UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE TELECOMUNICACIÓNES

TRABAJO DE INTEGRACIÓN CURRICULAR

TEMA:

"SISTEMA EMBEBIDO PARA LA OPTIMIZACIÓN DE LA REPRODUCCIÓN DE LOS PECES BETTA BASÁNDOSE EN ALGORITMOS DE DEEP LEARNING."

Trabajo de titulación previo a la obtención del título en Ingeniero en Telecomunicaciones

Línea de investigación: Innovación Tecnológica y de productos

AUTOR:

Andrés Joshue Cuenca Luna

DIRECTOR:

Edgar Alberto Maya Olalla

Ibarra – Ecuador 2026

UNIVERSIDAD TÉCNICA DEL NORTE
DIRECCIÓN DE BIBLIOTECA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO	
APELLIDOS Y NOMBRES:	Andrés Joshue Cuenca Luna

DATOS DE LA OBRA	
TÍTULO:	Sistema Embebido para la optimización de la reproducción de los Peces Betta basándose en algoritmos de deep learning
AUTOR (ES):	Andrés Joshue Cuenca Luna
FECHA: DD/MM/AAAA	09/09/2026
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	■ **PREGRADO** ☐ **POSGRADO**
TITULO POR EL QUE OPTA:	Ingeniería en Telecomunicaciones.
DIRECTOR:	MSC. Edgar Maya Olalla

2. CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 09 días del mes de septiembre de 2026

EL AUTOR:

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
ANDRES JOSHUE CUENCA LUNA

Andrés Joshue Cuenca Luna

CERTIFICACIÓN DEL DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Ibarra, 09 de septiembre 2026

Edgar Maya Olalla

DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

CERTIFICA:

Haber revisado el presente informe final del trabajo de Integración Curricular, el mismo que se ajusta a las normas vigentes de la Universidad Técnica del Norte; en consecuencia, autorizo su presentación para los fines legales pertinentes.

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
EDGAR ALBERTO MAYA OLALLA

(f) ……

Edgar Alberto Maya Olalla

C.C.: 1002702197

DEDICATORIA

Dedicado al esfuerzo de mis padres, a quienes estuvieron allí enseñándome biología, a mí mismo y a Dios que estuvo siempre a mi lado.

AGRADECIMIENTO

A todas las personas que estuvieron en mi camino formándome como persona y profesional. En especial a mis padres, Fatima Cedeño, a mis abuelos que lo dieron todo por mí, para estar en donde estoy y a mis amigos que me vieron crecer.

RESUMEN EJECUTIVO

La reproducción controlada de peces Betta splendens requiere la observación del comportamiento de los ejemplares y el seguimiento de las condiciones del entorno acuático, procesos que tradicionalmente dependen de la supervisión directa del encargado. En este trabajo se desarrolló un sistema embebido orientado al monitoreo automatizado de indicadores asociados al comportamiento reproductivo de esta especie mediante técnicas de deep learning. El prototipo integra una Raspberry Pi 4 como unidad principal de procesamiento, un ESP32 para la adquisición de variables fisicoquímicas, una cámara para la captura de video y sensores de temperatura, pH y sólidos disueltos totales (TDS). Para el procesamiento visual se entrenó e implementó un modelo YOLOv11 capaz de identificar al pez Betta macho, la hembra y la presencia de burbujas. A partir de las detecciones se incorporó una lógica de análisis espacial basada en la proximidad entre los ejemplares, permitiendo generar automáticamente eventos de avistamiento y realizar posteriormente la verificación de burbujas como indicador complementario del comportamiento reproductivo. La información obtenida se almacena en Firebase Realtime Database y puede ser consultada mediante una aplicación móvil desarrollada en MIT App Inventor, complementada con mecanismos de notificación a través de Telegram. Durante 150 días de operación se registraron 25 875 avistamientos, 10 482 imágenes asociadas a estos eventos, 2014 verificaciones positivas de burbujas y 1 405 903 registros históricos de las variables ambientales. Los resultados obtenidos evidencian la capacidad del prototipo para integrar visión artificial, sensado ambiental, almacenamiento en la nube y acceso remoto dentro de una arquitectura de monitoreo

continuo, constituyéndose en una herramienta tecnológica de apoyo para el seguimiento automatizado del proceso reproductivo de los peces Betta splendens.

Palabras claves: Betta splendens, deep learning, sistema embebido, visión artificial, Internet de las Cosas, Raspberry Pi.

ABSTRACT

The controlled breeding of Betta splendens requires continuous observation of fish behavior and monitoring of aquatic environmental conditions, processes that traditionally depend on direct human supervision. This work developed an embedded system for the automated monitoring of indicators associated with the reproductive behavior of this species using deep learning techniques. The prototype integrates a Raspberry Pi 4 as the main processing unit, an ESP32 for physicochemical data acquisition, a camera for real-time video capture, and sensors for temperature, pH, and total dissolved solids (TDS). A YOLOv11 model was trained and deployed to identify male and female Betta fish as well as the presence of bubbles. Based on these detections, a spatial analysis method was implemented to evaluate the proximity between the fish, enabling the automatic generation of sighting events and the subsequent verification of bubbles as a complementary indicator of reproductive behavior. The generated information is stored in Firebase Realtime Database and can be remotely accessed through a mobile application developed with MIT App Inventor, complemented by notifications through Telegram. During 150 days of operation, the system recorded 25,875 sighting events, 10,482 images associated with these events, 2,014 positive bubble verifications, and 1,405,903 historical

environmental records. The obtained results demonstrate the prototype's ability to integrate computer vision, environmental sensing, cloud-based data storage, and remote access within a continuous monitoring architecture, providing a technological support tool for the automated monitoring of the reproductive process of Betta splendens.

Keywords: Betta splendens, deep learning, embedded system, computer vision, Internet of Things, Raspberry Pi.

ÍNDICE DE CONTENIDOS

1. Capitulo I. Antecedentes. 16

1.1. Tema 16

1.2. Planteamiento del problema 16

1.3. Alcance 18

1.4. Justificación 19

1.5. Objetivos 22

1.5.1. Objetivo General 22

1.5.2. Objetivos Específicos 23

2. Capítulo II. Fundamentación Teórica 24

2.1. Acuariofilia 24

2.1.1. Biología de los peces Betta splendens 25

2.1.2. Parámetros fisicoquímicos 28

2.1.3. Reproducción 28

2.1.4. Ciclo de reproducción 29

2.1.5. Importancia en reproducción 31

2.1.6. Problemática de su reproducción 31

2.2. Factores o Variables para medir 32

2.2.1. Temperatura 32

2.2.2. Salinidad 33

2.2.3. pH 33

2.3. Inteligencia Artificial 34

2.3.1. Machine Learning 34

2.3.2. Deep Learning 35

2.3.3. Visión Artificial 36

2.3.4. Modelos de detección de objetos 36

3. Capítulo III. Metodología 47

3.1. Metodología de desarrollo 47

3.1.1. Modelo en cascada adoptado 47

3.2. Especificación de requerimientos 48

3.2.1. Nomenclatura de requerimientos 49

3.2.2. Requerimientos de los Stakeholders 49

3.2.3. Requerimientos del sistema 50

3.2.4. Requerimientos de arquitectura 52

3.3. Selección del hardware 53

3.3.1. Evaluación comparativa del hardware 53

3.4. Selección de Software 54

3.4.1. Lenguaje de programación 55

3.4.2. Algoritmos de deep learning 56

3.5. Diseño del sistema 57

3.5.1. Diagrama de bloques del sistema 58

3.5.2. Diagrama de arquitectura 60

3.5.3. Diagrama de flujo del sistema 61

3.5.4. Diagrama de conexión 64

3.5.5. Desarrollo del Software 67

4. IMPLEMENTACIÓN Y RESULTADOS 70

4.1. Implementación del Sistema 70

4.1.1. Implementación del subsistema de sensado ambiental 72

4.1.2. Implementación del subsistema de visión artificial 73

4.1.3. Implementación de la lógica de avistamientos 78

4.1.4. Implementación de la verificación de burbujas 80

4.1.5. Implementación de la aplicación móvil 81

4.2. Resultados del período de operación del sistema 83

4.3. Resultados del subsistema de visión artificial 84

4.4. Resultados del monitoreo de variables fisicoquímicas 85

5. Conclusiones y Recomendaciones 87

5.1. Conclusiones..... 87

5.2. Recomendaciones 89

6. Anexos..... 101

6.1. Anexo A. Evidencia fotográfica del prototipo implementado..... 101

6.2. Anexo B. Evidencia del sistema de visión artificial en funcionamiento 108

6.3. Anexo C. Evidencia de generación de avistamientos y verificación de burbujas 111

6.4. Anexo D. Evidencia del almacenamiento en Firebase Realtime Database 114

6.5. Anexo E. Fragmentos relevantes del código fuente 117

ÍNDICE DE FIGURAS

Figura 1 26

Figura 2 27

Figura 3 28

Figura 4 41

Figura 5 43

Figura 6 59

Figura 7 61

Figura 8 64

Figura 9 66

Figura 10 71

Figura 11 75

Figura 12 76

Figura 13 79

Figura 14 81

Figura 15 82

Figura 16 102

Figura 17 103

Figura 18 103

Figura 19 105

Figura 20 106

Figura 21 107

Figura 22 108

Figura 23 109

Figura 24 110

Figura 25 110

Figura 26 111

Figura 27 112

Figura 28 113

Figura 29 114

Figura 30 115

Figura 31 116

Figura 32 116

Figura 33 117

Figura 34 117

Figura 35 123

Figura 36 124

Figura 37 125

ÍNDICE DE TABLAS

Tabla 1 49

Tabla 2 50

Tabla 3 51

Tabla 4 52

Tabla 5 53

Tabla 6 55

Tabla 7 56

Tabla 8 69

Tabla 9 73

Tabla 10 83

Tabla 11 84

Tabla 12 85

ÍNDICE DE ECUACIONES

Ec. 1……40

Ec. 2 79

1. Capitulo I. Antecedentes.

En este capítulo se describen de manera exhaustiva los fundamentos que sustentan la elaboración de la presente tesis. Estos elementos comprenden el tema a tratar, la problemática abordada, los objetivos propuestos, el alcance del estudio y la justificación, con el propósito de destacar la relevancia de llevar a cabo e implementar este proyecto.

1.1. Tema

Sistema Embebido para la optimización de la reproducción de los Peces Betta basándose en algoritmos de deep learning.

1.2. Planteamiento del problema

La cría de peces ornamentales en Ecuador, actividad de significativa importancia económica y generadora de empleo a nivel nacional e internacional, se ve amenazada por un problema crítico: el desequilibrio provocado por la constante presión sobre las poblaciones. Esta presión se traduce en la extracción indiscriminada de especies de sus hábitats naturales para su comercialización, lo que resulta en la desaparición de especies nativas y la propiciación de entornos vulnerables. Este fenómeno plantea desafíos significativos para la sostenibilidad ambiental y destaca la necesidad de implementar medidas efectivas que regulen y mitiguen el impacto negativo de la cría de peces ornamentales en la biodiversidad local e internacional.

Los peces Betta se encuentra entre las veinte especies de peces ornamentales más demandadas en todo el mundo, habita en aguas poco profundas, turbias y vegetadas de

ríos y arrozales. Se los identifica por poseer colores intensos, dimorfismo sexual, y aletas bien desarrolladas con lo cual se comunican a la hora de reproducirse para atraer a las hembras. Los machos se los conoce también por ser agresivos y territoriales por lo que pueden atacar a las hembras si no están listos para reproducirse, estos mismos construyen nidos de burbujas y cuidan de sus progenitores. Las hembras por otro lado se reproducen desde las 12 semanas, aunque se recomienda esperar hasta las 20-28 semanas para su completo desarrollo. [48], [53]. Garantizar la madurez es crucial para evitar agresiones tras la puesta de huevos, la hembra abandona el nido y el macho incuba durante 2-4 horas se debe tener en cuenta el mantenimiento y limpieza adecuadamente los peces y alevines ya que es esencial para prevenir infecciones [4].

La cría de peces ornamentales de agua dulce es una actividad común, facilitada por el uso de acuarios. A nivel nacional, la producción de peces ornamentales se percibe como rentable, aunque algunos acuicultores enfrentan desafíos para comercializar cantidades suficientes. La alta demanda en el sector ornamental genera competencia, resultando en la producción masiva. En el caso de Betta splendens, una especie ornamental, la eficiencia productiva y el éxito comercial dependen en parte de sus características estéticas [18], [19], [45], [4], [43] Se ha estudiado la reproducción de estos peces debido a su importante e inusual comportamiento así mismo se ha empleado en diferentes ámbitos de la ciencia en la evaluación de formación y contaminantes ambientales [30]. El comercio en peces betta macho genera grandes ganancias debido a su llamativa forma y color, se ha visto que los desarrollos del sexo y biología de estos peces involucran aspectos y factores ambientales para determinar el sexo es por esta razón

por la que se busca extender la producción de esta especie involucrando el desarrollo tecnológico [15].

