soporte técnico en línea para preguntas sobre el funcionamiento de algún tipo de algoritmo, dispone de fotos para debates, preguntas o desarrollo. Ciertamente el costo de una de estas licencias es muy elevado, pero programas como Matlab disponen de descuentos para estudiantes de instituciones universitarias. En la Figura 6 se puede observar el costo de una de estas licencias.

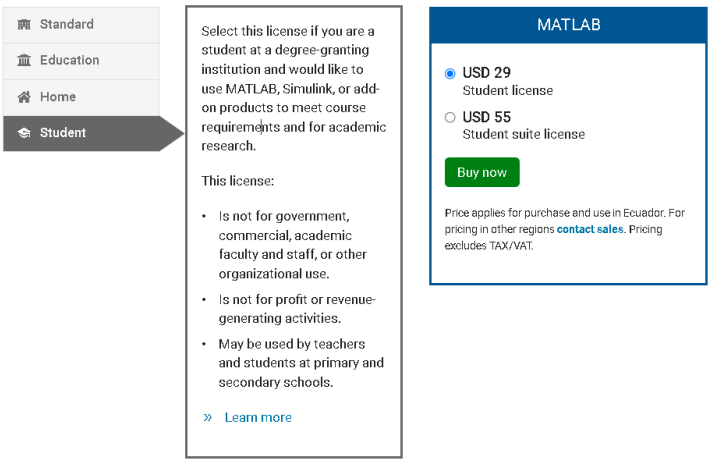


Figura 6. Licencias Matlab para estudiantes.

Fuente: Autoría Propia

Por otro lado, el software libre o de código abierto no requiere de la compra de una licencia para proceder a su utilización, pero tiene muchos limitantes en relación a librerías, información en línea, GUI (*Graphic User Interface*, Interface Gráfica de Usuario), entre otras características que no poseen este tipo de programas. Otra principal desventaja de este tipo de programas es la aparición de Bugs o más conocidos como errores de ejecución de algoritmos, también requieren de una serie de pasos de instalación y un análisis de compatibilidad de versiones entre el programa y las librerías a utilizarse.

Por este motivo el programa seleccionado para el desarrollo del presente proyecto de tesis es el software Matlab. A continuación, se describen sus ventajas más importantes:

- **Costo:** se puede adquirir licencias educativas desde 29 dólares con toolbox desde los 50 dólares como se muestra en la Figura 6.
- **Curva de aprendizaje:** Al poseer un lenguaje de programación propio, este programa tiene una curva de aprendizaje lenta, pero es muy importante considerar que durante el proceso de estudios universitarios la utilización de este programa es muy común; esto quiere decir que la curva de aprendizaje puede ser más corta en relación a otras opciones. Además, Matlab posee aplicativos para la adquisición de imágenes, procesamiento de información y entrenamiento de una red neuronal muy intuitivos.
- **Costo Computacional:** este ítem hace referencia al tiempo de ejecución de cada uno de los programas desarrollados en Matlab, cabe recalcar que este tiempo es más lento con relación a un programa de código abierto, pero se pueden aplicar técnicas de optimización de mejorar esta velocidad de ejecución.

4.2 Adquisición y etiquetado de imágenes

Una vez que se ha seleccionado adecuadamente el programa para la implementación del proyecto de investigación se procede a la adquisición y etiquetado de imágenes, para lo cual se utilizara la herramienta Image Labeler incorporada en el software Matlab. Esta herramienta permite etiquetar los datos que sea desea dentro de

una imagen o una serie de imágenes, en la Figura 7 se puede observar la interface de esta herramienta.

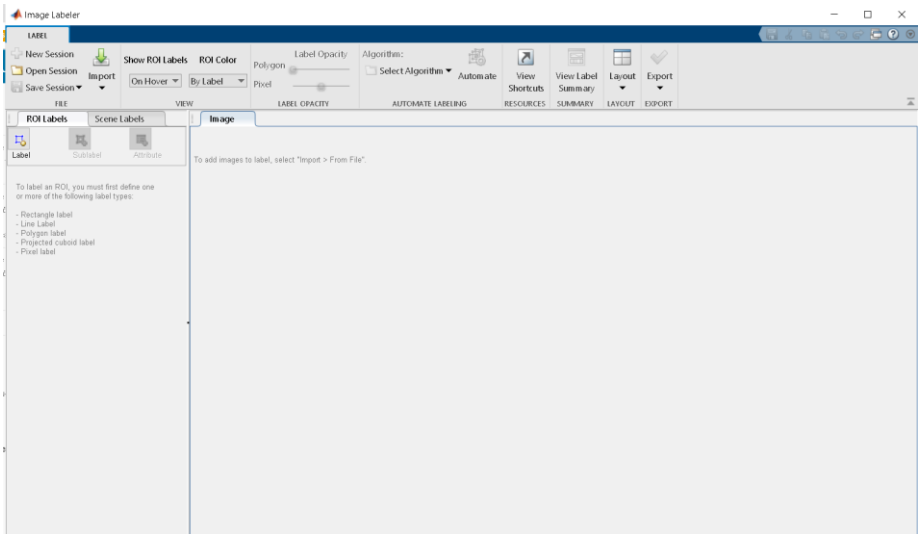


Figura 7. Interface de Image Labeler.

Fuente: Autoría Propia

Esta herramienta de etiquetado posee varias opciones para seleccionar los objetos de interés dentro de una imagen, dentro de las cuales se tiene:

ROI: esta etiqueta de regiones de interés puede ser rectangulares, polilínea, etiquetado en pixeles o polígonos. Este tipo de etiqueta sirve para seleccionar de una forma interactiva los datos reales que se desea detectar posteriormente con un sistema Deep Learning.

Subetiquetas ROI: este tipo de etiqueta está dentro de una ROI y sirve para proporcionar una mayor cantidad de detalle sobre la etiqueta principal, es decir, si se utiliza una etiqueta ROI para un vehículo, las sub etiquetas podrían ser las luces, la placa de matrícula, las ruedas, retrovisor, entre otros.

Atributos ROI: los atributos son datos o información adicional que se quiere otorgar a una etiqueta ROI o una sub etiqueta ROI.

La adquisición de imágenes se realiza a través del ordenador ASUS mediante la utilización de la cámara Web y un grabado de video. En estos videos se utilizan

secciones de video de 1 y 2 minutos, donde se ubican 3 y 6 peces, respectivamente. En la Figura 8 se puede observar la primera trama de un video con la presencia de 3 peces.

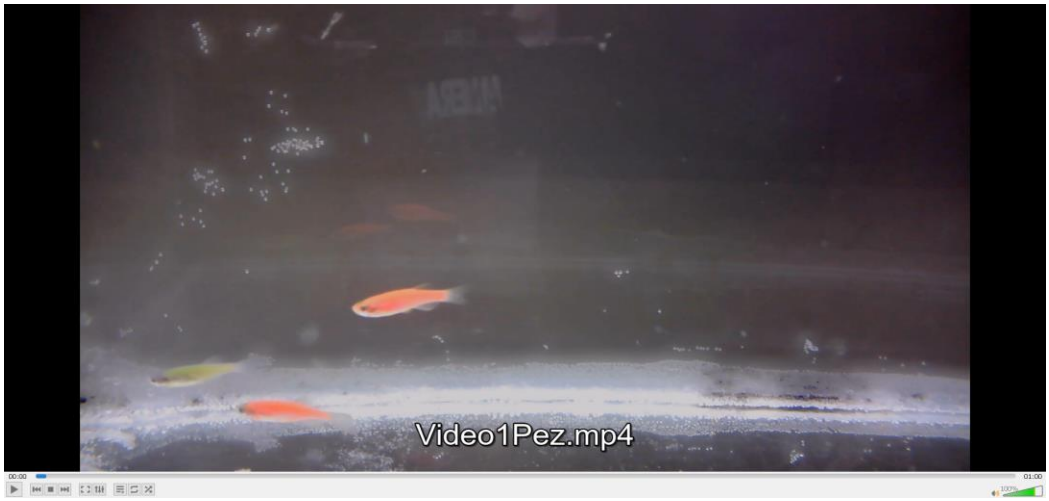


Figura 8. Trama del video capturado con la presencia de 3 peces.

Fuente: Autoría Propia

Por otro lado, en la Figura 9 se puede observar una trama del segundo video con la presencia de 6 peces. Estas imágenes son de vital importancia para el desarrollo de un dataset de entrenamiento para la red de detección de objetos con Deep Learning debido a que posee información real del entorno de detección. Se deben considerar la importancia de obtener las imágenes con este método, ya que los peces presentan comportamientos diferentes de frame a frame (captura a captura).

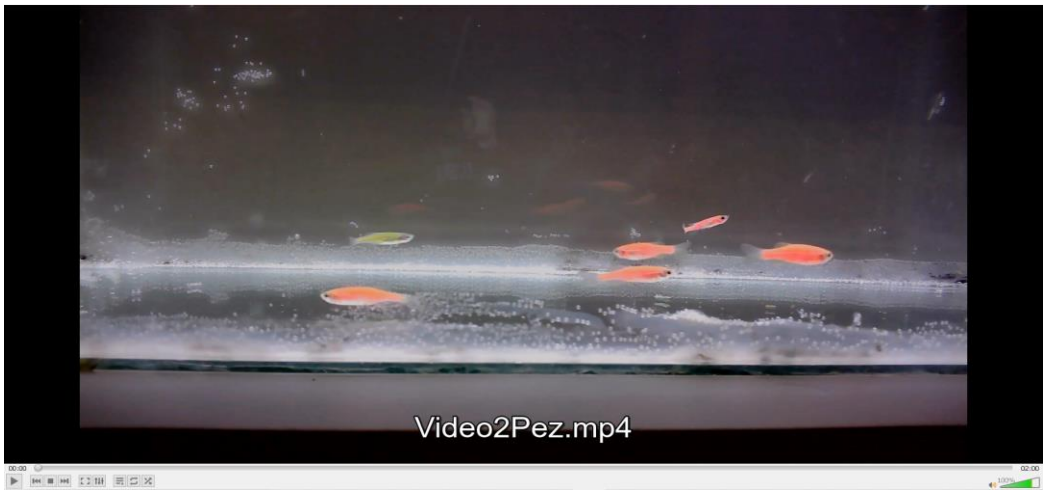


Figura 9. Trama del video con la presencia de 6 peces.

Fuente: Autoría Propia

El proceso de etiquetado se compone varios procesos, los mismos se describen en la Figura 10.

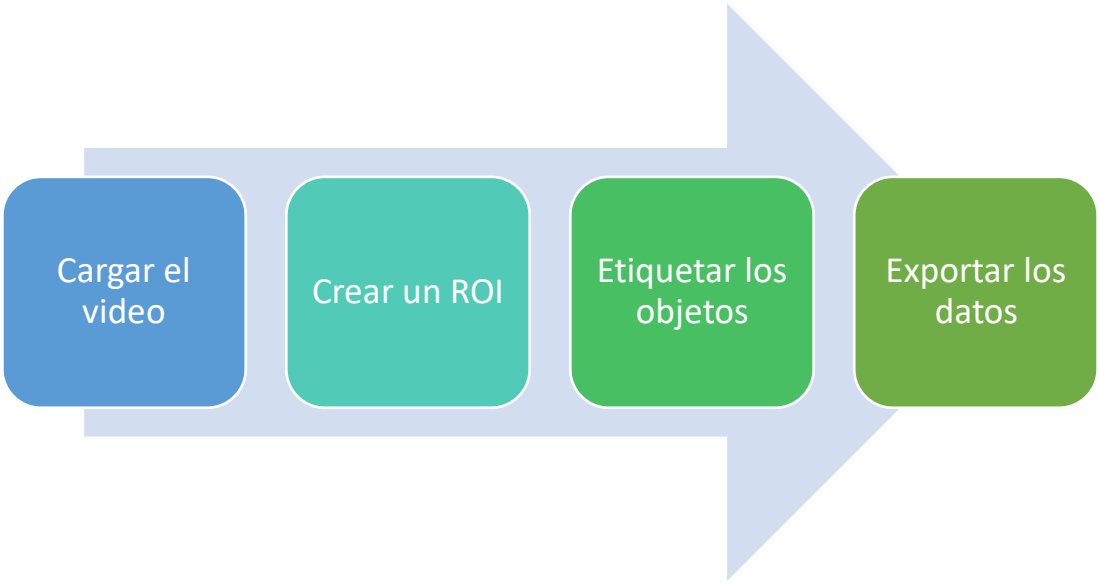


Figura 10. Proceso de etiquetado de imágenes.

Fuente: Autoría Propia

Carga del video

Una vez que se ha realizado la adquisición de los videos con 3 y 6 peces se procede a la carga de los archivos para lo cual se debe aplicar una separación en frames con la siguiente línea de código

```
ii = 1;

while hasFrame(shuttleVideo)
   img = readFrame(shuttleVideo);
   filename = [sprintf('%03d',ii) '.jpg'];
   fullname = fullfile(workingDir,'images',filename);
   imwrite(img,fullname)    % Write out to a JPEG file
(img1.jpg, img2.jpg, etc.)
   ii = ii+1;
end
```

Figura 11. Código de separación de video a frames.

Fuente: Autoría Propia

Las imágenes obtenidas son cargadas en el aplicativo de Image Labeler utilizando el comando Import. En la Figura 12 se puede observar el resultado del proceso de carga de imágenes.

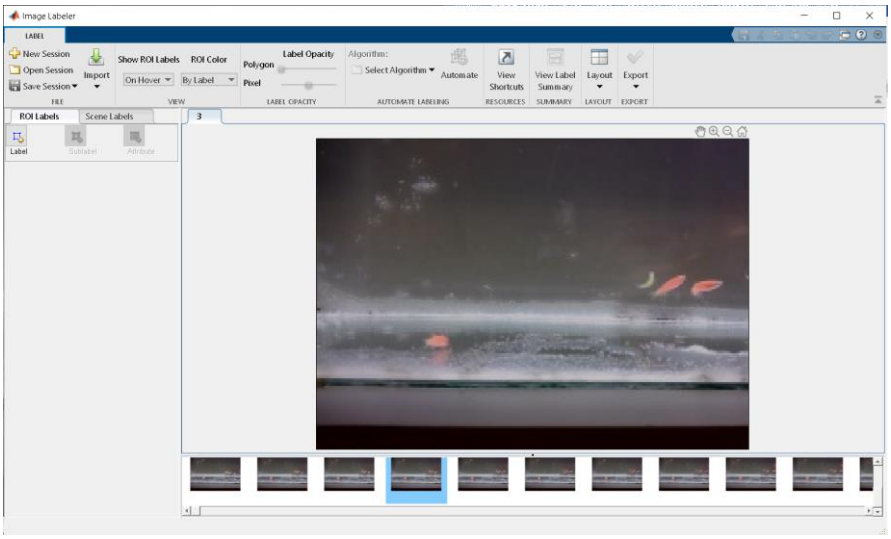


Figura 12. Imágenes cargadas exitosamente en Image Labeler.