Este estudio tiene como finalidad desarrollar un sistema embebido para facilitar la reproducción controlada de peces Betta en cautiverio. Este sistema incluye la implementación de varios sensores: temperatura (para mantener el agua en un rango óptimo que favorece la reproducción), pH (para asegurar que las condiciones del agua sean las adecuadas para la salud de los peces), salinidad (importante para mantener un ambiente acuático equilibrado), además de luces (que pueden simular ciclos naturales de luz y oscuridad, influenciando así el comportamiento reproductivo) y una compuerta (para permitir la interacción entre el macho y la hembra en el momento adecuado). El corazón del sistema es un algoritmo de Deep Learning, que asistirá al encargado enviando una alarma al teléfono cuando los peces estén listos para procrear. Este enfoque innovador no solo proporcionará un ambiente óptimo para la reproducción, sino que también identificará con precisión el momento adecuado para abrir la compuerta, facilitando así la reproducción exitosa de los peces Betta en cautiverio.

1.3. Alcance

El alcance del proyecto se inicia con la definición de la base teórica a través de una revisión bibliográfica exhaustiva. Esta fase es esencial para adquirir un conocimiento profundo de los aspectos biológicos de la reproducción de los peces Betta, así como de los fundamentos y parámetros del deep learning. Este conocimiento fundamenta el desarrollo tecnológico del sistema, asegurando que esté bien informado tanto biológica como tecnológicamente.

La siguiente etapa se centra en el diseño de la red neuronal utilizando algoritmos de deep learning para la clasificación de imágenes, junto con el desarrollo del sistema embebido basado en Raspberry Pi. Este sistema integrará sensores para medir el pH, la temperatura y la salinidad, además de incorporar actuadores como calentadores de agua, luces LED y un servomotor para el control ambiental de la pecera. Este paso es crucial, ya que representa la aplicación práctica de la teoría, enlazando la biología de los peces Betta con la inteligencia artificial.

Posteriormente, el proyecto avanza hacia la construcción física de la pecera que integra el sistema embebido y la inteligencia artificial, así como el desarrollo de una aplicación móvil. Esta aplicación facilitará a los usuarios la monitorización de la pecera, visualización de datos y recepción de alertas sobre el comportamiento reproductivo de los peces. Este componente es clave para la accesibilidad y utilidad del sistema, permitiendo una interacción sencilla y efectiva con el usuario final.

Finalmente, se lleva a cabo la validación del desempeño del sistema mediante pruebas exhaustivas en el prototipo. Este proceso es crucial para asegurar que el sistema funcione correctamente en términos de precisión, confiabilidad y eficacia. Se evaluarán todos los componentes del sistema, desde la red neuronal hasta la funcionalidad del sistema embebido y la usabilidad de la aplicación móvil, garantizando así que el sistema sea viable y efectivo en un entorno real.

1.4. **Justificación**

La reproducción de peces Betta ha ganado popularidad debido a su atractivo y fácil disponibilidad en el mercado de mascotas y acuarios. Esta demanda ha generado

interés en mejorar métodos de reproducción y conservación para abordar preocupaciones sobre sobrepesca y degradación del hábitat natural. Según Arboleda et al. [3], la alta mortalidad de alevines, causada por agresión, destaca la necesidad de introducir métodos seguros de reproducción. El proceso de apareamiento continuo y la compatibilidad entre macho y hembra son cruciales para la supervivencia y perpetuación. [2] Menciona que la primera mitad de 1992, Estados Unidos importó peces ornamentales por millones de dólares, principalmente de Asia y algunos países americanos. [39] Alega que en Ecuador la cría de peces betta está creciendo gracias a su adaptabilidad a diversas condiciones climáticas, lo que lo hace adecuado para el cautiverio, su comercio es atractivo debido a sus vibrantes colores, lo que lo posiciona como una opción económicamente viable para el desarrollo rural. Sin embargo, se necesita el desarrollo de técnicas para aumentar la producción y reducir costos. Las tecnologías están provocando cambios significativos a nivel global, impulsando una revolución tecnológica basada en el conocimiento científico, la creatividad y la comunicación inteligente [1].

La Internet, es una red global interconectada mediante el protocolo estándar (TCP/IP), brinda servicios a miles de millones de usuarios en todo el mundo, incluyendo redes privadas y gubernamentales [31]. La Internet de las Cosas (IoT) promete mejorar la calidad de vida en áreas como servicios médicos, ciudades inteligentes y gestión del agua, al ofrecer decisiones automatizadas y herramientas de optimización [38]. Con avances tecnológicos, la IoT se dirige hacia una red global conectada a Internet, anticipando ser una valiosa fuente de información con el aumento diario de dispositivos conectados. Sin embargo, el crecimiento rápido de Internet subraya la necesidad urgente de medidas de seguridad web para proteger la privacidad y la seguridad empresarial [7]. En la actualidad,

el despliegue de dispositivos IoT es común, organizándose en niveles, desde el "borde" hasta el "borde extremo", donde la eficiencia se logra evitando la transmisión de datos innecesarios y ahorrando energía [42]

Sin embargo, la programación embebida permite desarrollar instrucciones precisas para cumplir funciones específicas, lo anterior apunta a tener una aplicación final de tamaño reducido y aun bajo costo que cubren un amplio campo en el mundo como es el caso de las industrias automotriz, electrónica o de consumo donde su arquitectura puede resistir a distintas situaciones extremas como humedad o temperatura [14]. Los sistemas embebidos son omnipresentes en nuestra vida cotidiana a diferencia de las PC convencionales, estos sistemas carecen de elementos visibles como pantallas o teclados, ya que la mayoría de los componentes están integrados en la placa base. [5] En la fase de procesamiento, se integra el procesador embebido con la tecnología hardware-software necesaria. La asignación de tareas entre circuitos digitales especializados (FPGA) y sistemas microprocesados se basa en la flexibilidad de adaptación del software y la velocidad de ejecución del hardware dedicado [16].

El aprendizaje automático, es esencial en la informática y ha evolucionado desde algoritmos tradicionales hacia el aprendizaje profundo. Aunque las aplicaciones del aprendizaje profundo son limitadas, se anticipa que resolverá desafíos en diversos dominios, representando una contribución prometedora en varias áreas y aplicaciones. [49]. El deep learning, se emplean regularmente algoritmos de aprendizaje automático para extraer información valiosa de extensas bases de datos empresariales. Estas bases de

datos incluyen registros de mantenimiento de equipos, solicitudes de préstamos, transacciones financieras, registros de servicios médicos y otros datos similares. [32].

El uso de herramientas para el desarrollo simplifica la implementación de funciones en hardware y elimina barreras en la programación. Se promueve la creación de más herramientas para facilitar la implementación de modelos de deep Learning en diversas plataformas, incentivando la exploración de sistemas embebidos como opciones móviles seguras para realizar inferencias de modelos de ML. Los resultados de este proyecto pueden ampliarse al analizar diferentes arquitecturas, mejorando así las recomendaciones para la implementación de modelos de ML en sistemas embebidos [11].

Este proyecto también es relevante desde la perspectiva de la biología, la investigación científica y la venta de los peces ya que proporciona una oportunidad para estudiar en profundidad los parámetros de reproducción de los peces Betta y cómo responden a cambios en su entorno. También una ganancia en el comercio y la toma de datos recopilatorios a través del sistema embebido que pueden contribuir al conocimiento científico y a la comprensión de los procesos de reproducción de esta especie.

1.5. Objetivos

1.5.1. Objetivo General

Diseñar un sistema embebido con alerta empleando deep learning para identificar el comportamiento reproductivo de los peces Betta.

1.5.2. Objetivos Específicos

- Definir la base teórica del sistema propuesto a través de una revisión bibliográfica, con el fin de determinar los parámetros de reproducción de los peces Betta y los parámetros del deep learning.
- Desarrollar una red neuronal especializada en la clasificación de imágenes, utilizando algoritmos de deep learning integrado en un sistema embebido basado en Raspberry Pi.
- Estructurar una pecera equipada el sistema embebido con la inteligencia artificial, incluyendo una aplicación móvil para el acceso a datos y alertas en tiempo real.
- Validar el desempeño del proyecto mediante pruebas en el prototipo que evidencien su funcionamiento.

2. Capítulo II. Fundamentación Teórica

En este capítulo, se expone la revisión bibliográfica realizada en el marco de la investigación para la elaboración del presente proyecto. Se detalla información relativa a las prácticas reproductivas, los factores fisicoquímicos que inciden en el desarrollo y comportamientos reproductivos, los cuales pueden ser cuantificados, así como el ciclo de apareamiento de esta especie. Además, se analiza la implementación de tecnologías de deep learning, vinculadas al aprendizaje automático, con un enfoque específico en las Redes Neuronales Artificiales. Estas tecnologías buscan determinar con precisión el momento óptimo para iniciar y finalizar el proceso reproductivo, integrando así aspectos biológicos y computacionales para optimizar la gestión reproductiva.

2.1. Acuariofilia

La acuariofilia puede conceptualizarse como una pasión que se fundamenta en la crianza de peces y otros seres acuáticos, así como en el cultivo de plantas acuáticas, ya sea en acuarios o estanques. El término acuariofilia engloba de manera amplia a los organismos acuáticos, incluyendo peces e invertebrados, como corales, crustáceos, moluscos y equinodermos, así como elementos como la roca viva (con un enfoque particular en este trabajo hacia los peces). Al admirar un acuario de peces ornamentales, raramente se reflexiona sobre el origen de estos fascinantes animales. La mayoría ha sido recolectada en su entorno natural o criada específicamente para este propósito, pasando por diversas manipulaciones antes de llegar a acuarios.

A lo largo del tiempo y hasta la actualidad, esta afición ha experimentado un notorio crecimiento y evolución, atrayendo a millones de entusiastas en todo el mundo. Este fenómeno ha permitido la cautividad exitosa de un número cada vez mayor de especies, algunas de las cuales eran inicialmente consideradas difíciles de mantener en entornos acuáticos domésticos. [54]

2.1.1. Biología de los peces Betta splendens

Los Betta splendens pertenecen a la familia Belontiidae , conocida comúnmente como el pez luchador [3]. Presenta un cuerpo alargado y comprimido, su aleta caudal es larga, redondeada y ancha; la aleta dorsal es alta y estrecha; las ventrales son puntiagudas; se alimentan de zooplancton, larvas de insectos y mosquitos [8]. Estas especies habitan en aguas dulces; mediante su comportamiento, los machos Bettas no pueden convivir en sociedad con otros de su mismo sexo ya que son muy territoriales, sin embargo, las hembras son capaces de compartir un mismo espacio con otras hembras por su poca agresividad y para mostrar cuales son sus diferencias como se muestra en la Figura 1 de hembra y machos [40].

Figura 1

Diferencia entre pez Betta Splendens hembra y macho



Fuente: Wikipedia https://es.wikipedia.org/wiki/Betta_splendens

Nota. Distinciones de los géneros de los peces Betta Splendens.

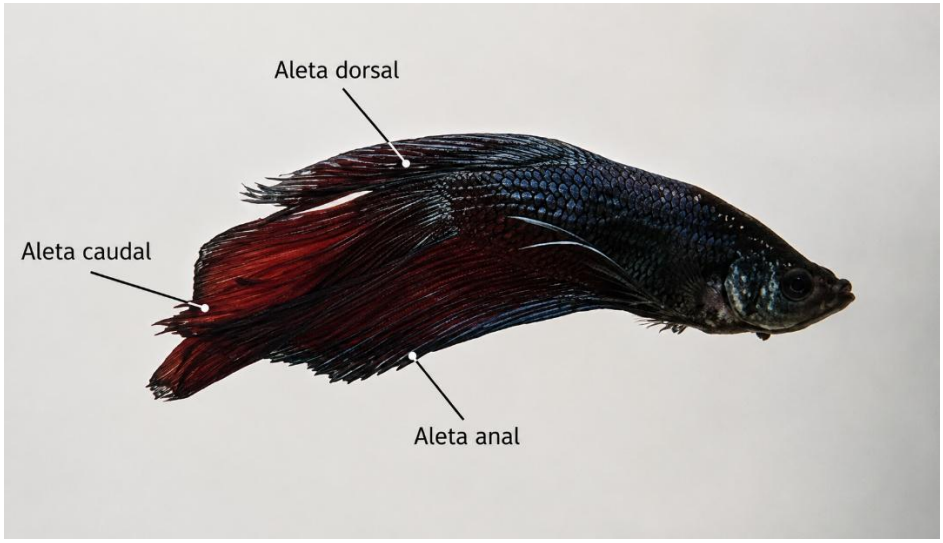
Los Betta splendens, comúnmente conocidos como peces Betta, son reconocidos por la exuberancia de sus colores y su naturaleza agresiva. El pez Betta se caracteriza por su comportamiento competitivo y defensivo, bien documentado en su uso para proteger territorios y competir por parejas entre los machos de su especie. Los machos muestran una clara diferencia en tamaño respecto a las hembras y destacan por sus aletas extensas y coloridas, las cuales despliegan al enfrentarse a otros machos o al cortejar a hembras en periodo de celo.

Durante las competiciones de territorio, se observan despliegues agonísticos que establecen jerarquías sociales, permitiendo que los machos de mayor estatus copulen con las hembras y dominen los territorios. Al notar la presencia de otro macho, los Betta

despliegan las aletas dorsal, caudal y anal, así como el opérculo[1]. En momentos de relajación, estas aletas se encuentran en descanso, al igual que el opérculo, tal y como se muestra en la Figura 2. En situaciones competitivas, las aletas dorsal, caudal y anal se despliegan en mayor área, y el opérculo se abre perpendicular al cuerpo, tal y como se muestra en la Figura 3. Además , los machos llevan a cabo acciones físicas como ataques, coletazos, mordiscos terminando en la copulación simbólica de cola y aleta en posición de abrazo. Estas acciones, desde su inicio hasta su persistencia y eventual abandono, están programadas en una red cerebral compleja en la que participan diversas moléculas neuromoduladores, tales como la serotonina, la dopamina y, arginina-vasotocina [40].