Fuente: Autoría Propia

Crear un ROI

Para iniciar con el proceso de etiquetado manual de cada uno de los peces existentes en las imágenes se debe crear una nueva etiqueta ROI en el programa Image Labeler, para lo cual se debe ingresar en la opción de *Define New ROI label*. En la Figura 13 se puede observar la creación de una etiqueta denominada PEZ de forma rectangular y de color verde fosforescente.

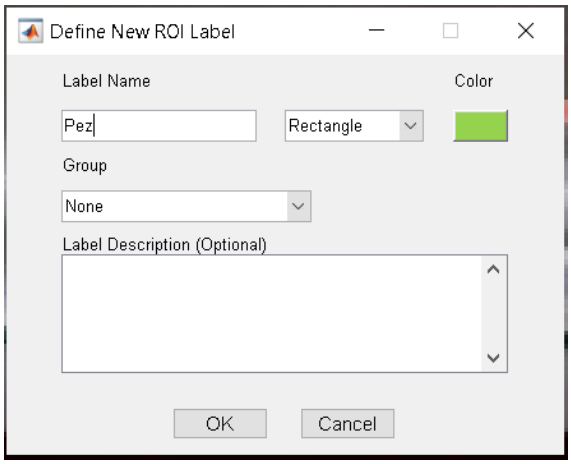


Figura 13. Creación de una etiqueta ROI.

Fuente: Autoría Propia

Esta etiqueta servirá para seleccionar todos los peces que aparezcan en las imágenes cargadas previamente y de esta manera el algoritmo de detección pueda entrenarse favorablemente, como se muestra en la Figura 14.

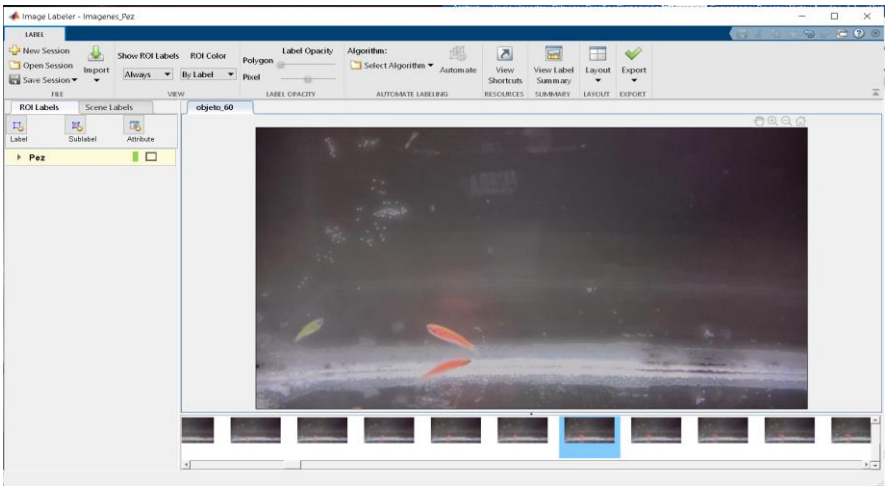


Figura 14. Etiquetas listas para selección de objetos en una imagen.

Fuente: Autoría Propia

Etiquetar los objetos

En el proceso de etiquetado de peces se utilizaron 393 imágenes con un promedio de aparición de 4.5 peces por imagen, dando un total de 1755 etiquetas generadas para el dataset de detección de objetos. En la Figura 15 se muestra el etiquetado de 3 peces.

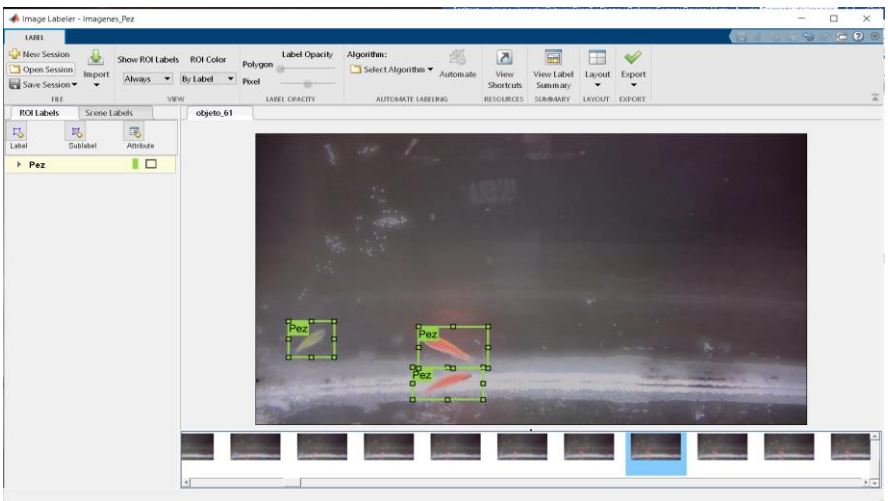


Figura 15. Etiquetado de peces en 390 imágenes.

Fuente: Autoría Propia

Exportar los datos

Una vez que se ha realizado exitosamente el etiquetado de todos los peces en las imágenes se procede a exportar el archivo en un formato groundTruth, para cual se utiliza la opción *Export to workspace*, como se muestra en la Figura 16

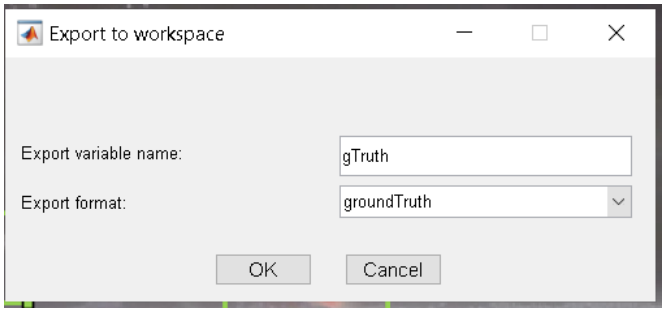


Figura 16. Datos exportados hacia el workspace.

Fuente: Autoría Propia

4.3 Entrenamiento de la red

Una red para detección de objetos dentro de una imagen utilizando una cámara web convencional puede utilizar diferentes algoritmos Deep Learning, dentro de los cuales se tienen:

- **Características *Haar*:** propuesto por Viola & Jones, utiliza características tipo *Haar* como funciones rectangulares simples en 2 dimensiones, donde se puede variar el tamaño y la posición de recuadros blancos y negros. Este algoritmo fue mejorado por Lienhart quien propuso una variación adicional de la imagen en algunos de 45 grados. Esto significa que se pueden encontrar 180 000 características en ventanas de 24x24 pixeles.
- **Patrones Binarios Locales (LBP):** propuesto por Zhang et al.; este algoritmo consiste en la comparación de la intensidad de los pixeles con sus respectivos vecinos para obtener algún patrón del tipo LBP. Estas características obtienen más información que las características tipo Haar debido a la redundancia de analizar los pixeles vecinos. Este tipo de entrenamiento tiene otra ventaja relacionada con el tiempo de entrenamiento, ya que es más corto que las características Haar.

- **Histograma de Gradientes Orientados (HOG):** propuesto por Dalal, este modelo utiliza una representación en histogramas de gradientes de orientación *(HOG)* de una imagen. Es muy utilizado para detección de peatones con modelos de aprendizaje. Los HOG son calculados en matrices NxN pixeles (Merchán et al., 2014).
- **ACF (Características Agregadas de Canal):** este algoritmo detecta objetos dentro de una imagen en función de un conjunto de imágenes de entrenamiento, previamente etiquetadas y de donde se puede obtener información de la ubicación de los objetos dentro de la imagen. (van Ranst et al., 2018)

En este proyecto de investigación se decidió utilizar una red avanzada para detección de objetos basada en Deep Learning, específicamente la denominada como ACF. El entrenamiento de la red se realizó aplicando el siguiente código:

```
load ('Etiquetas_Buenas.mat')
Pez = selectLabels(gTruth3,'Pez')
trainingData                                                    =
objectDetectorTrainingData(Pez,'SamplingFactor',2,
'WriteLocation','TrainingData');

detector = trainACFObjectDetector(trainingData,'NumStages',5);
```

Figura 17. Código de entrenamiento de la red.

Fuente: Autoría Propia

Las instrucciones utilizadas para el entrenamiento se especifican a continuación:

- **SelectLabels:** Mediante la utilización de esta instrucción se procede a cargar y obtener el dataset de 1755 etiquetas mostrado previamente, donde se encuentra la ubicación de los peces en cada una de las imágenes. En este caso se especifica entre comillas el nombre de la etiqueta con la que se entrenará a la red ACF con el nombre de “Pez”.
- **ObjectDetectorTrainingData:** Este comando devuelve una tabla de datos que será utilizada para el entrenamiento de la red con opciones adicionales especificadas de una en una, en este caso se especifica lo siguiente:

- SamplingFactor de 2, es decir, se realiza un submuestreo de imágenes donde se eliminan aquellas imágenes que tengan la misma secuencia o sean repetidas.
- WriteLocation: es la dirección de la ubicación de la carpeta donde se almacenarán las imágenes extraídas del dataset original, estas serán las imágenes reales utilizadas para el entrenamiento.
- TrainingData: este parámetro especifica que el tipo de entrenamiento será devuelto mediante una tabla.

- **TrainACFObjectDetector:** Este comando es el encargado de devolver un detector de objetos con las características de ACF entrenado. El funcionamiento de este entrenador consiste en la utilización de secciones positivas de los objetos en las imágenes proporcionadas en el TraningData y recopila automáticamente las secciones negativas de las imágenes durante el entrenamiento.
  - **NumStage:** es el número de etapas de entrenamiento para el proceso iterativo, en este caso se especifica un número de 5 etapas.

Una vez que se ejecuta el comando para el entrenamiento de la red se puede observar la carga del data set en la variable Pez con un número de imágenes igual 393, esta será la información de partida para la red ACF, cabe recalcar que cada imagen contiene su propio etiquetado de objetos.

```
Pez =

  groundTruth with properties:

          DataSource: [1×1 groundTruthDataSource]
    LabelDefinitions: [1×5 table]
           LabelData: [393×1 table]
```

Figura 18. Carga del dataset con 393 imágenes.

Fuente: Autoría Propia

Durante el proceso de entrenamiento el algoritmo ACF ha obtenido un total de 664 imágenes positivas donde se encuentran los peces y ha generado 3320 imágenes

negativas a partir del etiquetado manual realizado en Image Labeler. El tiempo de entrenamiento para este caso es de 58.1833 segundos. En la Figura 19 se puede observar los resultados del entrenamiento para 5 etapas.

```
Stage 5:
Sample negative examples(~100% Completed)
Found 2 new negative examples for training.
Compute aggregated channel features...Completed.
Train classifier with 664 positive examples and 3320 negative examples...Completed.
The trained classifier has 786 weak learners.
----------------------------------------
ACF object detector training is completed. Elapsed time is 58.1833 seconds.
```

Figura 19. Entrenamiento de una red ACF.

Fuente: Autoría Propia

Se realizaron diferentes pruebas de entrenamiento únicamente variando el número de etapas y el parámetro de SamplingFactor con el objetivo de determinar los tiempos de entrenamiento, los resultados se muestran en la Tabla 0.1 se muestran los tiempos de entrenamiento.

Tabla 0.1. Comparación de los tiempos de entrenamiento.

Tiempos de entrenamiento		
SamplingFactor	NumStage	Time (seg)
2	5	58,18
3	5	34,35
1	5	850,15
2	8	98.34
3	8	51.72
2	10	99.45
3	10	67.73

Fuente: Autoría Propia

El tiempo máximo de entrenamiento está relacionado con estos dos parámetros, es decir, a medida que se tiene un factor de muestreo (SamplingFactor) menor, el tiempo aumenta y cuando el número de etapas (NumStage) es mayor, el tiempo también se ve afectado. Esta información es muy importante debido a que el tiempo

de entrenamiento también se ve afectado por la capacidad de procesamiento que tiene el ordenador donde se está desarrollando el entrenamiento.

Para analizar los resultados obtenidos del tiempo de entrenamiento se decidió aplicar una relación matemática entre $SamplingFactor/NumStage$ de esta manera se puede identificar la variación del tiempo y su dependencia de estas variables. En la Figura 20 se puede observar el comportamiento del tiempo en función de la relación estipulada, donde se puede determinar que a medida que el SamplingFactor es mayor el tiempo de entrenamiento disminuye y cuando la relación tiende a cero el tiempo de entrenamiento incrementa.

Figura 20. Tiempo de entrenamiento.

Fuente: Autoría Propia

4.4 Raspberry Pi y Matlab

Una vez que se ha entrenado la red ACF satisfactoriamente se requiere de una implementación de la Raspberry Pi 3B para proceder a la adquisición de imágenes en tiempo real accediendo a los periféricos USB del mini ordenador y de esta manera obtener las imágenes de la cámara Web ubicada en la parte frontal de la pecera. Matlab cuenta con otra herramienta tecnología denominada *Hardware Setup*, esta opción de Matlab permite agregar controladores o miniordenadores de forma remota mediante comunicación WiFi, de esta manera se puede acceder a estos dispositivos desde codificación. Cabe recalcar que los pasos descritos a continuación son necesarios para proceder a una implementación de un sistema embebido con el algoritmo de detección de objetos basado en Deep Learning.

El primer paso para la configuración de una nueva tarjeta en Matlab inicia con la selección de la tarjeta, es muy importante saber exactamente la versión de la tarjeta que se va a utilizar porque de esta selección depende el sistema operativo que será instalado posteriormente. En este caso se posee una tarjeta Raspberry Pi 3B y se selecciona como se muestra en la Figura 21.

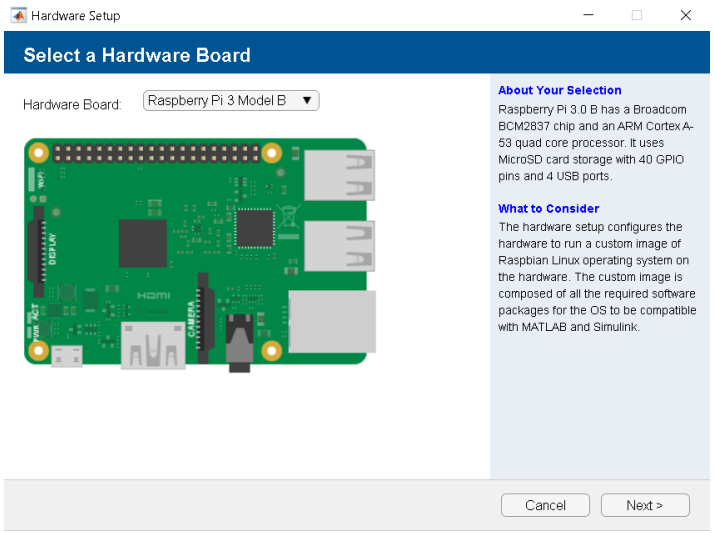


Figura 21. Interface de selección de tarjeta Raspberry.

Fuente: Autoría Propia

La segunda interface corresponde a la selección del sistema operativo que se desea instalar en el mini ordenador Raspberry Pi 3B, en este caso se ha seleccionado la opción de configurar el miniordenador con el sistema operativo MathWorks Raspberry. Este sistema operativo contiene dos sub sistemas basados en Raspbian Buster Lite y una serie de paquetes con librerías que permiten desarrollar aplicaciones en Matlab y Simulink.

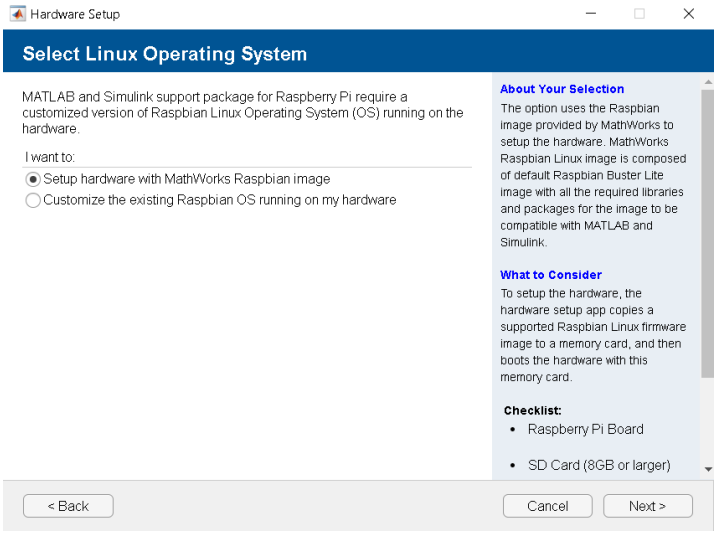


Figura 22. Interface de selección del sistema operativo.

Fuente: Autoría Propia

La tercera interface del proceso de configuración de tarjetas corresponde a la selección de la imagen de instalación con el sistema operativo deseado. En este caso se debe seleccionar la imagen de MathWorks Rasbian preparada para aplicaciones de Deep Learning, esta opción permite implementar algoritmos de detección de objetos para una ejecución en tiempo real en un sistema embebido. En la Figura 23 se puede observar los links de descarga para el sistema operativo.

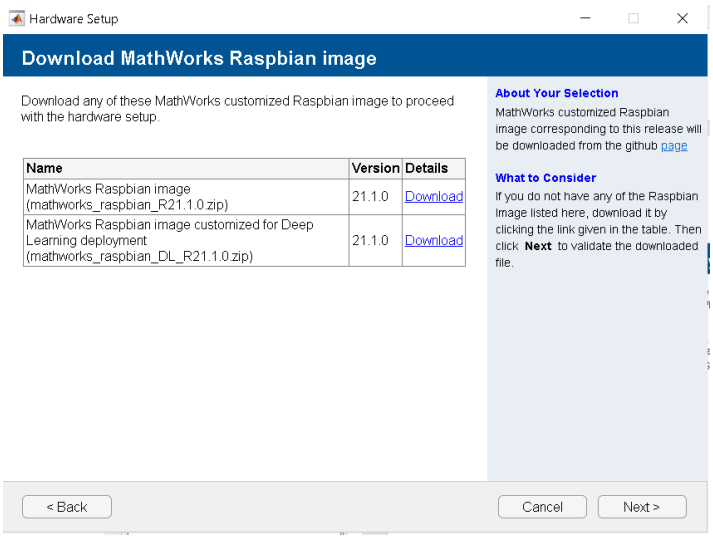


Figura 23. Descarga del sistema operativo con Deep Learning.

Fuente: Autoría Propia

Una vez descargado los archivos comprimidos se procede a realizar una validación de los documentos, en la Figura 24 se puede observar los resultados de este proceso.

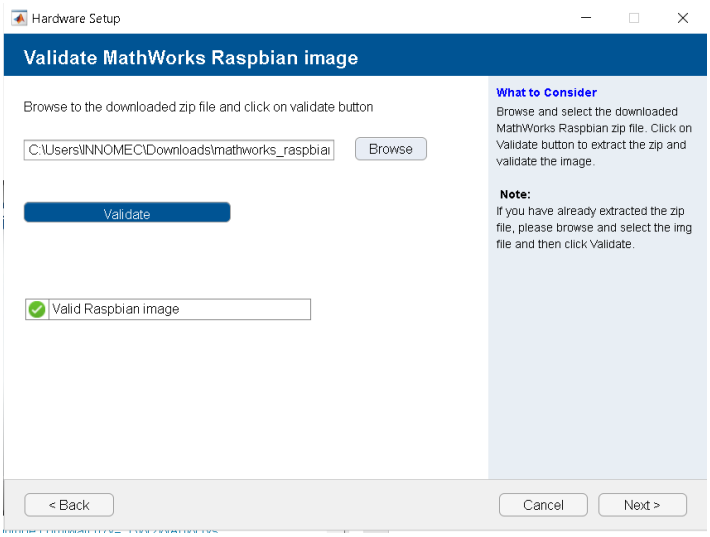


Figura 24. Validación del sistema operativo.

Fuente: Autoría Propia

En la quinta interface del proceso de configuración se configura el tipo de conectividad que se utilizará, este puede ser LAN, WiFi o cable Ethernet. En este caso se optó por la configuración de una red WiFi, como se muestra en la Figura 25.

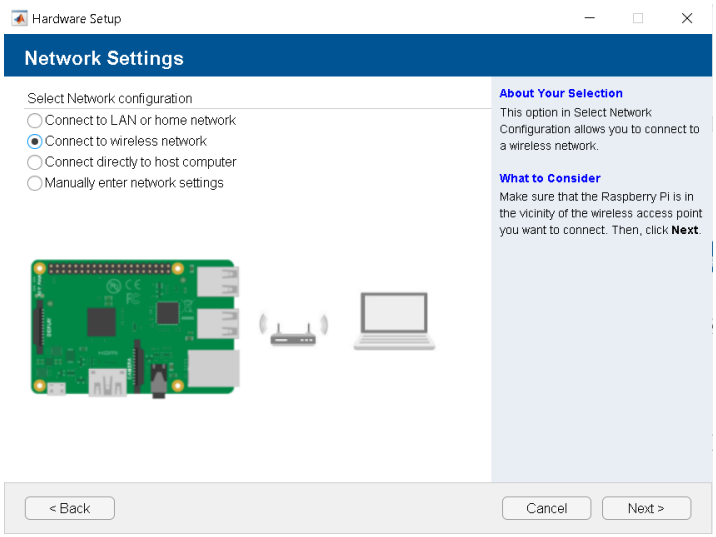


Figura 25. Configuración de una red WiFi.

Fuente: Autoría Propia

Para definir la configuración de la red WiFi que utilizará el mini ordenador Raspberry para conectarse a una red se debe ingresar el nombre de la red y la clave. En la Figura 26 se puede observar esta configuración.

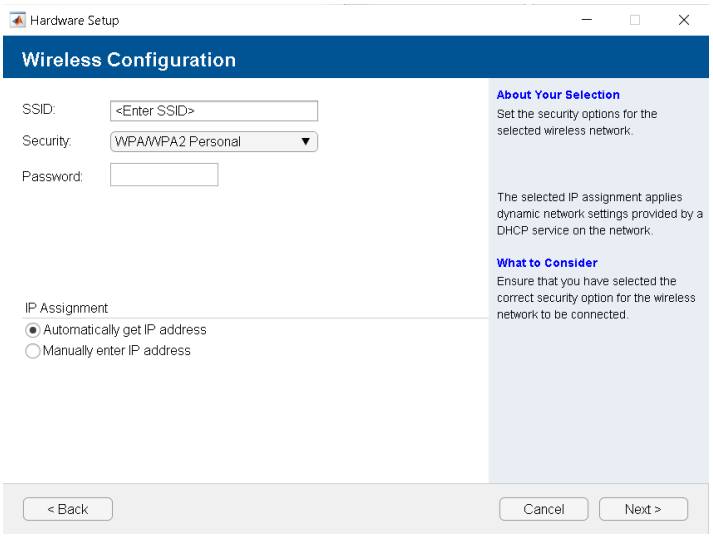


Figura 26. Ingreso de Usuario y Clave de la red WiFi.

Fuente: Autoría Propia

En la séptima interface se procede a seleccionar una memoria micro SD para realizar la grabación del sistema operativo conjuntamente con los siguientes paquetes:

- MathWorks Rasbian
- Paquetes y librerías Mathlab
- Paquetes Deep Learning
- Conexión WiFi

En la Figura 27 se puede observar la selección de la tarjeta microSD y la grabación del sistema operativo configurado.

Figura 27. Selección de memoria Micro SD.

Fuente: Autoría Propia

Finalmente se graba la información en la tarjeta y procede a la implementación en la Raspberry Pi 3B, este último paso se muestra en la Figura 28.

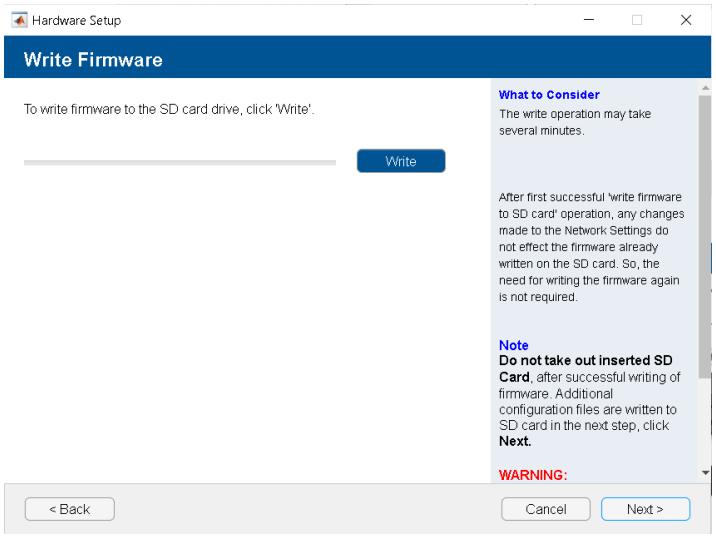


Figura 28. Grabación del sistema operativo en la microSD.

Fuente: Autoría Propia

4.4.1 Instalación del sistema operativo

La verificación de una correcta instalación del sistema operativo puede ser posible únicamente encendiendo el miniordenador Raspberry Pi 3B y visualizando a través de un monitor conectado al puerto HDMI o mediante un servidor VCN (Virtual Network Computing). En la Figura 29 se puede observar el correcto encendido del miniordenador, adicionalmente se tiene una conexión exitosa a la red de internet. En

este caso la tarjeta posee una dirección IP de 192.168.1.71, la misma que servirá para conectarse con el software Matlab y de esta manera ejecutar los algoritmos desarrollados.

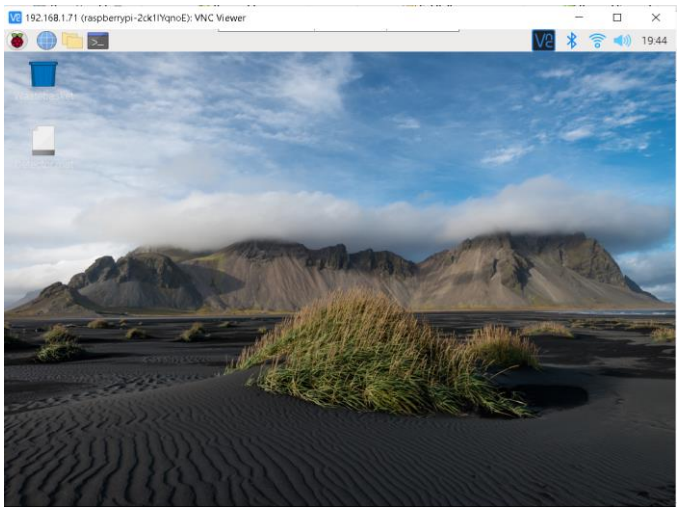


Figura 29. Encendido de la Raspberry Pi 3B.

Fuente: Autoría Propia

La verificación de la detección de la tarjeta Raspberry Pi3B instalada utilizando el software Matlab se realiza con el código:

```
raspiObj = raspi()
```

Figura 30. Código para enlazar las tarjeta Raspberry.

Fuente: Autoría Propia

El resultado de esta operación se puede observar en la Figura 31 donde se puede validar la dirección IP de 192.168.1.71. Adicionalmente en esta conexión se puede observar lo siguiente:

- Dirección IP
- Nombre de la tarjeta
- Número de leds habilitados

- Pines habilitados para conexiones de elementos electrónicos
- Canales de comunicación
- Puertos de comunicación I2C
- Cámaras Web conectadas a la tarjeta
- Velocidad del puerto I2C

La Figura 31 contiene la información del miniordenador Raspberry conectado desde la computadora a través de WiFi.

```
raspiObj =

  raspi with properties:

         DeviceAddress: '192.168.1.71'
                  Port: 18734
             BoardName: 'Raspberry Pi 3 Model B+'
         AvailableLEDs: {'led0'}
  AvailableDigitalPins: [4,5,6,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27]
  AvailableSPIChannels: {'CE0','CE1'}
     AvailableI2CBuses: {'i2c-0','i2c-1'}
      AvailableWebcams: {'USB 2.0 Camera: USB Camera (usb-3f980000.usb-1.1.3)'}
           I2CBusSpeed: 100000

  Supported peripherals
```

Figura 31. Conexión entre Matlab y Raspberry.

Fuente: Autoría Propia

4.5 Controlador Fuzzy

El controlador fuzzy desarrollado en esta sección es el encargado de tomar las decisiones de alimentar o no alimentar a los peces cuando su comportamiento lo amerite. La información que se obtienen del algoritmo de detección de peces debe ser procesada para ingresar al controlador, para lo cual se debe realizar el siguiente proceso:

- Identificar las zonas de alimentación, normal y satisfecho de los peces.
- Determinar el número de peces detectado por imagen.
- Identificar la posición de los peces dentro de las tres zonas.
- Cuantificar la presencia del cardumen mediante el cálculo de la frecuencia estadística.
- Determinar la frecuencia acumulada durante todas las imágenes.

4.5.1 Zonas de alimentación

Una vez que se ha logrado identificar los peces con un error menor igual al 25% se procede a implementar las zonas requeridas para determinar el comportamiento de los peces. Durante la revisión bibliográfica y una investigación de campo se pudo identificar que los peces tienen tres comportamientos básicos en cardumen.

- Cuando los peces tienen hambre se dirigen al fondo de la pecera, con el objetivo capturar los residuos de alimento o buscar entre las piedras.
- Cuando los peces son alimentados se dirigen hacia la superficie del agua, con el objetivo de ser los primeros en alimentarse.
- Cuando los peces se encuentran dispersos o nadando por toda la pecera significa que se encuentran satisfechos y no deben ser alimentados.

En su mayoría esta información fue obtenida a base de entrevistas informales y revisión bibliográfica de proyectos relacionados. En la Figura 32 se puede observar una de las entrevistas realizadas a personal capacitado en la crianza y cuidado de peces.

Figura 32. Investigación de campo en peceras.

Fuente: Autoría Propia

Las tres zonas de etiquetado en las imágenes obtenidas a través de la comunicación establecida entre la computadora y la Raspberry se realizan a partir de la resolución obtenida. Utilizando el comando:

```
cam = webcam(raspiObj,1);
I = snapshot(cam);
```

Figura 33. Código para activar la cámara web.