Figura 2

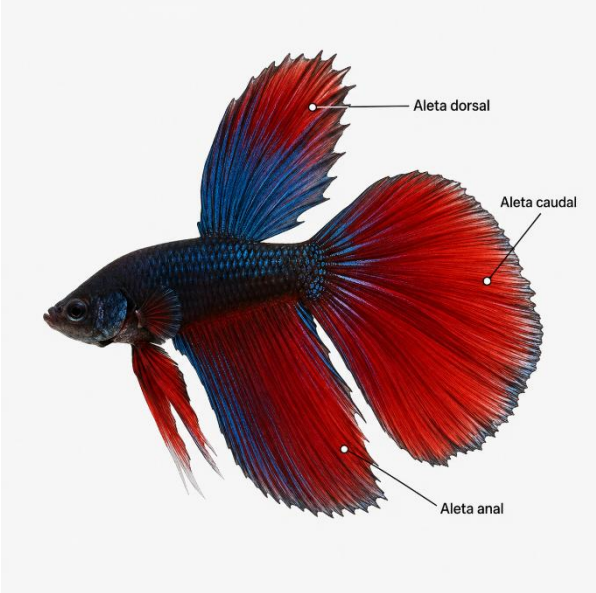
Pez macho en Descanso



[1]El opérculo: “Consiste en una aleta hecha de hueso sólido que resguarda y cubre las branquias.” [36].

Figura 3

Pez macho en posición de alerta.



2.1.2. Parámetros fisicoquímicos

Según Arboleda [3], estos peces poseen una notable capacidad para adaptarse con facilidad a una variedad de entornos, además exhiben la peculiar habilidad de tomar oxígeno directamente del aire. En el ámbito de su cuidado, se aconseja mantener condiciones óptimas para su desarrollo, procurando una temperatura que oscile entre 24 y 28°C, con un pH que se sitúe en el rango de 6,5 a 7,5.

2.1.3. Reproducción

El pez ornamental Betta, es una especie territorial que generalmente se encuentra en lagunas o aguas estancadas con poca corriente. Es fácil distinguir entre machos y hembras, ya que presentan un dimorfismo sexual evidente: los machos exhiben aletas más grandes y coloridas en comparación con las hembras. A partir del tercer mes de vida,

estos peces están listos para reproducirse, y para lograr este proceso de manera exitosa, se requiere mantener una temperatura constante en el rango de 27 a 30°C.

Durante el periodo reproductivo, es esencial proporcionar los cuidados adecuados, que incluyen el cortejo, desove y desarrollo de los alevines. En este proceso, el macho asume la responsabilidad de recolectar los huevos y cuidar el nido. Para prevenir posibles lesiones, la hembra debe ser retirada en ese momento. Aproximadamente al cuarto día de vida, cuando los alevines comienzan a nadar horizontalmente, se retira al macho para evitar que se alimente de ellos, ya que, en esta etapa, el macho se abstiene de comer para dedicarse al cuidado de los alevines [6].

Según Enríquez et al. [13], indica que para la reproducción del pez Betta, los machos comienzan a realizar burbujas en la superficie del agua, las cuales son elaboradas con su propia saliva para proteger los óvulos fecundados que la hembra produce durante el cortejo con el macho. La hembra expulsa entre 200 y 300 huevos durante 2 a 3 horas.

2.1.4. Ciclo de reproducción

Según Arboleda [3], el proceso de reproducción del Betta en acuarios se lleva a cabo mediante un protocolo específico que abarca diversas etapas:

- ***Adecuación del acuario*:** El acuario destinado a la reproducción no debe ser el mismo que el acuario comunitario. Se sugiere que los Bettas en el acuario de reproducción estén solos. Además, se recomienda decorar el acuario con plantas y mantener el agua calmada, disminuyendo la aireación o eliminándola por

completo. El acuario no debe contar con escondites y se sugiere que el nivel de agua no supere los 30 litros.

- ***Selección de la pareja*:** Para asegurar la atracción entre la pareja, se debe elegir una pareja fenotípicamente similar. En este caso, se prefieren machos y hembras con tonalidades rojas, como los seleccionados para el estudio.
- ***Alimentación*:** Durante el proceso de reproducción, se aconseja alimentar a la pareja de Bettas con una cantidad reducida de alimento en comparación con su ingesta habitual, ya que se ha observado que un exceso de alimento puede afectar la reproducción.
- ***Cortejo*:** El macho inicia el cortejo a la hembra mediante ataques, extendiendo sus aletas y abriendo sus opérculos. Construye un nido de burbujas en la superficie con su saliva para sostener los huevos.
- ***Apareamiento*:** Durante el apareamiento, el macho rodea a la hembra, realizando extrusiones para liberar los huevos, los cuales fertiliza. Después de la fecundación, el macho recoge los huevos con la boca y los lleva al nido de burbujas antes de que caigan al fondo. Este proceso puede durar aproximadamente 2 horas.
- ***Cuidado parental*:** Tras el apareamiento, el macho cuida el nido y patrulla para evitar la entrada de intrusos. El cuidado parental incluye evitar que la hembra deshaga el nido. Durante la incubación de los huevos, que dura 35 horas a 27°C, el macho se abstiene de comer y se mantiene cerca del nido, moviéndolo y proporcionándole aireación con sus aletas.
- ***Eclosión y larvicultura*:** Después de las 35 horas de incubación, las larvas, con un tamaño de 2 mm y color blanco, comienzan a eclosionar. El macho puede

empezar a consumirlas, por lo que se retira y se reduce el nivel del agua. Las larvas no comen inicialmente, ya que poseen un saco vitelino que se reabsorbe en 48 horas. Después de este periodo, las larvas empiezan a alimentarse, y es importante mantener un nivel de agua reducido para facilitar su desarrollo.

2.1.5. Importancia en reproducción

En el caso específico de los Betta splendens, se busca llevar a cabo un proceso de reversión sexual con el fin de obtener un elevado número de machos que presenten características fenotípicas más atractivas que las hembras. En el mercado de peces ornamentales, la demanda se centra en ejemplares que exhiban aletas caudales amplias y colores llamativos, lo que resalta la importancia de identificar un protocolo adecuado para el uso de hormonas de reversión. Cabe destacar que la práctica de la reversión sexual con hormonas no se limita exclusivamente a los Betta splendens; de hecho, ha sido aplicada con éxito en otras especies acuícolas destinadas al consumo humano. Este enfoque, respaldado por la ciencia acuícola, busca optimizar la producción y obtener individuos con características específicas que satisfagan las demandas del mercado y contribuyan al desarrollo sostenible de la acuicultura. [4]

2.1.6. Problemática de su reproducción

Ecuador ofrece una variedad de condiciones climáticas adecuadas para la cría de peces ornamentales, siendo esta una actividad de gran importancia económica que genera empleo a nivel nacional e internacional [29]. Uno de los problemas más comunes asociados a esta actividad es el desequilibrio que provoca la constante presión sobre las

poblaciones, lo que conlleva a la alteración de los ecosistemas ya que muchas especies siguen siendo extraídas de sus hábitats naturales para su comercialización, contribuyendo así a la desaparición de especies nativas [47].

2.2. Factores o Variables para medir

En la reproducción controlada de peces Betta, el monitoreo preciso de ciertos parámetros ambientales resulta esencial para garantizar un entorno óptimo que promueva el éxito reproductivo. Entre los factores clave que influyen en este proceso se encuentran la temperatura, la salinidad y el pH del agua, cada uno con un impacto directo en la salud y el comportamiento de los peces. Por ejemplo, la temperatura regula la cantidad de oxígeno disuelto en el agua, mientras que la salinidad y el pH deben mantenerse en rangos específicos para evitar estrés y favorecer el bienestar de los ejemplares. En este subtema, se abordarán en detalle estos parámetros, su relación con la biología de los peces Betta y las estrategias empleadas para su medición y control mediante el sistema embebido desarrollado.

2.2.1. Temperatura

La temperatura juega un papel crucial en la salud y el bienestar de los peces, ya que afecta varios aspectos de su entorno acuático. Por ejemplo, influye en la cantidad de gases disueltos en el agua, como el oxígeno, y un aumento de la temperatura puede reducir la disponibilidad de oxígeno en el agua. La temperatura óptima del agua que va desde 27°C a 30°C para los peces Betta. Para mantener un ambiente acuático adecuado, se utilizan calentadores eléctricos con termostato, los cuales mantienen la temperatura del

agua constante. Es importante que estos calentadores estén siempre sumergidos para evitar daños en su funcionamiento [41].

2.2.2. Salinidad

La presencia de materiales sólidos disueltos en el agua es esencial para los peces, ya que necesitan ciertos elementos para llevar a cabo procesos bioquímicos vitales. La cantidad de sales disueltas en el agua afecta tanto la densidad del agua como los requisitos de temperatura de algunas especies, por lo que cualquier cambio en la salinidad al transferir peces de un hábitat a otro debe ser gradual. La alcalinidad del agua, medida por su capacidad de neutralización de ácido, es crucial para mantener un equilibrio químico adecuado. Componentes como bicarbonatos, carbonatos, boratos y fosfatos contribuyen a esta alcalinidad, que se expresa en miliequivalentes por litro (mEq / L). Mantener una alcalinidad adecuada garantiza la capacidad buffer para los metales ácidos y el correcto funcionamiento de los biofiltros, con rangos recomendados entre 0,2-10 mEq / L, y para agua de mar, 2,5 mEq / L[28].

2.2.3. pH

El nivel de acidez del agua es esencial para la salud de los peces, ya que estos pueden tolerar un rango de pH entre 5 y 11, siendo óptimo entre 6 y 9 pH. Para medir el pH del agua, existen dos métodos: uno colorimétrico, que utiliza indicadores químicos específicos para diferentes rangos de pH, y otro que emplea papel tornasol, una tira de papel tratada químicamente que cambia de color al sumergirse en el agua, proporcionando así una referencia visual para determinar el pH mediante una tabla de colores [20].

2.3. Inteligencia Artificial

La inteligencia artificial (IA) es una disciplina de la informática orientada al desarrollo de sistemas capaces de realizar tareas que tradicionalmente requieren inteligencia humana, tales como el reconocimiento de patrones, la toma de decisiones, el aprendizaje a partir de datos y la resolución de problemas complejos. Según Russell y Norvig, la IA puede definirse como el estudio de agentes capaces de percibir su entorno y actuar de manera racional para maximizar la probabilidad de alcanzar determinados objetivos. En los últimos años, la inteligencia artificial ha tenido una expansión significativa debido al aumento en la capacidad de procesamiento, la disponibilidad de grandes volúmenes de datos y el desarrollo de nuevos algoritmos de aprendizaje. Estas tecnologías han permitido su aplicación en áreas como visión artificial, robótica, análisis de datos, automatización industrial y sistemas de monitoreo inteligente. En el contexto de sistemas embebidos e Internet de las Cosas (IoT), la inteligencia artificial permite que los dispositivos analicen información del entorno y tomen decisiones de forma autónoma o asistida, optimizando procesos y mejorando la eficiencia de los sistemas tecnológicos.

2.3.1. Machine Learning

El aprendizaje automático o Machine Learning es una subdisciplina de la inteligencia artificial que se centra en el desarrollo de algoritmos capaces de aprender patrones a partir de datos sin necesidad de ser programados explícitamente para cada tarea. De acuerdo con Géron, el aprendizaje automático consiste en construir modelos matemáticos que permiten a un sistema mejorar su desempeño a medida que se expone a mayor cantidad de datos. Este enfoque se basa en la identificación de relaciones

estadísticas dentro de los conjuntos de datos para generar predicciones o clasificaciones. Los métodos de Machine Learning se utilizan ampliamente en aplicaciones como reconocimiento de voz, sistemas de recomendación, detección de fraudes y análisis de imágenes. En sistemas de monitoreo inteligente, estos algoritmos permiten procesar información proveniente de sensores o cámaras para identificar comportamientos, patrones o eventos relevantes dentro del entorno observado.

2.3.2. Deep Learning

El Deep Learning o aprendizaje profundo es una técnica avanzada dentro del Machine Learning que utiliza múltiples niveles de procesamiento para extraer representaciones jerárquicas de los datos. Este enfoque permite que los sistemas aprendan características complejas de manera automática, sin necesidad de realizar una extracción manual de características. Según Goodfellow, Bengio y Courville, el aprendizaje profundo ha permitido avances significativos en tareas como reconocimiento de imágenes, procesamiento de lenguaje natural y visión artificial. Gracias al uso de arquitecturas especializadas y al incremento de la capacidad de procesamiento de hardware moderno, los modelos de aprendizaje profundo han logrado alcanzar niveles de precisión superiores a los métodos tradicionales en tareas de clasificación y detección de objetos. En aplicaciones de visión artificial, estas técnicas permiten analizar imágenes o video en tiempo real para identificar objetos, comportamientos o patrones dentro de un entorno determinado.