Fuente: Autoría Propia

Con esto se puede obtener la información sobre el tamaño en pixeles de las imágenes que se han procesado en el algoritmo de detección de objetos, en este caso los resultados se muestran en la Figura 34. donde se aprecia un tamaño de 320 x 240 pixeles.

```
wcam =

  webcam with properties:

                    Name: '/dev/video0'
              Resolution: '320x240'
    AvailableResolutions: {'320x240'  '640x480'}
```

Figura 34. Características de la webcam y resolución de imágenes.

Fuente: Autoría Propia

La sección de color verde de la pecera corresponde a la zona satisfecha, donde los peces estarán cuando se realice un proceso de alimentación. Esta zona tiene un tamaño de 320 x 100 pixeles, con un punto de origen (0,0) pixeles. Por otro lado, la sección inferior de la pecera de color rojo corresponde a la zona de alimentación donde los peces estarán ubicados con mayor frecuencia cuando sientan hambre. Esta segunda zona tiene un tamaño de 320 x 100 pixeles y un punto de inicio igual a (0,140) pixeles.

Finalmente, la sección generada entre estas dos zonas será la denominada zona normal, con un tamaño de 320 x 40 pixeles con un punto de origen (0,100) pixeles. La identificación de estas tres zonas en la imagen se muestra en la Figura 35.

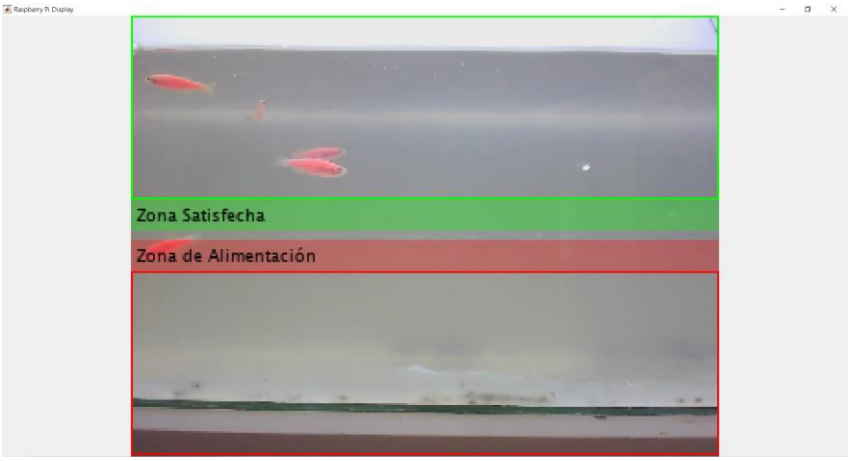


Figura 35. Identificación de las zonas en la pecera.

Fuente: Autoría Propia

4.5.2 Cálculos de presencia y repetibilidad

Una vez que los peces han sido identificados favorablemente y ahora el algoritmo sabe done están ubicados cada uno de ellos, se debe cuantificar este valor para ingresar al controlador fuzzy mediante una cantidad numérica. Para cual se decidió aplicar las siguientes fórmulas:

$$Total_{Pez} = \#S + \#N + \#H \quad (1)$$

Donde:

$Total_{Pez}$: corresponde al número de peces detectados por cada imagen

$\#S$: Número de peces detectados en la zona satisfecha

$\#N$: Número de peces detectados en la zona normal

$\#H$: Número de peces detectados en la zona alimentación

La repetibilidad o el número de peces que se repiten dentro de una misma zona se calcula aplicando la fórmula de la frecuencia estadística y se expresa en porcentajes, cabe recalcar que este valor es calculado por imagen.

$$fr_S = \frac{\#S}{Total_{Pez}} * 100\% \quad (2)$$

$$fr_N = \frac{\#N}{Total_{Pez}} * 100\% \quad (3)$$

$$fr_H = \frac{\#H}{Total_{Pez}} * 100\% \quad (4)$$

Donde:

fr_S: frecuencia satisfecha

fr_N: frecuencia normal

fr_H: frecuencia hambrienta

Por otro lado, es muy importante la determinación de las frecuencias acumulativas desde la ejecución del algoritmo, para lo cual se utilizaron las siguientes fórmulas:

$$Fr_S = \sum_{frame=1}^{\infty} fr_S(frame) \quad (5)$$

$$Fr_N = \sum_{frame=1}^{\infty} fr_N(frame) \quad (6)$$

$$Fr_H = \sum_{frame=1}^{\infty} fr_H(frame) \quad (7)$$

Donde:

Fr_S: frecuencia acumulada satisfecha

Fr_N: frecuencia acumulada normal

Fr_H: frecuencia acumulada hambrienta

Finalmente, se realiza una relación entre las tres variables para obtener un valor promedio de presencia de todos los peces durante todos los frames, con la siguiente fórmula:

$$Fuzzy_{SN} = \frac{Fr_S - FR_N}{\sum frames} \quad (8)$$

$$Fuzzy_{SH} = \frac{Fr_S - FR_H}{\sum frames} \quad (9)$$

$$Fuzzy_{NH} = \frac{Fr_N - FR_H}{\sum frames} \quad (10)$$

Donde:

$Fuzzy_{SN}$: Relación entre la frecuencia acumulada satisfecho y normal

$Fuzzy_{SH}$: Relación entre la frecuencia acumulada satisfecho y hambriento

$Fuzzy_{NH}$: Relación entre la frecuencia acumulada normal y hambriento

4.5.3 Desarrollo del controlador Fuzzy

El controlador fuzzy desarrollado será el encargado de tomar la decisión de accionar o no el sistema de alimentación a través de un microservo conectado a una mini tolva con alimento natural. Utilizando el toolbox fuzzy de Matlab se procede a declarar 3 variables de entrada y una variable de salida, como se muestra en la Figura 36.

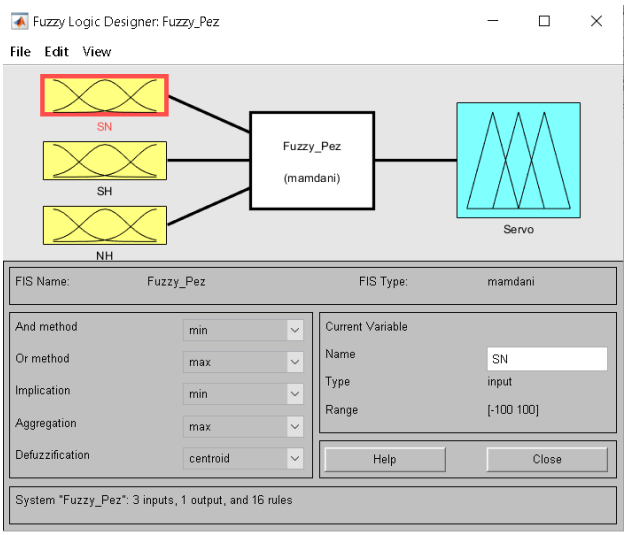


Figura 36. Controlador fuzzy de alimentación.

Fuente: Autoría Propia

Las entradas del controlador están basadas en las fórmulas (8), (9) y (10), donde se realiza una relación entre la frecuencia acumulada de los peces en las zonas satisfecha, normal y de alimentación. En este caso las variables SN, SH y NH, corresponden a los valores de FuzzySN, FuzzySH y Fuzzy NH, respectivamente.

Cada una de las entradas está configurada con una figura geométrica del tipo triángulo, las etiquetas para cada una de las variables son de un valor negativo, uno positivo y un valor intermedio, por ejemplo, en la función SN las etiquetas son SN(negativa), SN y SN(positiva), como se muestra en la figura 4.32.

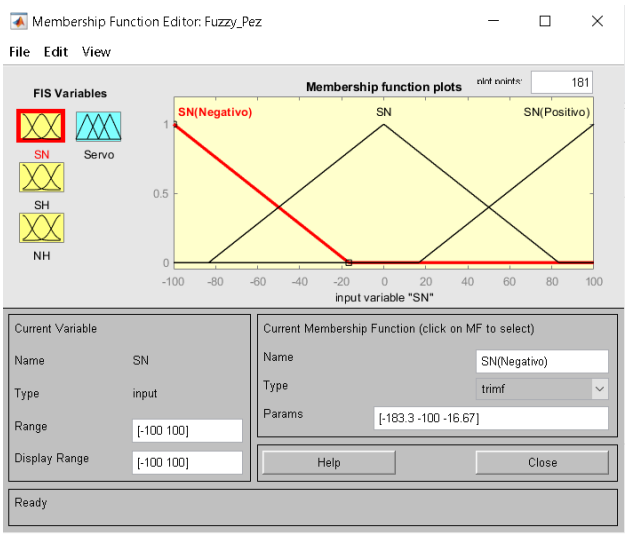


Figura 37. Configuración de las entradas del controlador fuzzy.

Fuente: Autoría Propia

Las salidas del controlador fuzzy están relacionadas con la cantidad de alimento que se debe suministrar a los peces. En este caso las etiquetas utilizadas para este caso fueron las siguientes: No_alimentar, Esperar y Alimentar. El rango de la salida es de 0 a 100, como se muestra en la Figura 38

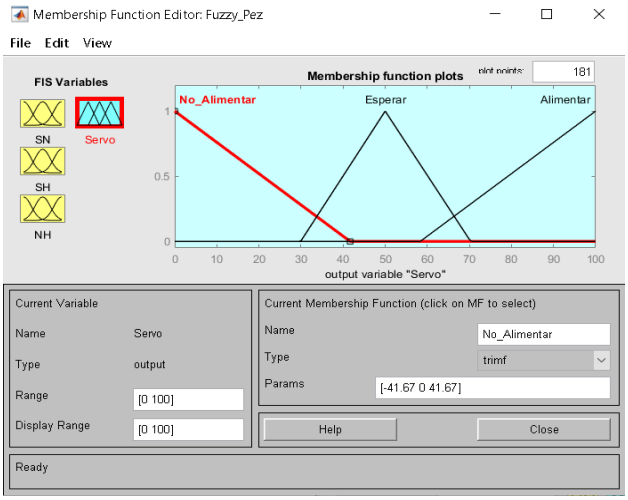


Figura 38. Configuración de las salidas del controlador fuzzy.

Fuente: Autoría Propia

Las reglas utilizadas para el controlador fuzzy y la toma de decisiones está basada en la revisión bibliográfica y las entrevistas realizadas al personal capacitado. En la Figura 39 se pueden observar las 16 reglas para el funcionamiento adecuado del controlador.

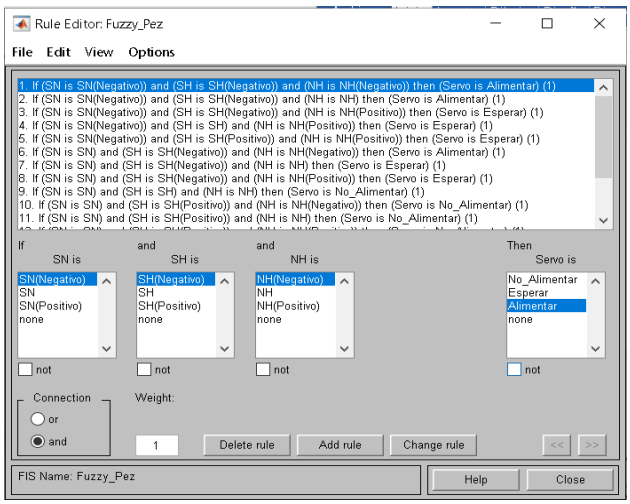


Figura 39. Reglas del controlador fuzzy.

Fuente: Autoría Propia

La validación de las reglas establecidas en el controlador fuzzy se puede realizar mediante el aplicativo de visualización, en este caso para un valor de cero SN = 0, SH = 0, NH = 0 se obtiene una salida de Servo = 13. Este valor no será capaz de activar el sistema de alimentación puesto que al tener un valor nulo en todas las

entradas quiere decir que físicamente los peces se encuentran bien y están distribuidos uniformemente sobre todas las zonas de la pecera. En la Figura 40 se puede observar las reglas y su variación de acuerdo a las entradas.

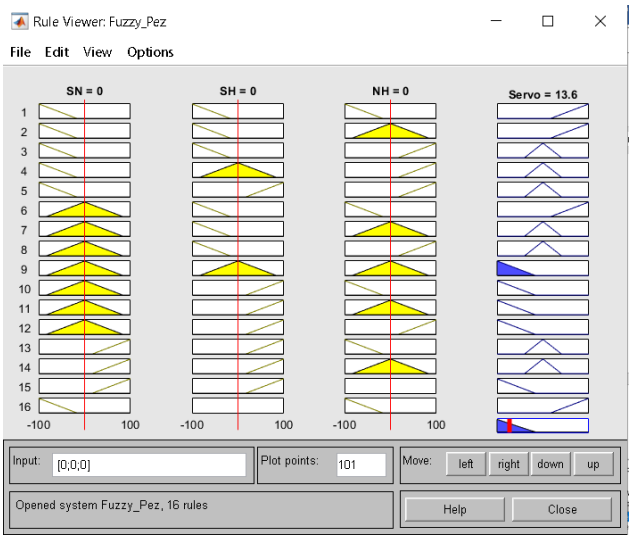


Figura 40. Visualizador de reglas fuzzy.

Fuente: Autoría Propia

Finalmente, otra forma de visualización y validación del controlador fuzzy desarrollado, es a través de un diagrama de superficies, como se muestra en la figura 4.36.

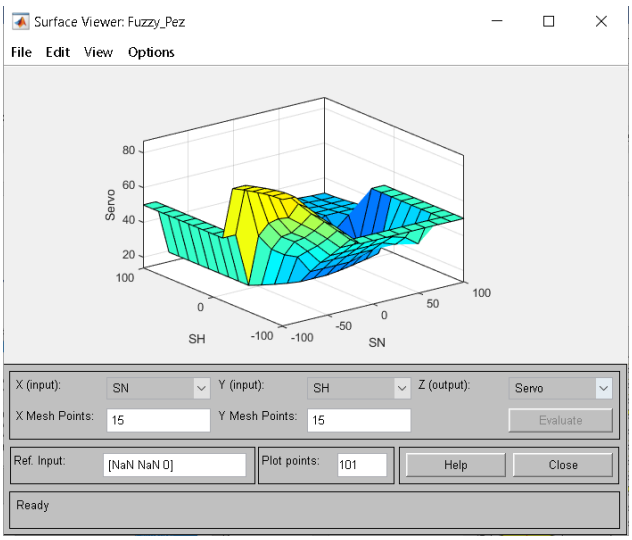


Figura 41. Superficie del controlador fuzzy.

Fuente: Autoría Propia

4.3.4 Implementación física

El sistema de control desarrollado consta de dos etapas: la primera es la detección de objetos en una imagen utilizando Deep Learning, en este caso el algoritmo desarrollado es el encargado de identificar y cuantificar la presencia de los peces en las tres zonas etiquetadas previamente para cual se utilizan los siguientes comandos:

```
detector =
acfObjectDetector(x.detector_estructura.Classifier,x.detector_est
ructura.TrainingOptions)
[bboxes, scores] = detect(detector,I);
```

Figura 42. Código para realizar la detección de objetos.

Fuente: Autoría Propia

Donde:

Detector: Es la estructura de la red Deep Learning desarrollada

Detect: Utiliza la red entrada y la imagen I capturada en tiempo real

Bboxes: es la ubicación en un recuadro de pixeles del objeto encontrado.

Scores: es el valor adimensional que cuantifica la similut del objeto real con el entrenado, este valor es calculado por frame.

Por otro lado la activación del mecanismo de alimentación es realizado por un controlador fuzzy configurado para analizar el comportamiento de los peces y decidir su alimentación. Este proceso se realiza utilizando los siguientes comandos

```
fis = readfis('Fuzzy_Pez');
alimentar = evalfis(fis,[Fuzzy_S_N Fuzzy_S_H Fuzzy_N_H])
```

Figura 43. Código para calcular el controlador fuzzy.

Fuente: Autoría Propia

Donde:

Fis: es el controlador fuzzy desarrollado.

Alimentar: es la variable de salida obtenida después de procesas el controlador fuzzy.

El sistema de control completo para este proyecto de investigación se muestra en la Figura 44, donde se especifican las etapas y procedimiento realizado.

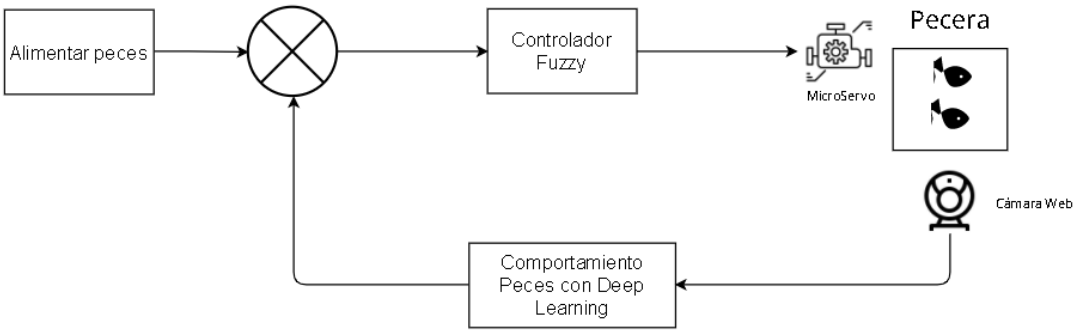


Figura 44. Sistema de control completo.

Fuente: Autoría Propia

En la Figura 45 se puede observar la conexión del mini ordenador Raspberry con la cámara web a través de un puerto USB, también se puede identificar las conexiones de VCC, GND y PWM requeridas por el servo motor para activar el mecanismo de alimentación. Cabe recalcar que esta tarjeta requiere de un ventilador para evitar el calentamiento por el procesamiento de la información e imágenes.

Figura 45. Mini ordenador Raspberry.

Fuente: Autoría Propia

Por otro lado, los elementos de iluminación requerido por el sistema de visión artificial constan de 9 leds de alta intensidad a 12 Vdc, ubicados en la parte superior de la pecera. Esta iluminación extra permitirá visualizar de mejor manera el comportamiento de los peces, como se muestra en la Figura 46.

Figura 46. Sistema de iluminación led.
Fuente: Autoría Propia

El mecanismo de alimentación fue realizado en impresión 3D con el material PLA un polímero poliláctico biodegradable de alta disponibilidad en el mercado local. El accionamiento del mecanismo es controlado por un micro servo colocado en la parte inferior de la carcasa, como se muestra en la Figura 47.

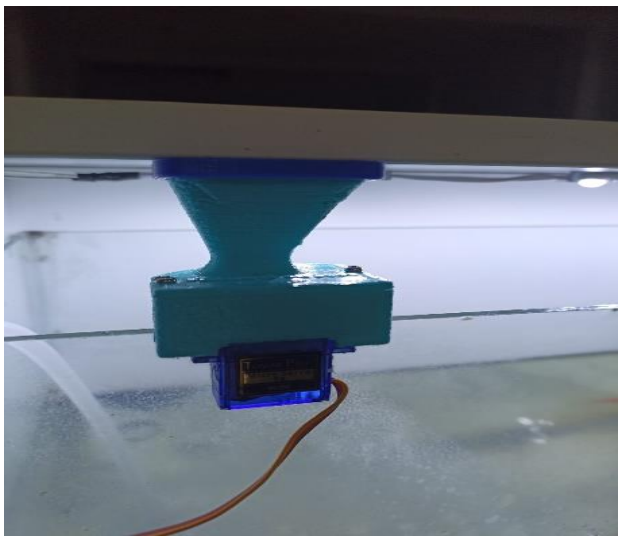


Figura 47. Sistema de alimentación automático.

Fuente: Autoría Propia

Finalmente, en la Figura 48 se puede observar la implementación real del sistema embebido para alimentar a los peces cebra dentro de una pecera con iluminación controlada por led.

Figura 48. Sistema implementado físicamente.

Fuente: Autoría Propia

4.6 Análisis de resultados

Los resultados obtenidos de la presente investigación se presentarán de forma progresiva iniciando por el error de detección, cálculo de la frecuencia acumulada de los peces en las zonas etiquetas en la pecera y por último el funcionamiento del controlador fuzzy desarrollado. En la Figura 49 se puede observar este proceso de resultados obtenidos.

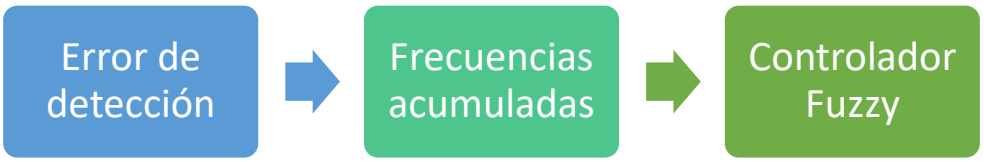


Figura 49. Presentación de resultados.

Fuente: Autoría Propia

4.6.1 Error de detección

A continuación, se describen los resultados obtenidos para un caso donde los peces están siendo alimentados por el operario. En Figura 50 se puede observar el comportamiento real de los peces, el cardumen se encuentra concentrado en la parte superior de la pecera, es decir, en la zona satisfecha porque efectivamente están siendo alimentados.

Figura 50. Comportamiento real de los peces.

Fuente: Autoría Propia

El primer resultado que se obtiene después de ejecutar el algoritmo de detección de peces, es el mostrado en la Figura 50 donde claramente se puede observar los peces encontrados delimitados por un cuadrado de color amarillo. En la Figura 51 se muestra el error relativo de detección para un total de 240 frames obtenidos cada 5 segundos durante un total de 20 minutos. En esta ejecución se obtuvo un mínimo error de 29.5% y máximo de 65%, cabe recalcar que este error es acumulativo y en varias ocasiones el algoritmo detecta 2 o 3 peces en un mismo cuadro delimitador y contabiliza dicho dato como 1 pez encontrado.

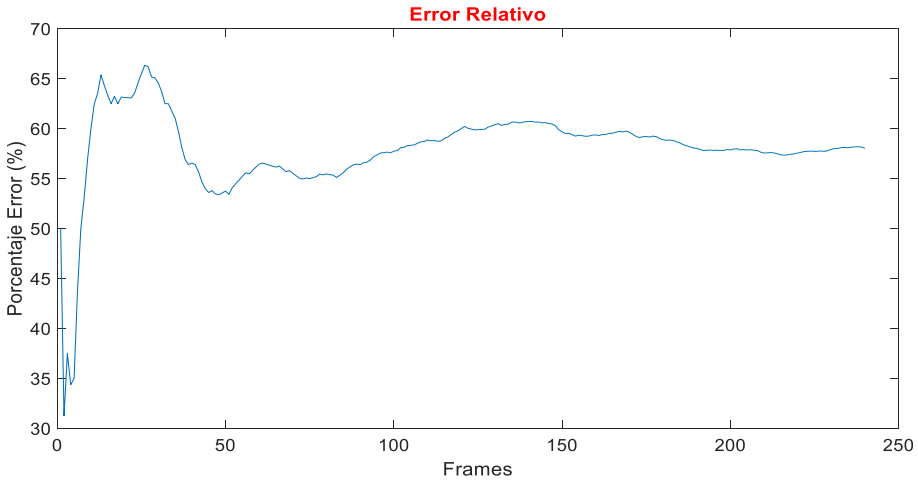


Figura 51. Error relativo de detección.

Fuente: Autoría Propia

4.6.2 Frecuencias acumuladas

Las frecuencias actuales son obtenidas directamente de cada imagen procesada por el algoritmo de detección, pero la frecuencia acumulada corresponde a un valor de presencia de los peces en una determinada zona. En este caso se trata de un análisis de un proceso de alimentación manual por parte del operario hacia los peces. En la Figura 52 se puede observar que existe un alto número de frames en los cuales la mayoría de peces se encuentran en la zona satisfecha y tiene relación con lo presentado en la Figura 50 donde se puede observar a los peces en la superficie del agua de la pecera.

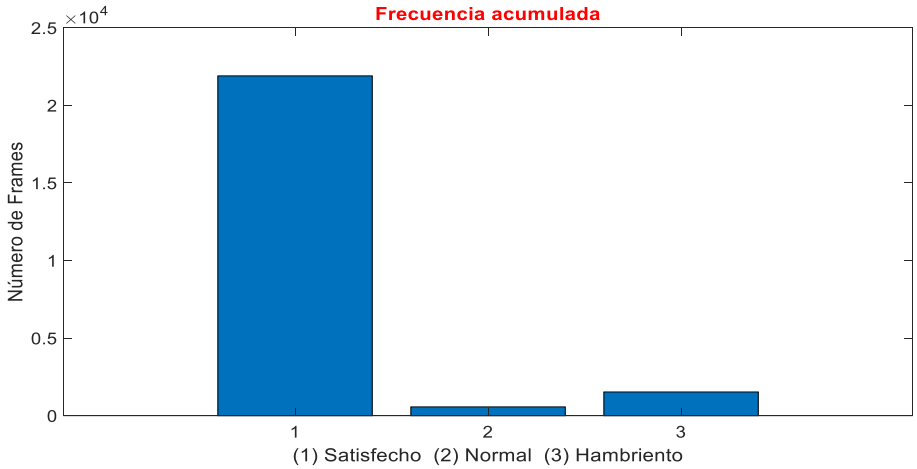


Figura 52. Frecuencia acumulada del cardumen.

Fuente: Autoría Propia

Esto quiere decir que el algoritmo detección y las fórmulas aplicadas pueden identificar el comportamiento de los peces con un alto grado de exactitud.

4.4.3 Resultados controladores fuzzy

Aplicando las fórmulas para el cálculo de los índices FuzzySN, FuzzySH y FuzzyNH se obtiene los resultados mostrados en la Figura 53, donde se puede evidenciar un alto grado de presencia del sentimiento satisfecho por parte del cardumen de peces y una mínima presencia del sentimiento hambriento.

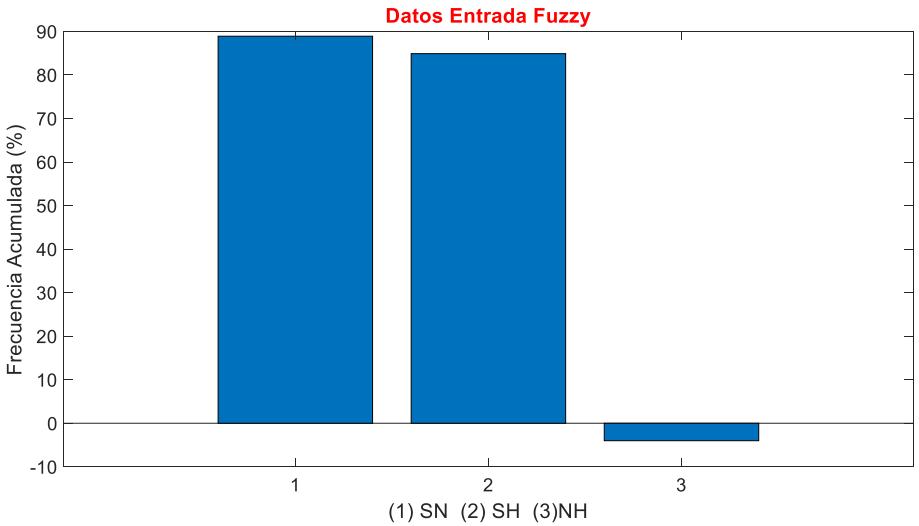


Figura 53. Índices de entrada fuzzy.

Fuente: Autoría Propia

Por otro lado, el controlador fuzzy procesa estos datos y genera un valor promedio de 31.5 en un rango de 0 a 100, cabe recalcar que las reglas establecidas en el controlador accionan el sistema de alimentación cuando la salida es superior a 60 y en caso de continuar con el sentimiento hambriento se realiza una doble alimentación cuando el valor de salida del controlador fuzzy supera los 80 puntos. En la Figura 54 se puede observar el comportamiento del controlador para este caso en particular, es decir, el controlador considera que no es necesario alimentar a los peces porque el sentimiento en todo el cardumen durante todo el tiempo de análisis es satisfecho.

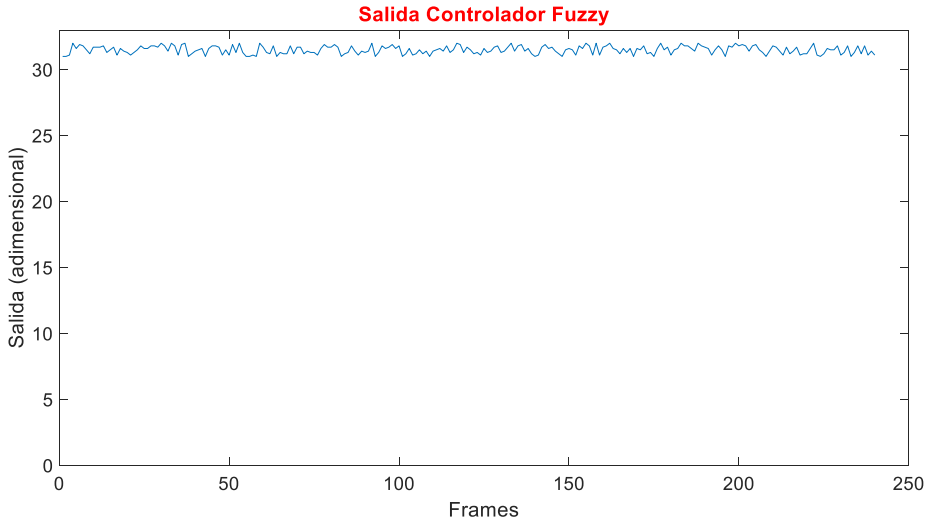


Figura 54. Salida del controlador fuzzy.

Fuente: Autoría Propia

4.6.3 Pruebas pez cebra y pez molly

Se ha realizado una prueba del algoritmo desarrollado utilizando 18 peces cebra y 5 peses de la raza molly de color negro, como se muestra en la Figura 55. Cabe recalcar que en este caso se utilizó el índice de detección es del 90%, es decir, el algoritmo buscara los peces que tengan ese porcentaje de similitud con el dataset utilizado para el entrenamiento de la red ACF.