2.3.3. Visión Artificial

La visión artificial es una rama de la inteligencia artificial que se enfoca en el desarrollo de sistemas capaces de interpretar información visual proveniente de imágenes o secuencias de video. Su objetivo principal es permitir que las máquinas puedan comprender el contenido visual del entorno de manera similar a la percepción humana. Según Szeliski, la visión artificial combina técnicas de procesamiento de imágenes, aprendizaje automático y análisis matemático para detectar características relevantes dentro de una imagen. Entre las aplicaciones más comunes se encuentran el reconocimiento facial, la inspección industrial automatizada, la navegación autónoma y el análisis de comportamiento en sistemas biológicos. En sistemas de monitoreo inteligente, la visión artificial permite identificar objetos específicos dentro de un entorno, facilitando la toma de decisiones automáticas basadas en la información visual capturada por sensores o cámaras.

2.3.4. Modelos de detección de objetos

Los modelos de detección de objetos permiten identificar y localizar múltiples objetos dentro de una imagen mediante cuadros delimitadores (bounding boxes). Estos modelos combinan técnicas de clasificación y localización para determinar la posición y categoría de los objetos presentes en una escena.

Entre los modelos más utilizados en visión artificial se encuentran:

- **R-CNN (Region-Based Convolutional Neural Network):** Este modelo fue uno de los primeros enfoques en utilizar redes neuronales convolucionales para detectar objetos en imágenes. Sin embargo, su principal desventaja es el alto costo computacional debido al procesamiento independiente de múltiples regiones de la imagen.
- **Faster R-CNN:** Es una mejora del modelo R-CNN que introduce una red de propuestas de regiones (Region Proposal Network), lo cual permite acelerar significativamente el proceso de detección.
- **SSD (Single Shot Detector):** Este modelo realiza detección de objetos en una sola pasada por la red neuronal, lo que mejora la velocidad de procesamiento en comparación con los modelos basados en regiones.
- **YOLO (You Only Look Once):** YOLO es uno de los modelos más eficientes para detección de objetos en tiempo real, ya que procesa la imagen completa en una única evaluación de la red neuronal.

Debido a su velocidad y precisión, YOLO ha sido ampliamente utilizado en aplicaciones de robótica, sistemas de vigilancia y análisis de video en tiempo real.

2.3.4.1. Método de detección YOLO

El modelo YOLO (You Only Look Once) corresponde a una arquitectura de detección de objetos basada en redes neuronales convolucionales que permite identificar y localizar múltiples objetos dentro de una imagen mediante un único proceso de

inferencia. A diferencia de los enfoques tradicionales de detección, en los cuales primero se generan regiones de interés y posteriormente se clasifican, YOLO formula el problema de detección como una regresión directa desde los píxeles de la imagen hacia las coordenadas de los objetos y sus probabilidades de clase. Este enfoque permite analizar la imagen completa en una sola pasada por la red neuronal, lo que mejora la eficiencia computacional y permite capturar el contexto global de la escena [46].

El funcionamiento del modelo se basa en dividir la imagen de entrada en una cuadrícula de tamaño $S \times S$. Cada celda de la cuadrícula es responsable de predecir uno o varios cuadros delimitadores (bounding boxes), los cuales representan la ubicación de los objetos detectados dentro de la imagen. Cada cuadro delimitador se define mediante un conjunto de parámetros que incluyen las coordenadas del centro del objeto (x,y), el ancho (w), la altura (h) y un valor de confianza asociado a la detección. Este valor de confianza indica la probabilidad de que el cuadro delimitador contenga un objeto y se calcula considerando la superposición entre la predicción del modelo y la posición real del objeto.

Una de las características principales del modelo YOLO es su capacidad para realizar detección en tiempo real, ya que todas las predicciones se generan en un único paso de la red neuronal. Este diseño permite que el modelo procese secuencias de video o imágenes de forma eficiente, alcanzando velocidades cercanas al tiempo real en sistemas de procesamiento general. Debido a estas características, YOLO se ha convertido en uno de los modelos más utilizados en aplicaciones de visión artificial donde se requiere una combinación entre precisión y velocidad de procesamiento [35].

Las versiones más recientes del modelo, entre ellas YOLOv11, incorporan mejoras en la arquitectura de la red que permiten optimizar el proceso de extracción de características y mejorar la precisión de detección. Estas mejoras incluyen optimizaciones en las capas convolucionales, mecanismos de regularización y mejoras en la normalización de los parámetros de la red. Como resultado, el modelo logra mejorar métricas de desempeño como la precisión media promedio (mAP), manteniendo al mismo tiempo una alta eficiencia computacional [51].YOLO es una arquitectura CNN que ha sido diseñada específicamente para abordar problemas de visión por computadora a altos niveles de rendimiento. Su estructura de red está diseñada específicamente para una detección de objetos rápida y eficaz en imágenes y fotogramas de vídeo. El enfoque de YOLO consiste en observar la imagen completa al hacer predicciones, lo que le permite capturar el contexto global de manera implícita, ayudando a modelar tanto el tamaño como la forma de los objetos, así como los detalles de su apariencia que dependen del contexto [35].

YOLO se construye utilizando una serie de capas convolucionales seguidas de capas completamente conectadas. En su versión inicial, contaba con un total de 24 capas convolucionales, seguidas por 2 capas completamente conectadas. La introducción de la normalización por lotes en todas las capas convolucionales con YOLOv11 mejoró la regularización del modelo, lo que resultó en un aumento de la precisión media promedio (mAP o mean Average Precision). Desde su primera versión hasta el lanzamiento de YOLOv11 y versiones posteriores, YOLO ha experimentado un desarrollo significativo, mejorando su regularización y rendimiento [46].

YOLO es capaz de detectar objetos en una imagen de manera única, prediciendo simultáneamente múltiples cuadros delimitadores que enmarcan los objetos y generando probabilidades condicionales por cada clase para cada uno de estos cuadros delimitadores. Esto permite una velocidad de ejecución de 45 fotogramas por segundo (fps) en computadoras de propósito general.

2.3.4.2. ***Capa Convolucional***

El proceso de detección en YOLO se fundamenta en el uso de capas convolucionales, las cuales permiten extraer características relevantes a partir de las imágenes de entrada. Estas capas aplican filtros o kernels que se desplazan sobre la imagen realizando operaciones matemáticas entre los valores del filtro y los píxeles de la imagen. Como resultado de esta operación se generan mapas de características, los cuales resaltan patrones importantes presentes en la imagen, tales como bordes, texturas o estructuras geométricas [10].

Matemáticamente, la operación de convolución donde $I(x,y)$ representa la imagen de entrada, $K(i,j)$ corresponde al kernel o filtro aplicado y $I*K$ representa el mapa de características generado, se presenta la Ec. 1:

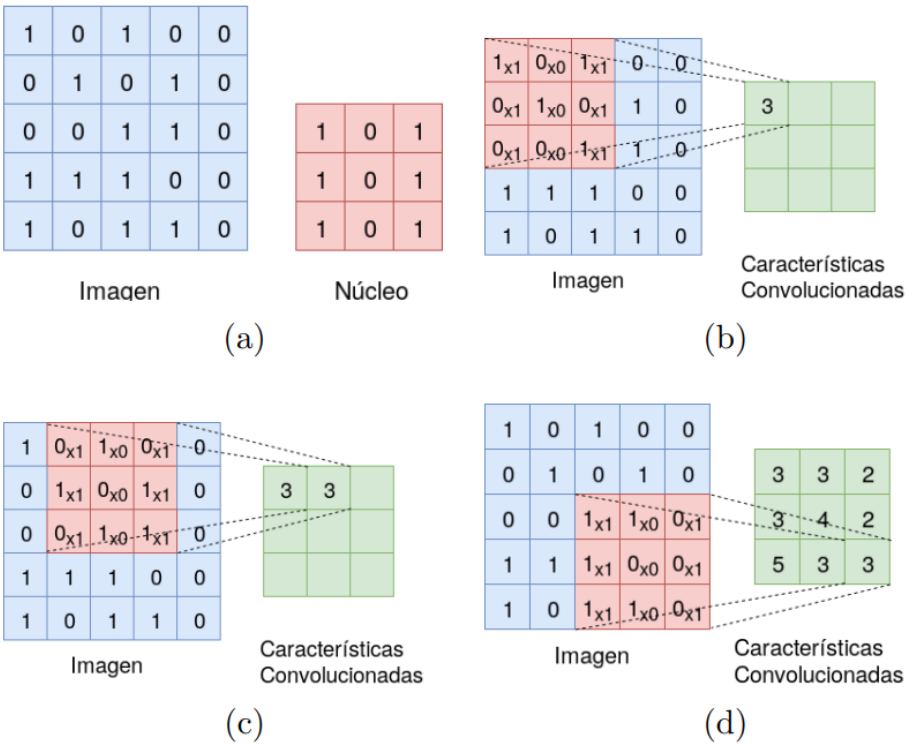
$$(I * K)(x, y) = \sum_{i=-m}^{m} \sum_{j=-n}^{n} I(x + i, y + j) \cdot K(i, j) \qquad \text{Ec. 1}$$

La Figura 4 ilustra el proceso de convolución aplicado sobre una matriz de imagen. En la figura se observa cómo un kernel se desplaza sobre la imagen realizando multiplicaciones elemento a elemento entre los valores del filtro y los píxeles de la región analizada. Posteriormente, los resultados obtenidos se suman para generar un nuevo valor

dentro del mapa de características. Este procedimiento se repite a lo largo de toda la imagen, permitiendo extraer información relevante que posteriormente será utilizada por las capas profundas de la red para identificar objetos dentro de la escena.

Figura 4

Operación convolución



Fuente: Tomado de Silva [50]

Como se observa en la Figura 4, la convolución se realiza desplazando un núcleo o kernel sobre la matriz de entrada. Para ejemplificar el procedimiento, se considera una imagen de 5×5y un kernel de 3×3. En la primera posición, el kernel se superpone sobre la región superior izquierda de la imagen y se realizan multiplicaciones elemento a

elemento entre ambas matrices. Posteriormente, los resultados se suman para obtener el primer valor del mapa de características. Por lo tanto, el primer valor generado en el mapa convolucionado es 3. Al repetir este procedimiento en todas las posiciones posibles del kernel sobre la imagen, sin aplicar relleno (padding) y con un desplazamiento (stride) de un píxel, se obtiene el siguiente mapa de características:

$$(1 \cdot 1) + (0 \cdot 0) + (1 \cdot 1) + (0 \cdot 1) + (1 \cdot 0) + (0 \cdot 1) + (0 \cdot 1) + (0 \cdot 0) + (1 \cdot 1) = 3$$

Este resultado permite comprender cómo una capa convolucional transforma una imagen de entrada en una representación más compacta, donde se resaltan patrones visuales relevantes. En el contexto del presente trabajo, este principio explica cómo el modelo YOLOv11 extrae características de los fotogramas del video para identificar elementos como el pez Betta macho, la hembra y la presencia de burbujas.

$$I * K = \begin{bmatrix} 3 & 3 & 2 \\ 3 & 4 & 2 \\ 5 & 3 & 3 \end{bmatrix}$$

Después de la etapa de convolución, las redes neuronales incorporan generalmente capas de reducción espacial, conocidas como pooling. Estas capas tienen como objetivo reducir la dimensionalidad de los mapas de características, conservando únicamente la información más representativa del patrón detectado. Esta reducción permite disminuir el costo computacional del modelo y mejorar su capacidad de generalización durante el entrenamiento.

A medida que la red neuronal aumenta su profundidad, las características detectadas se vuelven progresivamente más complejas. Las primeras capas suelen

identificar patrones simples como bordes o líneas, mientras que las capas más profundas combinan estas características para reconocer estructuras más complejas, lo que permite finalmente la identificación de objetos completos dentro de la imagen. Esta capacidad de aprendizaje jerárquico constituye una de las principales ventajas de las redes convolucionales en aplicaciones de visión artificial [12], [33].

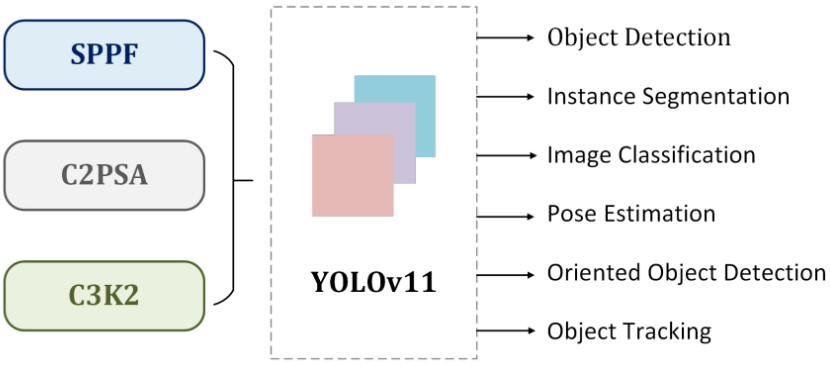
2.3.4.3. Arquitectura de YOLOv11.

Las versiones modernas de la familia YOLO han evolucionado significativamente con el objetivo de mejorar la precisión de detección y la eficiencia computacional en tareas de visión artificial. En particular, YOLOv11 introduce mejoras estructurales en su arquitectura interna que permiten optimizar el procesamiento de características visuales y la capacidad de generalización del modelo. Esta arquitectura se encuentra diseñada bajo un enfoque modular que integra diferentes bloques especializados encargados de la extracción, procesamiento y refinamiento de características dentro de la red.

Como se observa en la Figura 5, la arquitectura de YOLOv11 se compone principalmente de tres bloques fundamentales: SPPF, C2PSA y C3K2, los cuales trabajan de manera conjunta para mejorar la representación de características en diferentes escalas. Estos módulos permiten al modelo procesar información espacial y contextual de forma más eficiente, lo cual resulta especialmente útil en tareas de detección de objetos en tiempo real.