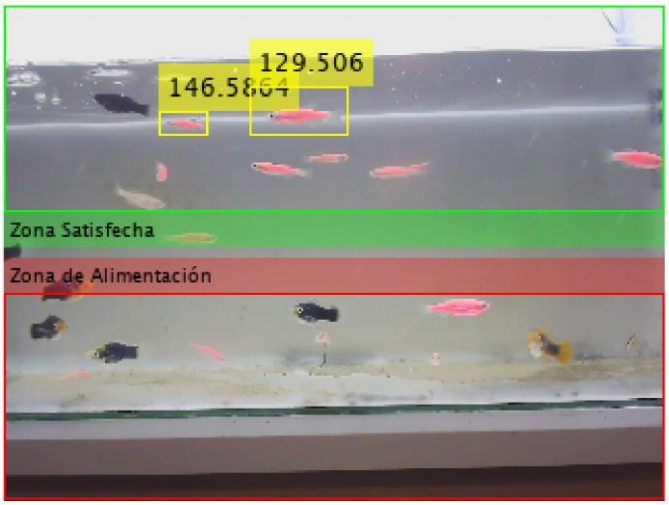


Figura 55. Peces cebra y peces molly.

Fuente: Autoría Propia

Los resultados obtenidos arrojaron un error máximo de 87% y un error mínimo de 82% entre los peces detectados por el algoritmo y los peces reales. Sin embargo, se puede interpretar estos datos como correctos debido a que el porcentaje de similitud es muy elevado, incluso algunos peces cebras no son detectados.

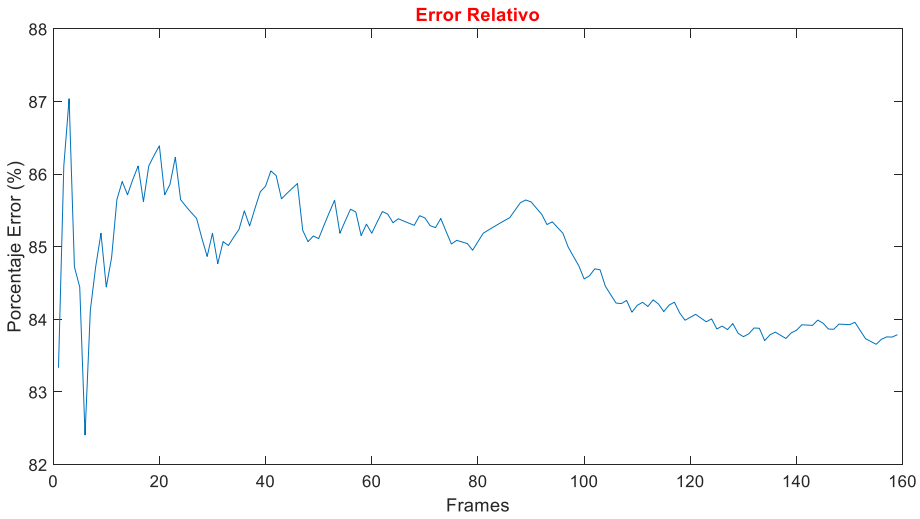


Figura 56. Error de detección peces molly.

Fuente: Autoría Propia

Por otro lado, el sistema ha funcionado correctamente incluso con el porcentaje alto de error porque los peces se encuentran en la parte superior de la pecera y eso significa que están en un estado satisfecho. En la Figura 57 se puede observar como el sistema ha reaccionado correctamente y la franja de satisfecho esta mas alta.

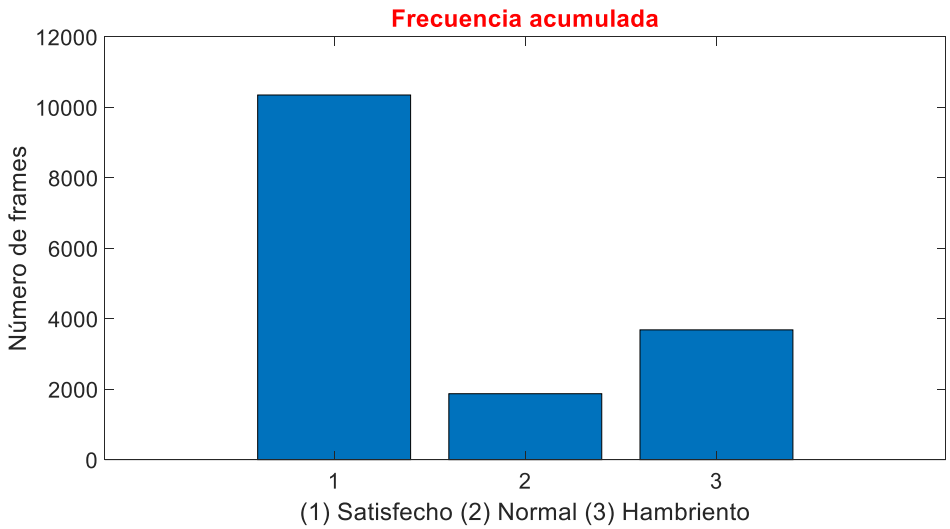


Figura 57. Peces cebra y peces molly.

Fuente: Autoría Propia

La entrada de los datos fuzzy obtenidos para el caso de los 18 peces cebra y molly se muestran en la Figura 58. Después de obtener estos resultados se procede aplicar el algoritmo difuso para determinar la salida correspondiente y evaluar la activación o desactivación del sistema de alimentación automático.

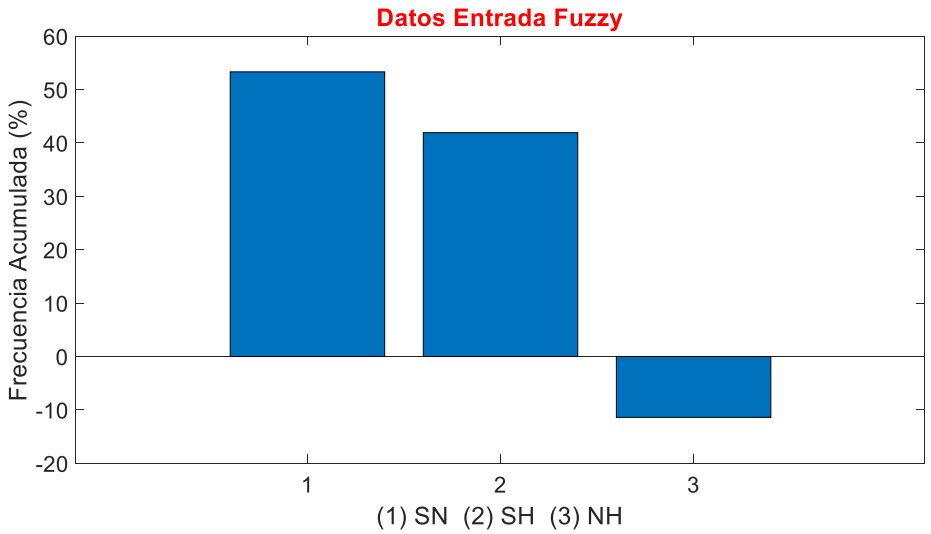


Figura 58. Datos fuzzy pez molly.

Fuente: Autoría Propia

Finalmente, los resultados del controlador difuso para el caso de los peces molly y cebra se muestran en la Figura 59, donde se puede apreciar que el valor promedio es de 30, por lo tanto no se activara el alimento y esto concuerda con los análisis posteriores.

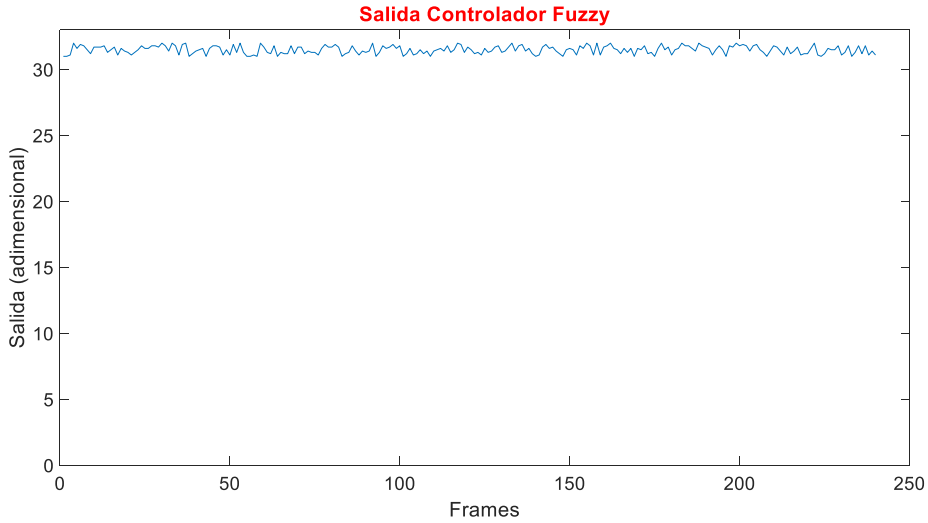


Figura 59. Salida del controlador difuso caso molly.

Fuente: Autoría Propia

CAPITULO V

CONCLUSIONES, RECOMENDACIONES, BIBLIOGRAFIA Y ANEXOS

5.1 Conclusiones

- Existen varios algoritmos utilizados para la detección de animales dentro de una imagen, pero existe una deficiencia notable en lo que se refiere a detección de peces dentro de una imagen o un video. El proyecto DeepFish es uno de los pioneros en la detección y utilización de algoritmo Deep Learning para identificar peces dentro de un entono marítimo real. Sin embargo, este tipo de tecnología requiere de equipos con alto costo computacional para su operabilidad.
- En este proyecto se utilizó las características agregadas de canal (ACF) para desarrollar un algoritmo de detección de peces de la raza Cebra (Danio Renio) mediante con Deep Learning y basados en un dataset etiquetado manualmente con 393 imágenes y 1755 etiquetas.
- El algoritmo desarrollado permitió obtener un error de detección mínimo y máximo de 29.5% y 65%, respectivamente. Cabe recalcar que este error se debe a que el algoritmo basado en el detector ACF delimita a 2 o 3 peces en un mismo recuadro y lo contabiliza como un solo objeto, considerando este comportamiento se podría reducir el error hasta en un 15.4% y 45%.
- En el caso expuesto en los resultados se pudo determinar el correcto análisis del estado natural de los peces. En este caso se realizó una alimentación manual de los peces para determinar si el algoritmo desarrollado era capaz de identificar dicho evento y la reacción del cardumen dentro de la pecera. Esta prueba arrojó un valor de salida promedio de 31.5% muy por debajo del valor de alimentación de 60%.
- El sistema desarrollado en una pecera y en un entorno controlado permitió generar un dataset propio, al mismo tiempo que se pudo realizar diferentes pruebas para identificar el comportamiento de los peces; de esta manera se pudo comprender el sentimiento y movimientos del cardumen cuando están satisfechos o hambrientos.

- Los peces cebra (Danio Rerio) utilizados en el desarrollo del presente proyecto de tesis son muy utilizados en los proyectos investigativos de diferentes asignaturas como, por ejemplo: la medicina, la biología y la ingeniería. Por ese motivo se decidió utilizar este tipo de peces, adicionalmente se puede encontrar bastante información sobre el comportamiento natural de estos peces.

5.2 Recomendaciones

- En proyectos futuros se debe considerar la utilización de miniordenadores con tarjetas GPU incorporadas, debido a que los algoritmos Deep Learning necesitas de un alto costo computacional para el tratamiento de las matrices, pesos y las capas ocultas que componen una red de detección de objetos.
- Implementar una pecera con un sistema de iluminación adecuado para mejorar la calidad de las imágenes y de esta manera mejorar los resultados obtenidos por el algoritmo desarrollado.
- Se debe considerar la utilización del proceso descrito en esta tesis para generar nuevos proyectos de investigación y a su vez caracterizar el comportamiento natural de los peces cuando tienen otros instintos diferentes al de alimentarse o cuando tienen algún tipo de enfermedad.
- Se recomienda para entrenar modelos con deep learning tener una GPU pudiendo ser de minutos en vez de horas o días en lugar de meses entre ellos NVIDIA, AMD y GTX 1050 Ti de 4GB o 2GB de memoria.

5.3 BIBLIOGRAFÍA:

Aguirre, M. B. (2004). Comportamiento de los peces en la búsqueda y la captura del alimento. *Revista Colombiana de Ciencias Pecuarias*, *17*(1), 63–75. https://revistas.udea.edu.co/index.php/rccp/article/view/323926

Alshdaifat, N. F. F., Talib, A. Z., & Osman, M. A. (2020). Improved deep learning framework for fish segmentation in underwater videos. *Ecological Informatics*, *59*, 101121. https://doi.org/10.1016/J.ECOINF.2020.101121

Asus. (2019). *Laptop Asus* . Creditoseconomicos. https://www.creditoseconomicos.com/laptop-asus-g512li-p89655-15-6-8gb-ram-512gb-ssd-core-i7-color-negro/p

BricoGeek. (2021). *Raspberry Pi 4 | BricoGeek.com*. https://bit.ly/3xQ34Dv

Cayuel Fuentes, M. L., Alcaraz Pérez, F., & Flageu, M. A. (2012). El pez cebra, al servicio de la investigación en Cáncer. *Especial Biomedicina*, *28*, 1–3. https://digitum.um.es/digitum/bitstream/10201/29730/1/El%20pez%20cebra,%20al%20servicio%20de%20la%20investigaci%C3%B3n%20en%20c%C3%A1ncer.pdf

Cui, S., Zhou, Y., Wang, Y., & Zhai, L. (2020). Fish Detection Using Deep Learning. *Applied Computational Intelligence and Soft Computing*, *2020*. https://doi.org/10.1155/2020/3738108

Dawkins, M., Sherrill, L., Fieldhouse, K., Hoogs, A., Richards, B., Zhang, D., Prasad, L., Williams, K., Lauffenburger, N., & Wang, G. (2017). An open-source platform for underwater image & video analytics. *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, 898–906. https://doi.org/10.1109/WACV.2017.105

Fathima Syreen, R., & Merriliance, K. (2020). Deep Convolutional Networks for Underwater Fish Localization and Species Classification. *International Journal of Advanced Research in Engineering and Technology*, *11*(11), 91–100. https://doi.org/10.34218/IJARET.11.11.2020.009

Guio Rodriguez, C. F. (2021). *Desarrollo de un sistema de seguimiento de la navegación de peces cebra en 3 dimensiones usando aprendizaje profundo*. https://repository.usta.edu.co/handle/11634/35675

Gupta, S., Mukherjee, P., Chaudhury, S., Lall, B., & Sanisetty, H. (2021). DFTNet: Deep Fish Tracker with Attention Mechanism in Unconstrained Marine Environments. *IEEE Transactions on Instrumentation and Measurement*, *70*. https://doi.org/10.1109/TIM.2021.3109731

Ishaq, O., Sadanandan, S. K., & Wählby, C. (2017). Deep Fish: Deep Learning-Based Classification of Zebrafish Deformation for High-Throughput Screening. *Journal of Biomolecular Screening*, *22*(1), 102–107. https://doi.org/10.1177/1087057116667894