Figura 5

Módulos Arquitectónicos del YOLO11



Nota. Tomado de Khanam y Hussain [27].

El primer componente relevante es el módulo SPPF (Spatial Pyramid Pooling Fast). Este bloque tiene como objetivo principal ampliar el campo receptivo de la red sin incrementar significativamente el costo computacional. El SPPF realiza múltiples operaciones de pooling sobre los mapas de características generados por las capas anteriores, permitiendo que el modelo capture información contextual a diferentes escalas espaciales. De esta manera, el modelo puede reconocer objetos independientemente de su tamaño dentro de la imagen. Este mecanismo es especialmente importante en aplicaciones donde los objetos de interés pueden aparecer en diferentes posiciones o escalas dentro del entorno visual.

El segundo componente de la arquitectura corresponde al módulo C2PSA, el cual introduce mecanismos de atención dentro del procesamiento de características. Este bloque permite que la red priorice aquellas regiones de la imagen que contienen información relevante para la tarea de detección. En términos generales, los mecanismos de atención funcionan asignando diferentes pesos a las características extraídas, de modo que el modelo pueda enfocarse en patrones visuales más representativos y reducir el

impacto de ruido o información irrelevante presente en la imagen. Esta estrategia mejora la precisión del modelo al momento de identificar objetos específicos dentro de escenarios complejos.

El tercer módulo fundamental es C3K2, el cual corresponde a una estructura optimizada para la extracción de características profundas dentro de la red. Este bloque permite mejorar la propagación de la información a través de las capas del modelo, facilitando el aprendizaje de representaciones más robustas y discriminativas. La estructura C3K2 combina operaciones de convolución con conexiones internas que favorecen la reutilización de características previamente aprendidas, lo que contribuye a reducir la pérdida de información durante el proceso de entrenamiento. Gracias a este mecanismo, el modelo logra una mayor eficiencia en la detección de objetos manteniendo un equilibrio entre precisión y velocidad de procesamiento.

Una vez que los mapas de características han sido procesados por estos módulos, el modelo YOLOv11 puede realizar diversas tareas dentro del campo de la visión artificial. Tal como se muestra en la Figura 3, la arquitectura permite abordar múltiples problemas relacionados con el análisis de imágenes, entre los cuales se incluyen:

- Detección de objetos (Object Detection)
- Segmentación de instancias (Instance Segmentation)
- Clasificación de imágenes (Image Classification)
- Estimación de pose (Pose Estimation)
- Detección orientada de objetos (Oriented Object Detection)

- Seguimiento de objetos (Object Tracking)

Estas capacidades convierten a YOLOv11 en una herramienta versátil para aplicaciones que requieren análisis visual en tiempo real. En particular, su eficiencia computacional permite su implementación en plataformas de hardware con recursos limitados, como sistemas embebidos o dispositivos utilizados en aplicaciones IoT. En el contexto del presente trabajo, el modelo YOLOv11 se utiliza específicamente para la detección de objetos dentro del sistema de monitoreo de comportamiento de peces Betta, permitiendo identificar patrones visuales relevantes como la presencia del macho, la hembra y la formación de burbujas dentro del acuario.

3. Capítulo III. Metodología

El presente capítulo describe el desarrollo metodológico experimental del sistema embebido propuesto para la optimización de la reproducción de los peces Betta splendens mediante técnicas de aprendizaje profundo. El proceso de ingeniería se estructura bajo un modelo secuencial en cascada, el cual permite organizar el proyecto en fases claramente definidas y ejecutadas de manera progresiva, garantizando coherencia técnica y trazabilidad en cada etapa.

3.1. Metodología de desarrollo

El desarrollo del sistema embebido propuesto se estructuró bajo un enfoque metodológico formal, orientado a garantizar organización, trazabilidad y coherencia técnica durante todo el ciclo de vida del proyecto. Considerando la naturaleza experimental del prototipo y la claridad de los objetivos planteados, se adoptó un modelo de desarrollo secuencial que permite dividir el proceso en fases claramente delimitadas. Este enfoque facilita la documentación estructurada del sistema y asegura que cada etapa contribuya progresivamente al cumplimiento de los objetivos de la investigación.

3.1.1. Modelo en cascada adoptado

El modelo de desarrollo seleccionado corresponde al ciclo de vida en cascada, el cual organiza el proceso de ingeniería en fases secuenciales: análisis, especificación de requerimientos, diseño, implementación, verificación, validación y pruebas. Este modelo resulta adecuado para sistemas embebidos experimentales, donde los requerimientos

pueden definirse desde etapas tempranas y el control estructurado del desarrollo minimiza riesgos técnicos.

La adopción del modelo en cascada fue complementada con la alineación a estándares internacionales IEEE, permitiendo que cada fase del proceso se respalde en lineamientos formales de ingeniería. La fase de requerimientos se estructuró conforme a IEEE 29148 [24][2], la definición arquitectónica se apoyó en principios de IEEE 42010 [25][3], el diseño del sistema se alineó con IEEE 1016 [22][4], mientras que las actividades de verificación y validación se fundamentaron en IEEE 1012 [21] [5]y la planificación de pruebas en IEEE 29119 [23][6]. Esta integración metodológica permitió desarrollar el prototipo bajo criterios técnicos rigurosos y verificables.

3.2. Especificación de requerimientos

La especificación de requerimientos constituye la base estructural del desarrollo del sistema, ya que define de manera formal las funcionalidades esperadas, los atributos de calidad y las restricciones técnicas del prototipo experimental. Esta fase se desarrolló

[2] IEEE/ISO/IEC 29148: estándar que establece lineamientos para la ingeniería de requisitos en sistemas y software, incluyendo su definición, análisis, documentación y gestión (IEEE Standards Association, 2018).

[3] IEEE/ISO/IEC 42010: estándar que define cómo estructurar y expresar la descripción arquitectónica de sistemas, software y otros tipos de entidades tecnológicas (IEEE Standards Association, 2022).

[4] IEEE 1016: estándar que establece el contenido y la organización de una descripción de diseño de software, usada para documentar y comunicar decisiones de diseño (IEEE Standards Association, 2009).

[5] IEEE 1012: estándar orientado a los procesos de verificación y validación de sistemas, software y hardware durante su ciclo de vida (IEEE Standards Association, 2024).

[6] IEEE/ISO/IEC 29119: familia de estándares que proporciona conceptos, procesos, documentación y técnicas para la planificación y ejecución de pruebas de software y sistemas (IEEE Standards Association, 2021).

bajo principios de requerimientos alineados con estándares IEEE, asegurando claridad, consistencia y trazabilidad durante las etapas posteriores de diseño e implementación.

3.2.1. Nomenclatura de requerimientos

Con el fin de asegurar claridad, trazabilidad y control durante el desarrollo del sistema, los requerimientos fueron clasificados y codificados conforme a una nomenclatura estructurada. Esta codificación permite identificar de manera inequívoca el tipo de requerimiento y facilita su verificación en etapas posteriores del ciclo de vida, tal y como se muestra en la Tabla 1.

Tabla 1

Abreviatura de los requerimientos

Abreviatura	Tipo de requerimiento
StSR	Requerimientos de Stakeholders
SySR	Requerimientos del Sistema
ArSR	Requerimientos de Arquitectura

3.2.2. Requerimientos de los Stakeholders

En el presente proyecto no existe un cliente externo ni un usuario comercial, dado que el sistema corresponde a un prototipo experimental desarrollado con fines académicos. No obstante, desde el enfoque de ingeniería de requerimientos (IEEE 29148 [24]), se consideran stakeholders aquellos actores o elementos que influyen directa o indirectamente en la definición y operación del sistema.

En este contexto, los stakeholders identificados son:

- El investigador-desarrollador.

- El entorno experimental de laboratorio.
- La infraestructura tecnológica de almacenamiento en la nube.
- El entorno físico donde se instala el acuario.

A partir de esta identificación se establecieron los siguientes requerimientos, se menciona la Tabla 2, para la implementación del sistema.

Tabla 2

Requerimientos de Stakeholders (StSR)

Requerimientos Operacionales		
#	**Requerimiento**	**Prioridad**
StSR1	Disponibilidad de suministro eléctrico estable para Raspberry Pi 4 y ESP32	Alta
StSR2	Disponibilidad de red WiFi para transmisión de datos	Alta
StSR3	Espacio físico adecuado para instalación del acuario experimental	Alta
StSR4	Condiciones de iluminación controladas para captura de video	Alta
Requerimientos de Monitoreo Experimental		
StSR5	Visualización en tiempo real del procesamiento de video	Alta
StSR6	Registro histórico de eventos de detección	Alta
StSR7	Acceso remoto a parámetros ambientales almacenados	Media
StSR8	Generación de notificaciones ante eventos reproductivos	Media

3.2.3. Requerimientos del sistema

Los requerimientos del sistema describen las capacidades funcionales y operativas que el prototipo debe cumplir para alcanzar el objetivo de monitoreo inteligente del comportamiento reproductivo de peces Betta, tal como se presenta en la Tabla 3.

Tabla 3

Requerimientos generales

Requerimientos de uso		
#	**Requerimiento**	**Prioridad**
SySR1	El sistema debe capturar video en tiempo real mediante cámara conectada a la Raspberry Pi	Alta
SySR2	El modelo YOLOv11 debe ejecutarse automáticamente al iniciar el sistema	Alta
SySR3	El sistema debe detectar pez Betta macho, pez Betta hembra y nidos de burbujas	Alta
SySR4	El sistema debe registrar eventos de detección en la base de datos en la nube	Alta
Requerimientos de Interfaces		
SySR5	Disponer de interfaz de cámara compatible con Raspberry Pi	Alta
SySR6	Disponer de interfaz WiFi para transmisión de datos	Alta
SySR7	Disponer de puertos GPIO para conexión de sensores	Alta
Requerimientos de Estados		
SySR8	La Raspberry Pi debe estar operativa para ejecutar el modelo	Alta
SySR9	El ESP32 debe estar activo para adquisición sensorial	Alta
SySR10	El sistema debe operar de manera continua durante el monitoreo	Media
Requerimientos Físicos		
SySR11	El sistema debe instalarse en entorno de laboratorio controlado	Alta
SySR12	El sistema debe ser integrable en un acuario estándar	Media

3.2.4. Requerimientos de arquitectura

Los requerimientos de arquitectura definen las condiciones estructurales y tecnológicas necesarias para la correcta implementación del sistema embebido, en la que se representa los requerimientos de la arquitectura en la Tabla 4.

Tabla 4

Requerimientos de Arquitectura (ArSR)

Requerimientos Lógicos		
#	**Requerimiento**	**Prioridad**
ArSR1	Separación modular entre procesamiento de visión y adquisición sensorial	Alta
ArSR2	Comunicación inalámbrica hacia base de datos en la nube	Alta
Requerimientos de Hardware		
ArSR3	Uso como unidad de procesamiento para inferencia	Alta
ArSR4	Uso de microchip como nodo de adquisición análoga y digital	Alta
ArSR5	Integración con sensores externos	Alta
ArSR6	Costo	Medio
Requerimientos de Software		
ArSR7	Compatibilidad con sistema operativo GNU/Linux	Alta
ArSR8	Un solo lenguaje principal de desarrollo en un solo ambiente de programación	Alta
ArSR9	Integración con Firebase Realtime Database	Alta
ArSR10	El lenguaje seleccionado debe ser eficiente en consumo de recursos en hardware embebido	Alta
ArSR11	El entorno debe permitir ejecución en tiempo real o cercana al tiempo real	Alta
ArSR12	Las herramientas utilizadas deben ser accesibles sin licencias comerciales restrictivas	Media
Requerimientos Eléctricos		
ArSR13	Fuente estable de 5V para Raspberry Pi	Alta
ArSR14	Alimentación adecuada para sensores y microcontrolador	Alta

3.3. Selección del hardware

La selección del hardware del sistema se realizó tomando como base los requerimientos de arquitectura definidos previamente (ArSR3–ArSR6), los cuales establecen las condiciones mínimas que deben cumplir los dispositivos físicos para garantizar el correcto funcionamiento del prototipo experimental. Estos requerimientos incluyen la capacidad de procesamiento para inferencia de modelos de deep learning, la adquisición sensorial analógica y digital, la integración con sensores externos y la viabilidad económica del sistema.

3.3.1. Evaluación comparativa del hardware

La Tabla 5 presenta la comparación de las plataformas candidatas en función del cumplimiento de los requerimientos ArSR3–ArSR6.

Tabla 5

Comparación de plataformas hardware basada en requerimientos de arquitectura

Requerimiento	Raspberry Pi 4	Raspberry Pi 5	Jetson Nano	Orange Pi 5	BeagleBone
ArSR3	1	1	1	1	0
ArSR4	0	0	0	0	0
ArSR5	1	1	0	1	1
ArSR6	1	0	0	0	1
Total de cumplimiento	**3**	**2**	**1**	**2**	**2**
Criterio de evaluación: 0 = No cumple 1 = Cumple					

Dado que el sistema requiere simultáneamente procesamiento de visión artificial y adquisición sensorial, se adoptó una arquitectura híbrida compuesta por:

- Raspberry Pi 4 como unidad principal de procesamiento para la ejecución del modelo YOLOv11.
- ESP32 como nodo independiente de adquisición sensorial encargado de la lectura de los sensores DS18B20, pH y TDS.