Martínez Peiró, J. (2021). *Técnicas Avanzadas de Visión por Computador (VC) basadas en Deep Learning (DL) aplicadas a la monitorización de especies marinas (Bluefin Tuna)* [Universitat Politècnica de València]. https://riunet.upv.es/handle/10251/173351

Merchán, F., Galeano, S., & Poveda, H. (2014). Mejoras en el Entrenamiento de Esquemas de Detección de Sonrisas Basados en AdaBoost. *I+D Tecnológico*, *10*(2), 17–30. https://revistas.utp.ac.pa/index.php/id-tecnologico/article/view/21/html

Palomino Echeverría, S. (2018). *Mantenimiento y cría de peces cebra (Danio rerio) en el laboratorio: reproducción natural y artificial*. https://academica-e.unavarra.es/xmlui/handle/2454/29910

Pedersen, M., Haurum, J. B., Gade, R., Moeslund, T. B., & Madsen, N. (2019). Detection of Marine Animals in a New Underwater Dataset with Varying Visibility. In *Cumputer Vision Fundation* (pp. 18–26). https://www.kaggle.com/aalborguniversity/

Qin, H., Li, X., Liang, J., Peng, Y., & Zhang, C. (2016). DeepFish: Accurate underwater live fish recognition with a deep architecture. *Neurocomputing*, *187*, 49–58. https://doi.org/10.1016/J.NEUCOM.2015.10.122

Seguritec. (2015). *Cámaras de vigilancia para restaurantes y hoteles* . Seguritec.Cat. https://www.seguritec.cat/camaras-de-vigilancia-para-restaurantes-y-hoteles/

van Ranst, W., de Smedt, F., & Goedemé, T. (2018). GPU Accelerated ACF Detector. *Proceedings of the 13th International Joint Conference on Computer Vision*, *5*, 242–248. https://doi.org/10.5220/0006585102420248

Vargas-Vargas, R. A. (2017). Pez cebra (Danio rerio) y anestesia. Un modelo animal alternativo para realizar investigación biomédica básica. *Anestesia En México*, *29*(Supl. No.1), 86–96. http://www.scielo.org.mx/scielo.php?pid=S2448-87712017000400086&script=sci_arttext

Vidaor Maristany, A. (2020). *Desarrollo de un sistema de detección de peces* [Universitat Politècnica de Catalunya]. https://upcommons.upc.edu/handle/2117/329979

Viera-Maza, G. (2017). *PROCESAMIENTO DE IMÁGENES USANDO OPENCV APLICADO EN RASPBERRY PI PARA LA CLASIFICACIÓN DEL CACAO*.

Zhang, W., Wu, C., & Bao, Z. (2021). DPANet: Dual Pooling-aggregated Attention Network for fish segmentation. *IET Computer Vision*. https://doi.org/10.1049/CVI2.12065

5.4 ANEXOS

Programación completa Matlab

```
close all
clear all
clc

raspiObj = raspi();
s = servo(raspiObj,18);
writePosition(s,180)

cam = webcam(raspiObj,1);
%getFile(raspiObj,'D:\MECHATRONICS SYSTEMS\Proyectos\Maestria Vision Pez\DeepLearning_Matlab\Detector.mat')
x=coder.load('detector_estructura.mat')
detector = acfObjectDetector(x.detector_estructura.Classifier,x.detector_estructura.TrainingOptions)

i = 1;
results = struct('Boxes',[0 0 0 0],'Scores',[0]);
start = tic;
fprintf('Entering into while loop.\n');

valor = [0]
caja = [0 0 0 0]
T_frecuencia(1) = 0;
T_frecuencia(2) = 0;
T_frecuencia(3) = 0;
frame = 0;
```

```
while true

    %Capture image from webcams
    I = snapshot(cam);
    frame = frame + 1;

    elapsedTime = toc(start);
    %Process frames at 1 per second
    if elapsedTime > 2

       [bboxes, scores] = detect(detector,I);

        % Select strongest detection
        [~,idx] = max(scores);
        k=1;
        satisfecho = 0;
        normal = 0;
        hambriento = 0;
        valor = [0];
        caja = [0 0 0 0];

        for j = 1:length(scores)

            if (scores(j) >= max(scores)-65)
                caja(k,:)=bboxes(j,:);
                valor(k)=scores(j);
                k=k+1;
            end
        end

        for j = 1:length(valor)%
```

```
                if (caja(j,2) >= 0 )
                    if (caja(j,2) < 100 )
                        satisfecho = satisfecho + 1;
                    end
                end

                if (caja(j,2) >= 100)
                    if (caja(j,2) < 140 )
                        normal = normal + 1;
                    end
                end

                if (caja(j,2) >= 140)
                    if (caja(j,2) < 240 )
                        hambriento = hambriento + 1;
                    end
                end
            end

            total = satisfecho + normal +
hambriento;
            frecuencia(1)=satisfecho/total*100;
            frecuencia(2)=normal/total*100;
            frecuencia(3)=hambriento/total*100;


            T_frecuencia(1)=
T_frecuencia(1)+frecuencia(1);
            T_frecuencia(2)=
T_frecuencia(2)+frecuencia(2);
            T_frecuencia(3)=
T_frecuencia(3)+frecuencia(3);
```

```
            Fuzzy_S_N = (T_frecuencia(1) -
T_frecuencia(2))/frame;
            Fuzzy_S_H = (T_frecuencia(1) -
T_frecuencia(3))/frame;
            Fuzzy_N_H = (T_frecuencia(2) -
T_frecuencia(3))/frame;

            T_Fuzzy(1)= Fuzzy_S_N ;
            T_Fuzzy(2)= Fuzzy_S_H ;
            T_Fuzzy(3)= Fuzzy_N_H ;

            fis = readfis('Fuzzy_Pez');
            alimentar = evalfis(fis,[Fuzzy_S_N
Fuzzy_S_H Fuzzy_N_H])


            figure (1);
            bar(frecuencia);
            title('Frecuencia por
segundo','FontSize',18,...
            'FontWeight','bold','Color','r');
            ylim([0 100]);


            figure (2);
            bar(T_frecuencia);
            title('Frecuencia
acumulada','FontSize',18,...
    'FontWeight','bold','Color','r');

            figure (3);
```

```
            bar(T_Fuzzy);
            title('Datos Entrada
Fuzzy','FontSize',18,...
    'FontWeight','bold','Color','r');


            num_pez(i) = length(valor);
            promedio = sum(num_pez)/length(num_pez);
            error(i) = abs(promedio-8)/8*100;

            figure (4);
            plot(error);
            title("Error Relativo",'FontSize',18,...
            'FontWeight','bold','Color','r');


            if (alimentar >= 60)
                writePosition(s,180)
                pause(1)
                writePosition(s,110)
                pause(1)
            end

            if (alimentar >= 90)
                writePosition(s,180)
                pause(1)
                writePosition(s,110)
                pause(1)
                writePosition(s,180)
                pause(1)
                writePosition(s,110)
                pause(1)
```

```
            end

             I =
insertObjectAnnotation(I,'rectangle',[1 1 320
100],"Zona
Satisfecha",'TextBoxOpacity',0.3,'FontSize',9,'Color
','green');
             I =
insertObjectAnnotation(I,'rectangle',[1 140 320
100],"Zona de
Alimentación",'TextBoxOpacity',0.3,'FontSize',9,'Col
or','red');
            for j=1:length(valor)
                I =
insertObjectAnnotation(I,'rectangle',caja(j,:),valor
(j));
            end


            i = i + 1;
        end

        figure (5);
        displayImage(raspiObj,I);


    end

    results = struct2table(results);

    release(vidPlayer);
```

Entrenamiento de la red

```
%% Entrenamiento Deep Learning
load ('Etiquetas_Buenas.mat')
Pez = selectLabels(gTruth3,'Pez')
trainingData =
objectDetectorTrainingData(Pez,'SamplingFactor',3,'W
riteLocation','TrainingData');

detector =
trainACFObjectDetector(trainingData,'NumStages',10);

save ('Detector.mat','detector')
```

Programación Raspberry

```
function Raspi_Detector()

raspiObj = raspi();
s = servo(raspiObj,18);
writePosition(s,180)
cam = webcam(raspiObj,1);

x=coder.load('detector_estructura.mat')
detector =
acfObjectDetector(x.detector_estructura.Classifier,x
.detector_estructura.TrainingOptions)

fis = getFISCodeGenerationData ('Fuzzy_Pez');

i = 1;
```

```
    results = struct('Boxes',[0 0 0
0],'Scores',[0]);
    start = tic;
    fprintf('Entering into while loop.\n');

    valor = [0; 0]
    caja = [0 0 0 0;0 0 0 0]
    T_frecuencia = [0 0 0];
    frecuencia = [0 0 0];
    T_Fuzzy = [0 0 0];
    scores = [];
    bboxes = [];
    frame = 0;
    while true

        %Capture image from webcams
        I = snapshot(cam);
        frame = frame + 1;

        elapsedTime = toc(start);
        %Process frames at 1 per second
        if elapsedTime > 1


           [bboxes, scores] = detect(detector,I);

            % Select strongest detection
            [~,idx] = max(scores);
            k=1;
            satisfecho = 0;
            normal = 0;
            hambriento = 0;
```

```
valor = [0; 0];
caja = [0 0 0 0];

for j = 1:length(scores)
    if (scores(j) >=( max(scores) - 40))
        caja(k,1:4) = bboxes(j,1:4);
        valor(k) = scores(j);
        k=k+1;
    end
end

%I =
insertObjectAnnotation(I,'rectangle',caja(1,:),lengt
h(valor));


for j = 1:length(valor)
    if (caja(j,2) >= 0 )
        if (caja(j,2) < 100 )
            satisfecho = satisfecho + 1;
        end
    end

    if (caja(j,2) >= 100)
        if (caja(j,2) < 140 )
            normal = normal + 1;
        end
    end

    if (caja(j,2) >= 140)
        if (caja(j,2) < 240 )
            hambriento = hambriento + 1;
```

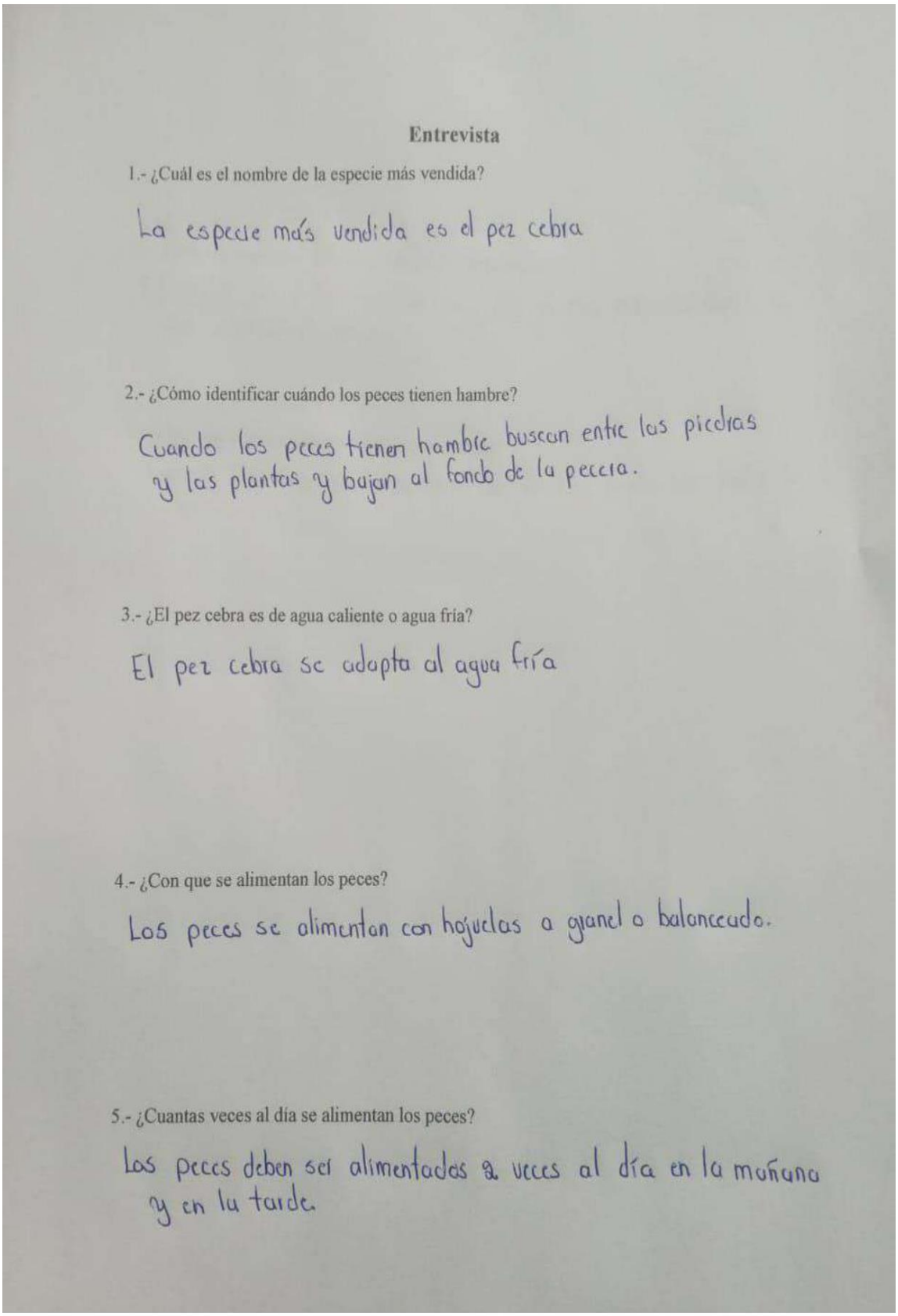
```
                    end
                end
            end

            total = satisfecho + normal +
hambriento;
            frecuencia(1)=satisfecho/total*100;
            frecuencia(2)=normal/total*100;
            frecuencia(3)=hambriento/total*100;

            T_frecuencia(1)=
T_frecuencia(1)+frecuencia(1);
            T_frecuencia(2)=
T_frecuencia(2)+frecuencia(2);
            T_frecuencia(3)=
T_frecuencia(3)+frecuencia(3);


            Fuzzy_S_N = (T_frecuencia(1) -
T_frecuencia(2))/frame;
            Fuzzy_S_H = (T_frecuencia(1) -
T_frecuencia(3))/frame;
            Fuzzy_N_H = (T_frecuencia(2) -
T_frecuencia(3))/frame;

            T_Fuzzy(1)= Fuzzy_S_N ;
            T_Fuzzy(2)= Fuzzy_S_H ;
            T_Fuzzy(3)= Fuzzy_N_H ;

            alimentar = evalfis(fis,[Fuzzy_S_N
Fuzzy_S_H Fuzzy_N_H])
```

```
                if (alimentar >= 60)
                        writePosition(s,180)
                        pause(1)
                        writePosition(s,110)
                        pause(1)
                end

                if (alimentar >= 90)
                    writePosition(s,180)
                    pause(1)
                    writePosition(s,110)
                    pause(1)
                    writePosition(s,180)
                    pause(1)
                    writePosition(s,110)
                    pause(1)
                end
                for j=1:length(valor)
                    I =
insertObjectAnnotation(I,'rectangle',caja(j,:),valor
(j));
                end

                i = i + 1;
            end
            displayImage(raspiObj,I);

        end
        release(vidPlayer);

        end
```

Programación para implementar en Raspberry

```
clear all
close all
clc

board = targetHardware('Raspberry Pi')
board.CoderConfig.TargetLang = 'C++';
dlcfg = coder.DeepLearningConfig('arm-compute')
dlcfg.ArmArchitecture = 'armv7'
dlcfg.ArmComputeVersion = '20.02.1'
clc
board.CoderConfig.DeepLearningConfig = dlcfg
deploy(board, 'Raspi_Detector')
```

Entrevista



Entrevista

1.- ¿Cuál es el nombre de la especie más vendida?