Esta arquitectura permite desacoplar el procesamiento intensivo de la adquisición de datos ambientales, mejorando la estabilidad operativa del sistema. Mientras la Raspberry Pi 4 ejecuta la inferencia del modelo de detección en tiempo real, el ESP32 realiza la captura de datos analógicos y digitales sin afectar el rendimiento del procesamiento de imágenes.

3.4. Selección de Software

La selección de las tecnologías de software se realizó tomando como base los requerimientos de arquitectura definidos en la sección anterior (ArSR7–ArSR12). Estos requerimientos establecen las condiciones mínimas que debe cumplir el entorno de desarrollo para garantizar compatibilidad con el sistema embebido, eficiencia operativa y viabilidad en un contexto académico experimental.

3.4.1. Lenguaje de programación

Se consideraron como alternativas principales los lenguajes Python y Matlab, debido a su uso frecuente en aplicaciones de visión artificial y procesamiento de imágenes.

La Tabla 6 presenta la comparación de los lenguajes evaluados en función del cumplimiento de los requerimientos de software establecidos (ArSR7–ArSR12).

Tabla 6

Comparación de lenguajes de programación basada en los requerimientos de software

Requerimiento	Descripción	Python	Matlab
ArSR7	Compatibilidad con sistema operativo GNU/Linux	1	0
ArSR8	Un solo lenguaje principal de desarrollo en un solo ambiente de programación	1	1
ArSR9	Integración con Firebase Realtime Database	1	0
ArSR10	Eficiencia en consumo de recursos en hardware embebido	1	0
ArSR11	Ejecución en tiempo real o cercana al tiempo real	1	0
ArSR12	Accesibilidad sin licencias comerciales restrictivas	1	0
Total de cumplimiento		**6**	**1**

Criterio de evaluación:
0 = No cumple
1 = Cumple

Como se observa en la Tabla 6, Python presenta un mayor grado de cumplimiento respecto a los requerimientos definidos. En particular, destaca su compatibilidad con entornos GNU/Linux (Raspberry Pi OS), su amplio ecosistema de librerías para visión artificial y deep learning (OpenCV, Ultralytics, PyTorch), su facilidad de integración con Firebase y su naturaleza de software libre.

3.4.2. Algoritmos de deep learning

Una vez definido el lenguaje de programación y el entorno de desarrollo, se procede a seleccionar el modelo de detección de objetos que sería implementado en el sistema embebido. La elección del modelo debía cumplir con los requerimientos de software establecidos (ArSR7–ArSR12), especialmente en lo referente a eficiencia computacional, compatibilidad con Python, ejecución en tiempo real y funcionamiento en hardware de recursos limitados como Raspberry Pi 4.

Como se observa en la Tabla 7, el modelo YOLOv11 presenta el mayor grado de cumplimiento respecto a los requerimientos definidos para el sistema embebido. Este modelo destaca por su arquitectura de detección en una sola etapa (single-stage detector), lo cual permite realizar inferencia en tiempo real con menor latencia en comparación con modelos de dos etapas como Faster R-CNN.

Tabla 7

Comparación de modelos de detección en función de los requerimientos del sistema

Requerimiento	YOLOv11	Faster R-CNN	MobileNet-SSD	EfficientDet-Lite0

ArSR7	1	1	1	1
ArSR8	1	1	0	0
ArSR9	0	0	1	1
ArSR10	1	0	1	1
ArSR11	1	0	0	0
ArSR12	1	1	1	1
Total de cumplimiento	**5**	**3**	**4**	**4**

Criterio de evaluación:
0 = No cumple
1 = Cumple

Adicionalmente, YOLOv11 ofrece una adecuada relación entre precisión y velocidad de inferencia, aspecto fundamental para aplicaciones en Raspberry Pi 4, donde los recursos computacionales son limitados. Su compatibilidad con Python y el ecosistema Ultralytics facilita la integración con el flujo de entrenamiento en Google Colab y posterior despliegue en el entorno embebido.

3.5. Diseño del sistema

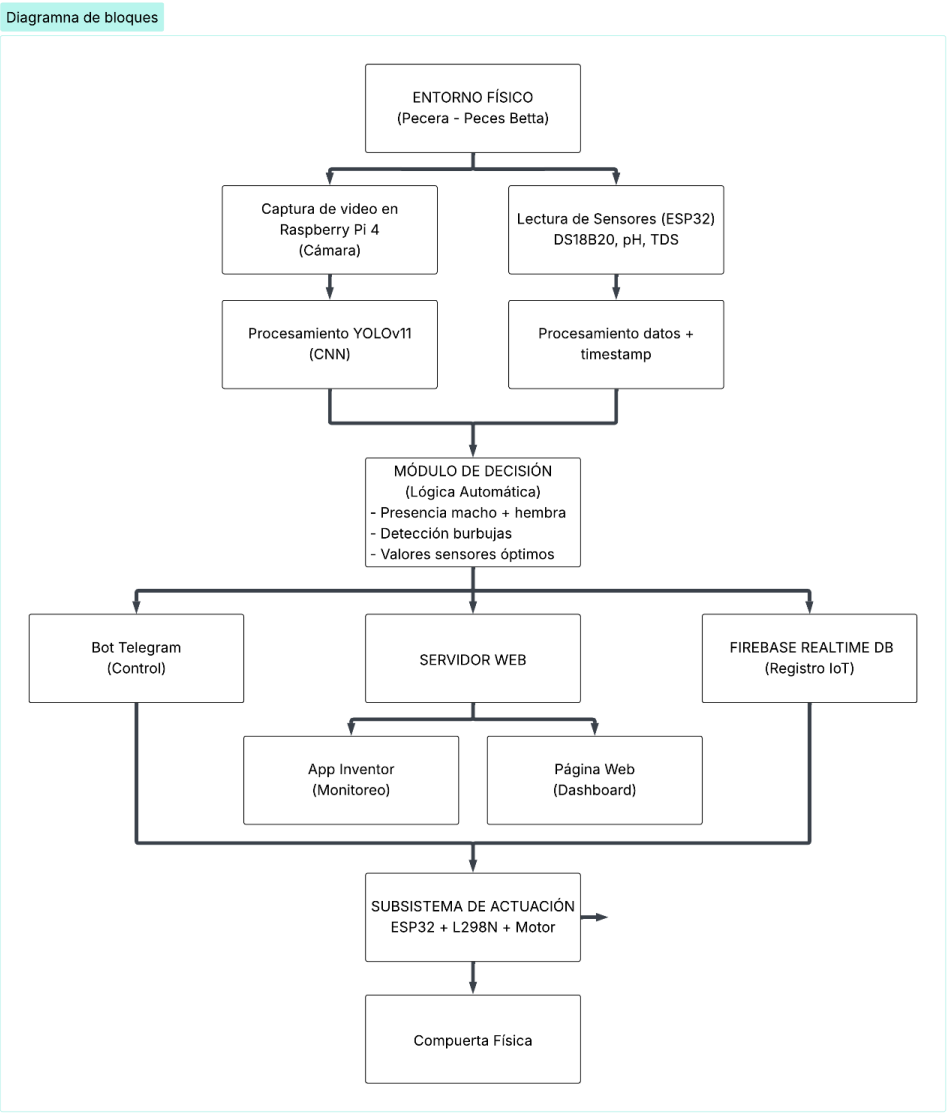
El diseño del sistema embebido se desarrolló a partir de los requerimientos definidos en la fase de arquitectura, integrando los subsistemas de procesamiento de visión artificial y adquisición sensorial bajo un enfoque modular distribuido. Este diseño busca garantizar separación funcional, estabilidad operativa y facilidad de mantenimiento, permitiendo que cada componente cumpla una función específica dentro del prototipo experimental.

3.5.1. Diagrama de bloques del sistema

El diagrama de bloques representa la organización general del sistema y la relación funcional entre sus componentes principales. Como se observa en la Figura 6, el sistema está dividido en dos subsistemas principales: el subsistema de procesamiento de visión artificial y el subsistema de adquisición sensorial.

Figura 6

Diagrama de bloques del sistema embebido



3.5.2. Diagrama de arquitectura

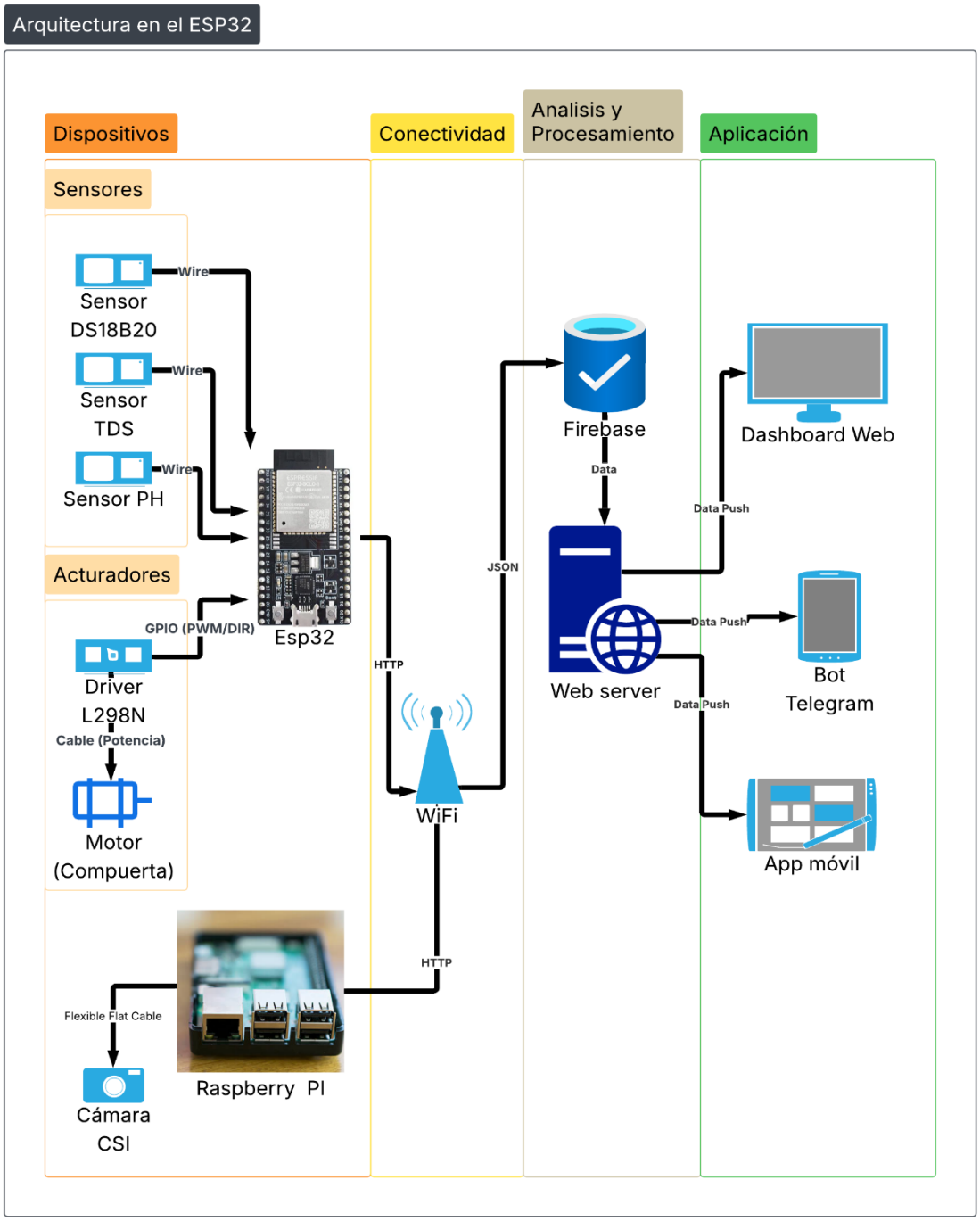
El diagrama de arquitectura muestra la interacción lógica entre los componentes del sistema y los servicios externos. Como se observa en la Figura 7, la arquitectura adoptada corresponde a un modelo distribuido, en el cual las tareas de procesamiento intensivo y adquisición sensorial se encuentran desacopladas.

La Raspberry Pi 4 ejecuta el modelo de detección en tiempo real, generando eventos de avistamiento. Paralelamente, el ESP32 realiza la adquisición periódica de datos ambientales, el control de la compuerta y los transmite hacia la base de datos Firebase. Esta arquitectura permite reducir la carga computacional sobre un único dispositivo y mejora la estabilidad general del sistema.

El uso de Firebase Realtime Database como intermediario permite centralizar la información y facilitar la visualización remota desde dispositivos finales.

Figura 7

Arquitectura del ESP32



3.5.3. Diagrama de flujo del sistema

El diagrama de flujo describe el funcionamiento dinámico del sistema embebido, desde la captura de información visual y ambiental hasta el almacenamiento y visualización remota de los datos. Como se observa en la Figura 8, el proceso inicia en la Raspberry Pi 4 con la activación de la cámara y la carga del modelo de inteligencia artificial YOLOv11. Posteriormente, el sistema captura fotogramas del video en tiempo real y ejecuta el reconocimiento de las clases definidas, correspondientes al pez Betta macho, la hembra y las burbujas. Cuando el algoritmo identifica la presencia simultánea del macho y la hembra en condición de proximidad, registra un avistamiento, almacena la imagen del evento y publica el conteo correspondiente en Firebase.

De forma paralela, el ESP32 inicializa la conexión WiFi, los sensores y el módulo de control L298N. Luego, realiza la lectura periódica de los parámetros fisicoquímicos del agua, incluyendo pH, TDS y temperatura, junto con una marca temporal. Estos datos son enviados a Firebase en intervalos definidos, permitiendo su consulta desde la aplicación, el servidor web o Telegram. Además, el algoritmo verifica el estado de la compuerta y evalúa si las condiciones ambientales se encuentran dentro de los rangos establecidos; en caso afirmativo, ejecuta la apertura de la compuerta o responde a una orden remota enviada desde la aplicación. Este procedimiento se resume en el Algoritmo 1, el cual representa la lógica principal implementada en el sistema.

Algoritmo 1. Funcionamiento general del sistema embebido

```
Inicio

Inicializar cámara en Raspberry Pi.
Cargar modelo YOLOv11.
Inicializar sensores, WiFi, módulo L298N y Firebase.
```

Mientras el sistema esté activo:

Capturar video en tiempo real.
Extraer fotograma actual.
Procesar fotograma con YOLOv11.

Si se detectan macho y hembra:
Calcular distancia entre ambos peces.
Calcular separación horizontal entre cajas.

Si los peces cumplen la condición de cercanía:
Registrar avistamiento.
Guardar imagen del evento.
Actualizar contador en Firebase.

Si se detectan burbujas cerca del macho:
Registrar presencia de burbujas.
Actualizar Firebase.

Leer temperatura, pH y TDS.

Si transcurrieron 5 minutos:
Enviar lecturas de sensores a Firebase.

Consultar estado de la compuerta en Firebase.

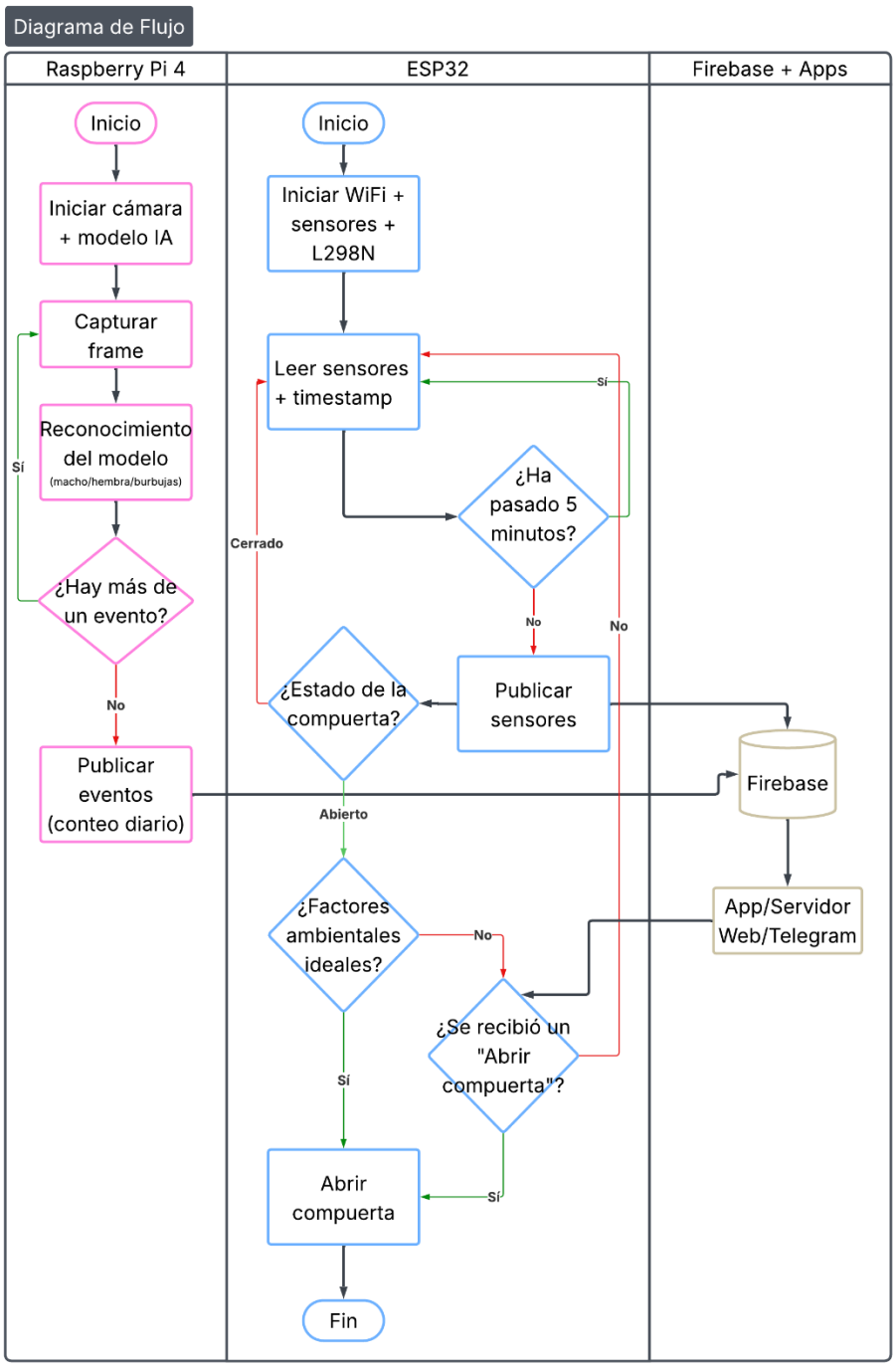
Si los parámetros ambientales son adecuados:
Abrir compuerta.
Actualizar estado en Firebase.

Si se recibe una orden remota desde la aplicación:
Ejecutar acción solicitada.
Actualizar estado en Firebase.

Fin

Figura 8

Diagrama de Flujo



3.5.4. Diagrama de conexión

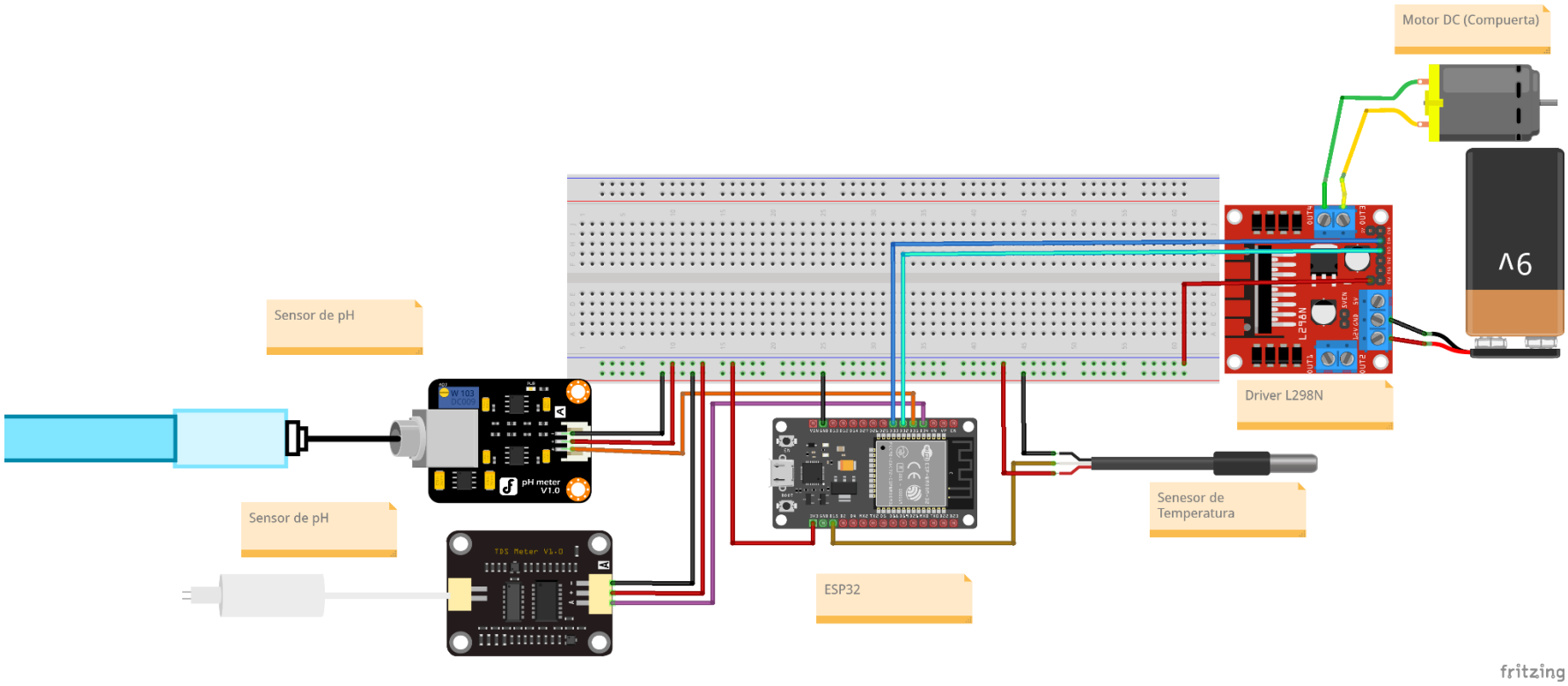
El diagrama de conexiones eléctricas representa la interconexión física entre el microcontrolador ESP32, los sensores encargados del monitoreo de las variables fisicoquímicas y el sistema de accionamiento. Como se observa en la Figura 9, el sensor de temperatura DS18B20 utiliza una interfaz digital de un solo hilo, mientras que los sensores de pH y TDS se conectan a las entradas analógicas del ESP32 para realizar la adquisición de sus respectivas mediciones.

Además, en la Figura 9 se puede observar la conexión del driver L298N, utilizado para controlar el motor encargado del accionamiento de la compuerta. Este módulo permite separar la etapa de control del ESP32 de la alimentación requerida por el actuador, proporcionando una conexión adecuada para su funcionamiento.

La distribución de alimentación y conexiones mostrada permite integrar los diferentes componentes del sistema de manera organizada, facilitando tanto la adquisición de datos ambientales como el control del mecanismo de apertura y cierre de la compuerta.

Figura 9

Diagrama de Conexiones



3.5.5. Desarrollo del Software

El desarrollo del software del sistema embebido se llevó a cabo siguiendo un enfoque estructurado conforme al modelo en cascada adoptado en esta investigación. Esta etapa comprende la consolidación de los conjuntos de datos, el entrenamiento supervisado del modelo de detección basado en YOLOv11, la generación de los pesos finales del modelo y su posterior despliegue en la Raspberry Pi 4 para ejecución en tiempo real.

3.5.5.1. Desarrollo del sistema de visión artificial

El desarrollo del sistema de visión artificial basado en aprendizaje profundo para la detección de objetos relevantes en imágenes. Se justifica la selección del modelo de red neuronal, se describen los conjuntos de datos utilizados —incluyendo su origen y configuración—, así como las etapas de entrenamiento del modelo de detección.

3.5.5.2. Selección del modelo YOLO11

YOLO11 permite no solo la detección de objetos, sino que también soporta tareas avanzadas de clasificación y segmentación, lo que lo hace versátil para visión artificial más allá de la detección tradicional. Además, al estar integrado tanto en Ultralytics como en plataformas de entrenamiento personalizadas como Roboflow, la gestión de datasets y la adaptación del modelo al problema específico resulta más sencilla y robusta.

3.5.5.3. Construcción y descripción de los conjuntos de datos

Para entrenar el modelo YOLO11 se emplearon tres conjuntos de datos personalizados disponibles en Roboflow Universe, cada uno correspondientes a una clase

de objeto objetivo. Estos datasets fueron recopilados y etiquetados por diferentes autores de la comunidad Roboflow, y se seleccionaron por su relevancia para el problema a abordar, así como por contar con un número adecuado de imágenes y anotaciones para entrenamiento supervisado.

Los datasets utilizados fueron:

1. Burbujas pequeñas de peces betta Computer Vision Model:

Dataset centrado en la detección de burbujas del pez macho, recuperado de: https://universe.roboflow.com/personas-lk0j1/burbujas-pequenas-de-peces-betta

2. Betta Fish Classification (hembra):

Contiene imágenes de peces betta hembra con anotaciones correspondientes, obtenido de: https://universe.roboflow.com/object-detection-egkf0/betta-fish-classification

3. Pez Betta Splender (macho):

Incluye imágenes de peces betta macho etiquetados para detección, descargado de: https://universe.roboflow.com/andres-qtcjl/pez-betta-splender

Cada uno de estos datasets fue importado a la plataforma Roboflow donde se verificaron y unificaron las anotaciones en formato compatible con YOLO (archivos .txt). Además, se procedió a balancear las clases y mejorar la diversidad de imágenes mediante técnicas de data aumentación integradas en la plataforma, como rotación, escalado, y cambio de brillo, con el fin de fortalecer el aprendizaje del modelo.

A continuación, se detalla la configuración de los datasets empleada para el entrenamiento:

Tabla 8

Tablas de los dataset

Dataset	Número de imágenes	Clases	Resolución final
Burbujas pequeñas de peces betta	1214 imágenes	burbujas	640×640 px
Betta Fish Classification	5000 imágenes	hembra	640×640 px
Pez Betta Splender	504 imágenes	macho	640×640 px

La elección de esta resolución responde a que modelos como YOLO11 suelen asumir múltiples escalas y tamaños de entrada para lograr un alto rendimiento en detección sin incrementar excesivamente el tiempo de entrenamiento o la complejidad computacional.