La especie más vendida es el pez cebra

2.- ¿Cómo identificar cuándo los peces tienen hambre?

Cuando los peces tienen hambre buscan entre las piedras y las plantas y bajan al fondo de la pecera.

3.- ¿El pez cebra es de agua caliente o agua fría?

El pez cebra se adapta al agua fría

4.- ¿Con que se alimentan los peces?

Los peces se alimentan con hojuelas a granel o balanceado.

5.- ¿Cuantas veces al día se alimentan los peces?

Los peces deben ser alimentados 2 veces al día en la mañana y en la tarde.

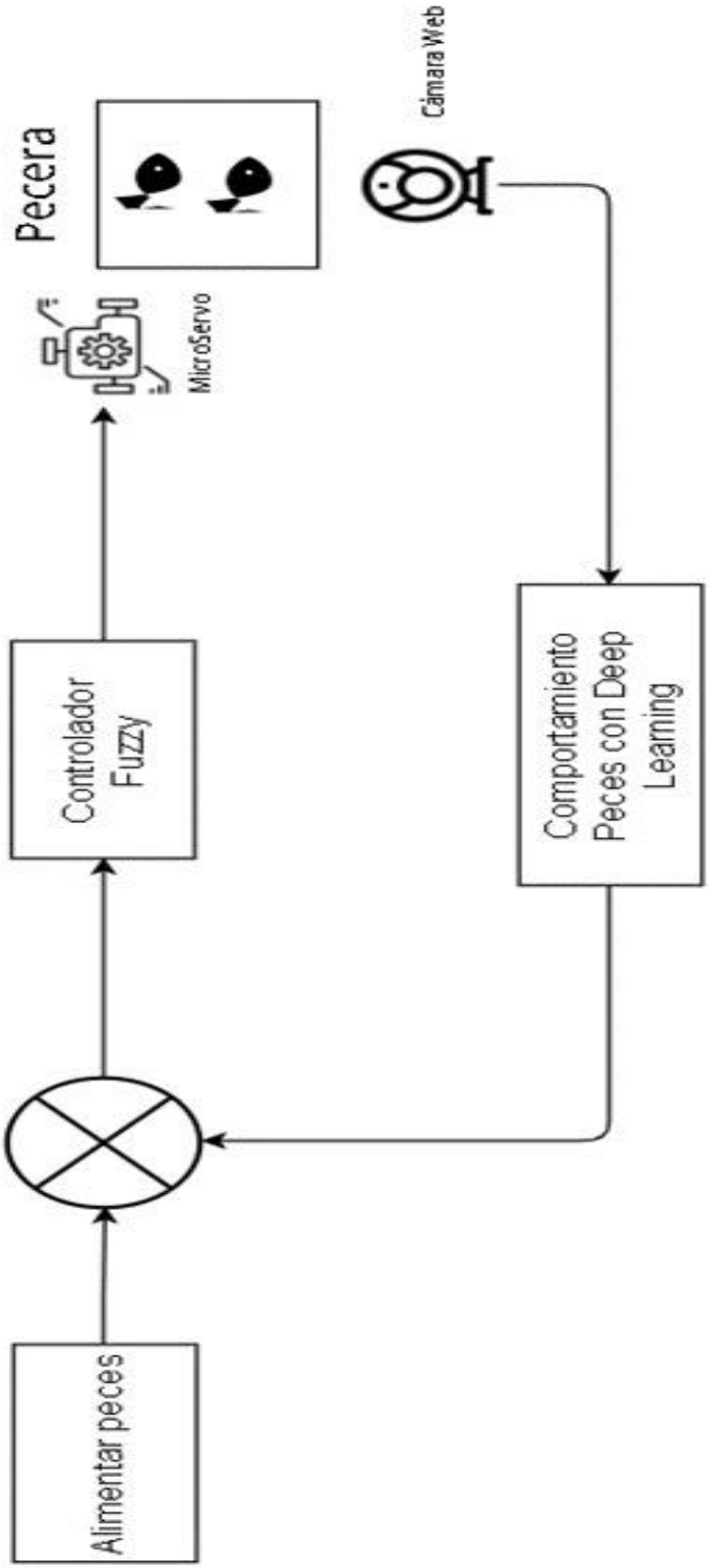
6.- ¿Cada que tiempo se debe cambiar el agua de la pecera?

El agua se debe cambiar una vez al mes dependiendo del número de peces.

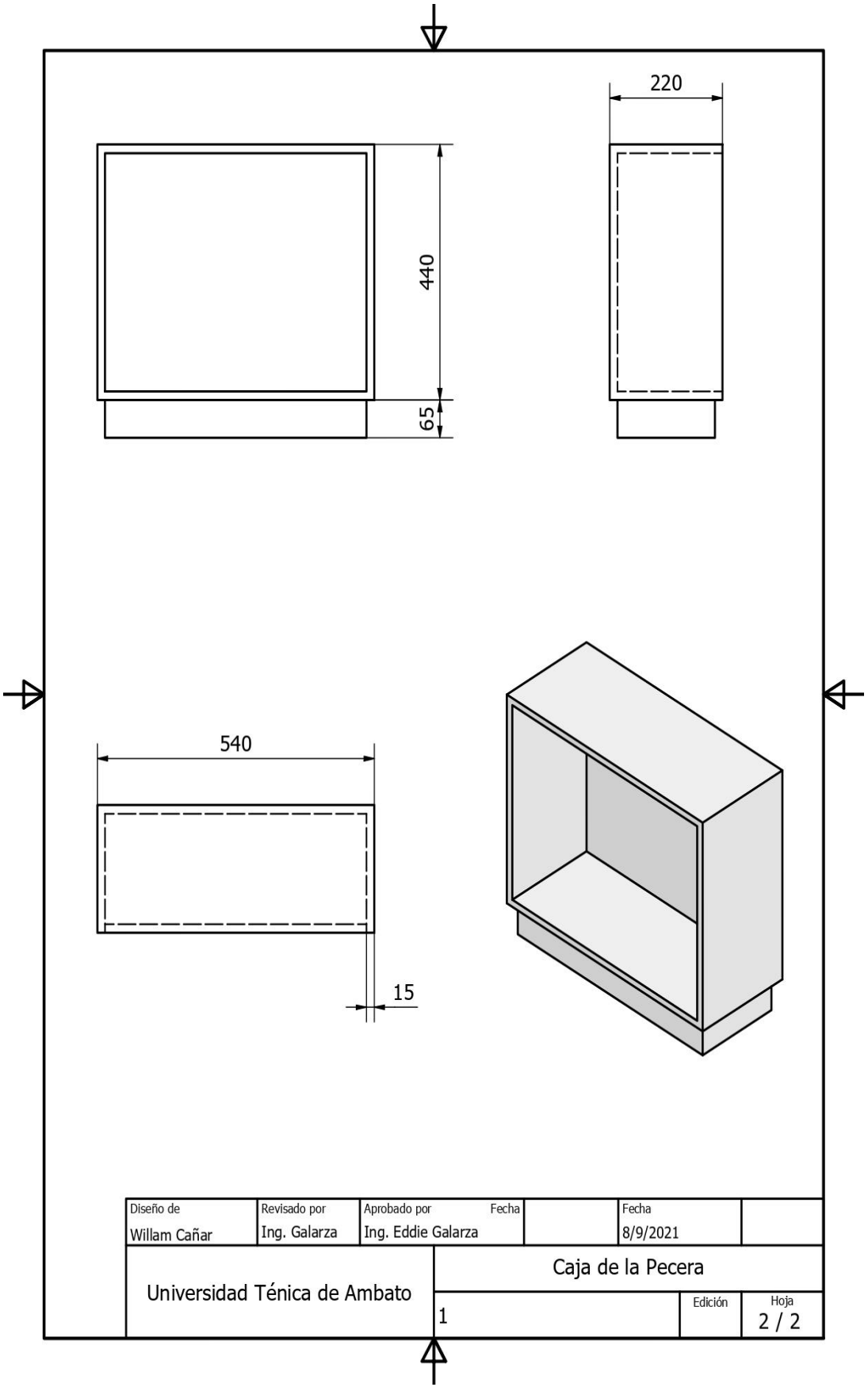
7.- ¿Como saber cuándo los peces están satisfechos?

Cuando los peces se encuentran satisfechos nadan por toda la pecera.

Diagrama sistema de control



Plano pecera

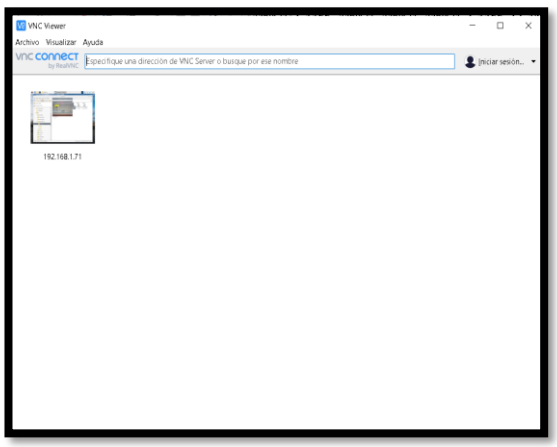


Manual de manejo

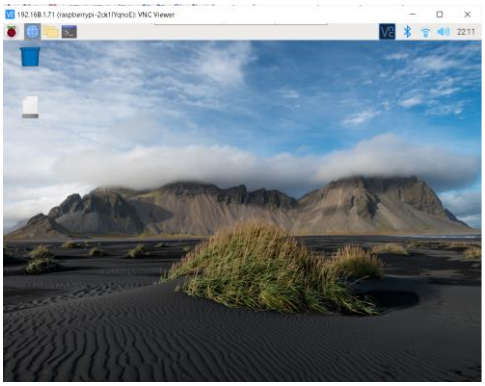
A continuación, se detallará los pasos para poner en funcionamiento la plataforma.

1. Encender la RapsBerry para se pueda establecer una conexión a internet.

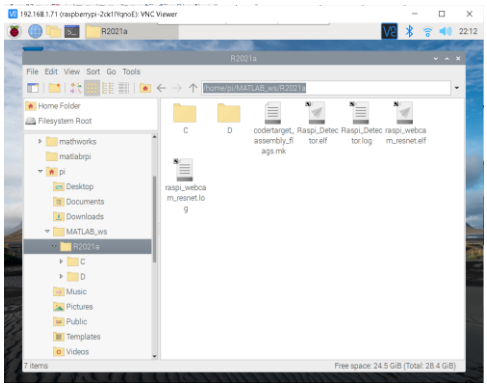
2. Ejecutar el programa VNC viewer donde se podrá encontrar la tarjeta controladora con una dirección IP, esto servirá para establecer una conexión remota con el sistema embebido.



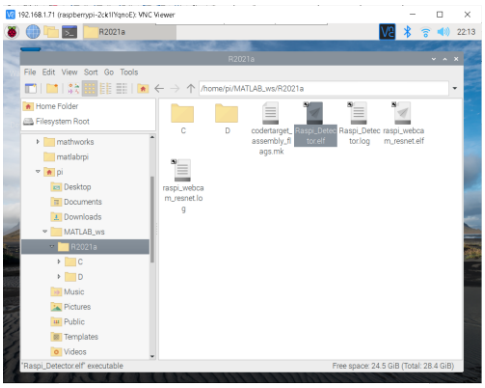
3. Ingresar a la tarjeta que se encuentra con la dirección IP, en este caso es 192.168.1.71



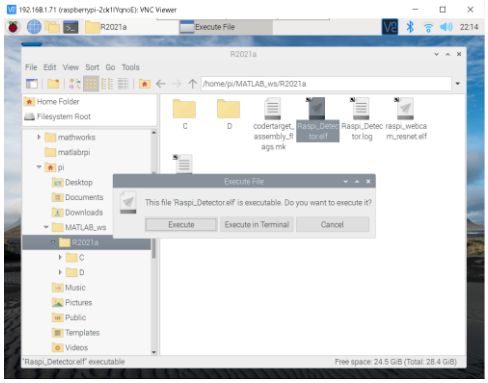
4. Dirigirse a la carpeta con la siguiente dirección /home/pi/MATLAB_ws/R2021a



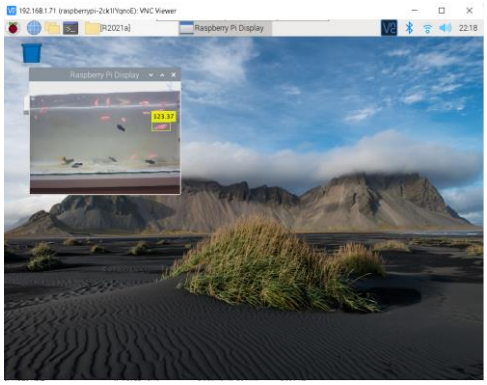
5. En esta dirección se debe identificar el archivo con el nombre de Raspi_Detector.elf. Este documento es el que contiene todo el funcionamiento del sistema de detección de peces y alimentación automática.



6. Ejecutar este archivo como administrador

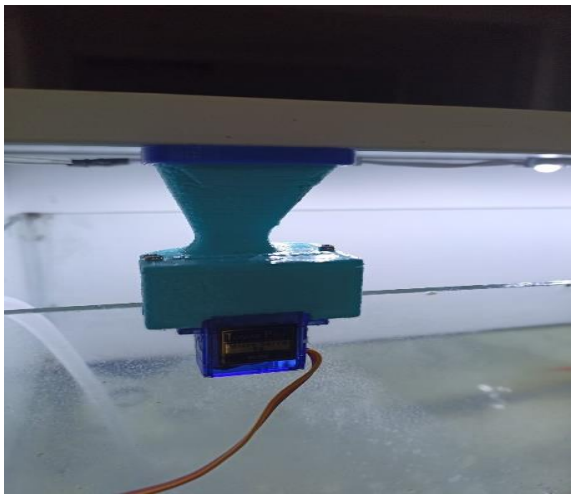


7. El sistema inicia su funcionamiento



8. Encender luces led para iluminación de pecera.

9. Proveer de alimento al dosificador



10. Ejecución del controlador para alimentación de peces con Deep Learning

Sugerencias

- Seguir las instrucciones de funcionamiento
- Utilizar el equipo en ambiente limpio y seco
- Instalar en un ordenador con buen rendimiento de GPU.
- Ubicar la cámara y sistema embebido en un sitio seguro
- Prender la luz del siempre cuando no tenga buena iluminación el ambiente.
- Dar seguimiento a la alimentación automática, proveer de alimento