4. IMPLEMENTACIÓN Y RESULTADOS

En este capítulo se recopila la parte de la implementación y el desarrollo del sistema embebido, dando los resultados de los capítulos anteriores. En esta sección se describen la integración del hardware y el software, con el ajuste con pruebas en tiempo real, con lo cual se demuestra el rendimiento del sistema embebido para satisfacer las necesidades planteadas.

4.1. Implementación del Sistema

La implementación del sistema embebido desarrollado para la optimización de la reproducción de los peces Betta splendens, así como los resultados obtenidos durante su operación en el entorno experimental. En contraste con el capítulo anterior, donde se describieron la metodología de desarrollo, el diseño del sistema, la selección del hardware y software, y el entrenamiento del modelo de visión artificial, en este apartado se expone la integración física y lógica del prototipo, su puesta en funcionamiento en tiempo real y el análisis de los resultados generados durante su operación prolongada.

El sistema implementado integra una arquitectura de monitoreo inteligente compuesta por captura de video en tiempo real, procesamiento local mediante una Raspberry Pi, adquisición de parámetros fisicoquímicos mediante sensores, almacenamiento en la nube mediante Firebase Realtime Database y visualización de resultados en una interfaz de monitoreo. Esta integración puede apreciarse en la Figura 10, donde se presenta la implementación general del sistema propuesto. A su vez, la distribución específica de sus componentes puede detallarse en la Figura 10, en la que se

identifica la disposición del acuario, la cámara, la Raspberry Pi, el ESP32 y los sensores utilizados para el monitoreo ambiental.

Figura 10

Pecera junto al sistema embebido



Desde el punto de vista funcional, el sistema fue diseñado para trabajar de forma continua. La cámara captura video en tiempo real dentro del acuario experimental y los fotogramas son procesados por el modelo YOLOv11 previamente desarrollado. Cuando el sistema detecta simultáneamente al macho y a la hembra y verifica una condición de proximidad entre ambos, genera un evento de avistamiento y almacena la imagen correspondiente. Sobre dichas imágenes almacenadas se ejecuta posteriormente una

verificación de burbujas, la cual complementa el análisis del comportamiento reproductivo. Paralelamente, el sistema registra valores de temperatura, pH y TDS, permitiendo relacionar los eventos visuales con las condiciones ambientales del agua.

4.1.1. Implementación del subsistema de sensado ambiental

El subsistema de sensado ambiental fue implementado con el propósito de registrar variables fisicoquímicas relevantes dentro del acuario experimental, específicamente temperatura, pH y sólidos disueltos totales. Estas variables fueron seleccionadas por su influencia directa sobre la estabilidad del entorno acuático y su relación con el comportamiento de los peces Betta splendens durante el monitoreo.

La lectura de temperatura se realizó directamente desde la Raspberry Pi mediante el sensor correspondiente, mientras que los valores de pH y TDS fueron adquiridos a través de un ESP32 encargado de muestrear las señales analógicas y enviarlas al sistema principal. Esta lógica de adquisición puede relacionarse con el desarrollo previo al sistema, donde se define que el procesamiento embebido se realiza en Raspberry Pi y que pH y TDS son recibidos desde un ESP32, mientras que la temperatura se obtiene localmente.

Durante la implementación, las lecturas fueron actualizadas periódicamente y almacenadas en Firebase bajo nodos diferenciados entre valores actuales e históricos. La estructura de este almacenamiento se muestra en la Figura 29 al 34 en los Anexos, donde se observan los nodos correspondientes a temperatura, pH, TDS y estado de compuerta. Esta organización permitió conservar un historial del comportamiento ambiental del

sistema durante meses de operación, evitando limitar el monitoreo a simples lecturas instantáneas.

4.1.2. Implementación del subsistema de visión artificial

El subsistema de visión artificial fue implementado para trabajar sobre video en tiempo real utilizando el modelo YOLOv11 entrenado previamente en el Capítulo III. Para obtener dicho modelo, se empleó un conjunto de imágenes del pez Betta splendens y de burbujas provenientes de Roboflow, las cuales fueron etiquetadas mediante cajas delimitadoras para las clases de interés: macho, hembra y burbujas. Posteriormente, el dataset fue organizado en subconjuntos de entrenamiento, validación y prueba, permitiendo que el modelo aprendiera las características visuales de cada clase antes de ser integrado en el sistema embebido.

La relación entre el dataset utilizado y el despliegue final del modelo se resume en la Tabla 9. Esta tabla permite evidenciar que los resultados visuales obtenidos en tiempo real no corresponden a una detección manual, sino al uso de un modelo entrenado previamente con imágenes etiquetadas y posteriormente incorporado en la Raspberry Pi.

Tabla 9

Uso del dataset para el entrenamiento y despliegue del modelo YOLOv11

Etapa	Descripción	Resultado

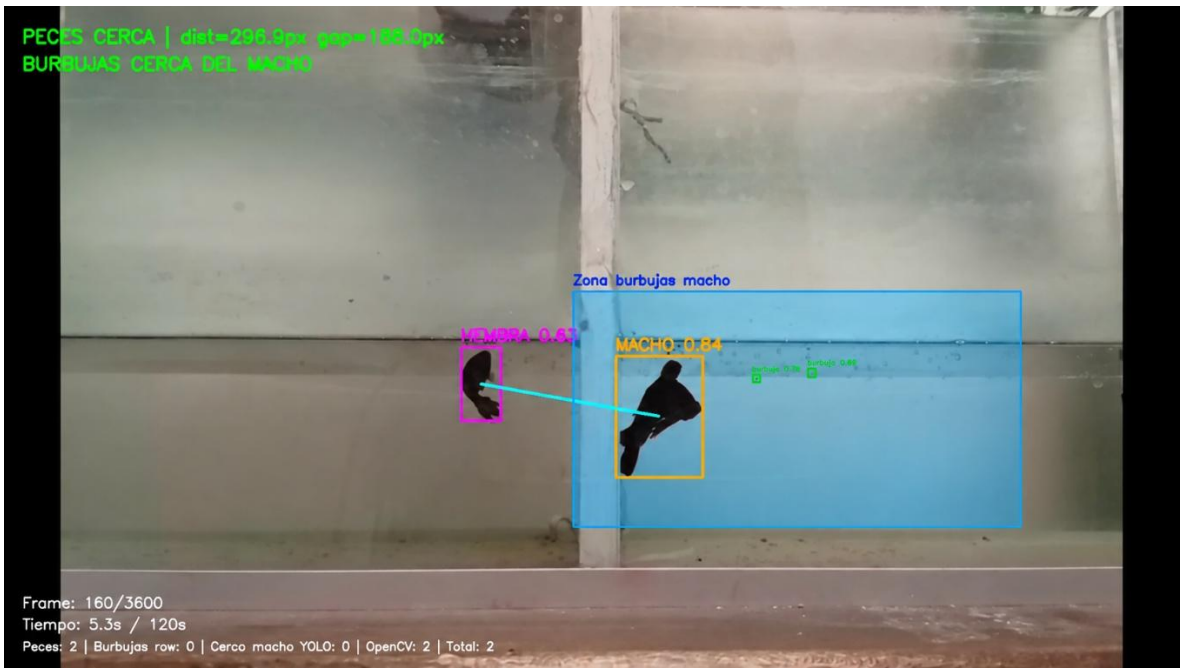
Recolección de imágenes	Se utilizaron imágenes del pez Betta macho, hembra y burbujas obtenidas desde Roboflow.	Dataset inicial del proyecto
Etiquetado	Se marcaron las clases macho, hembra y burbujas mediante cajas delimitadoras.	Imágenes anotadas
División del dataset	El conjunto de datos se organizó en entrenamiento, validación y prueba.	Datos preparados para entrenamiento
Entrenamiento	Se entrenó el modelo YOLOv11 con las imágenes etiquetadas.	Ajuste de pesos del modelo
Selección del modelo	Se seleccionó el archivo con mejor desempeño durante el entrenamiento.	Archivo best.pt
Despliegue	El modelo best.pt fue cargado en la Raspberry Pi para procesar video en tiempo real.	Detección en el prototipo

Nota. La tabla resume el flujo seguido desde la preparación del dataset hasta la implementación del modelo dentro del sistema embebido.

Una vez obtenido el modelo entrenado, el archivo best.pt fue incorporado en la Raspberry Pi, la cual actúa como unidad principal de procesamiento del sistema. La cámara instalada en el acuario captura continuamente el entorno experimental y cada fotograma es analizado por YOLOv11 para identificar los elementos de interés. Como se observa en la Figura 11, el sistema genera cuadros delimitadores sobre los objetos detectados, mostrando la clase correspondiente y el valor de confianza asignado por el modelo.

Figura 11

Detección simultanea

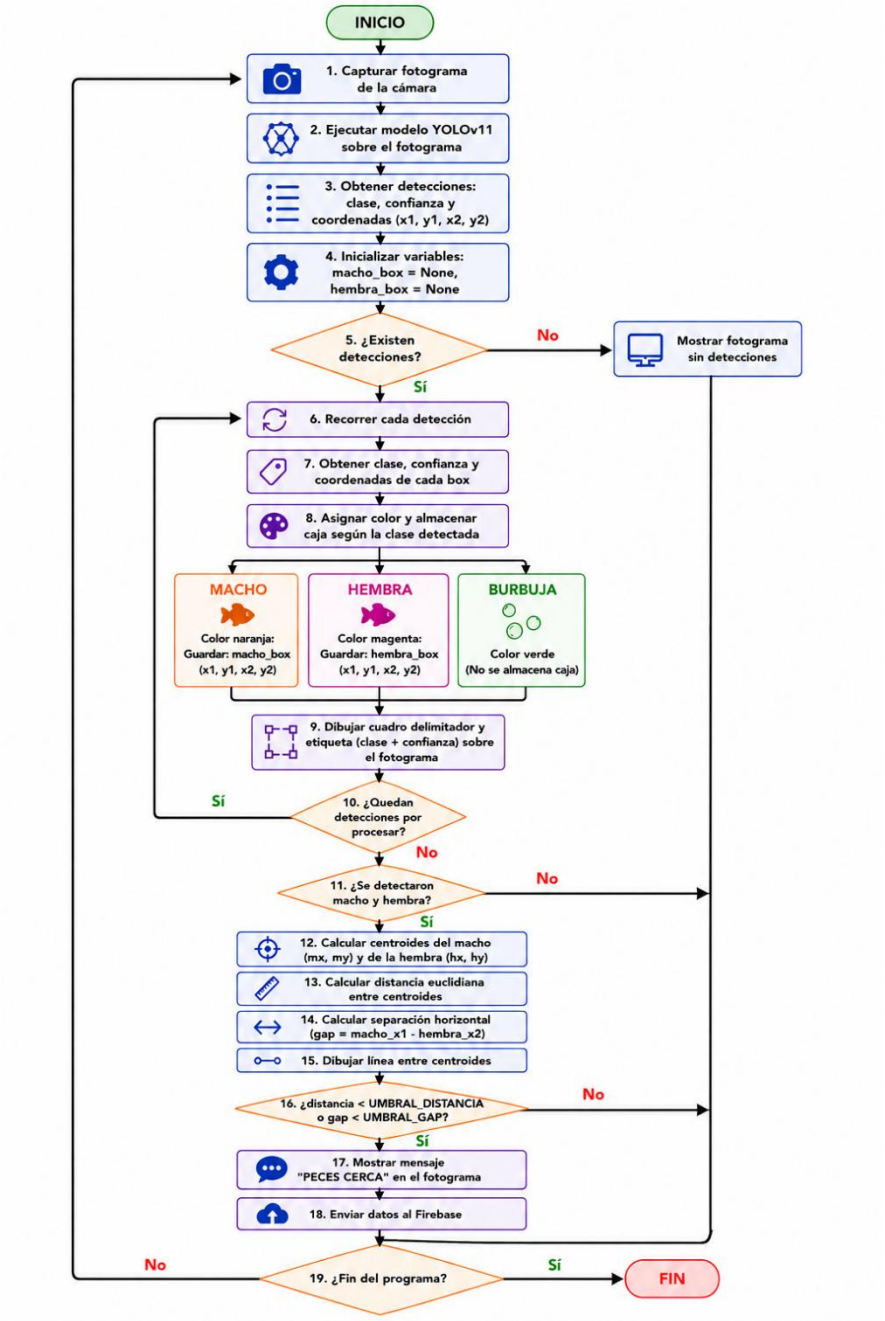


Los cuadros de detección observados en la Figura 11 se generan a partir de las coordenadas entregadas por YOLOv11 para cada objeto identificado. Estas coordenadas corresponden a los puntos de la caja delimitadora, los cuales son utilizados por OpenCV para dibujar el rectángulo sobre el fotograma. Además, el sistema muestra la etiqueta de la clase detectada y su valor de confianza, permitiendo visualizar en tiempo real si el objeto corresponde al macho, la hembra o una burbuja.

Para comprender la lógica implementada en el subsistema de visión artificial, la Figura 12 representa de forma secuencial el procesamiento realizado sobre cada fotograma capturado por la cámara, desde la ejecución del modelo YOLOv11 hasta el análisis de proximidad entre los peces y el envío de información al sistema de almacenamiento.

Figura 12

Diagrama de flujo del procesamiento de imágenes y análisis de proximidad mediante YOLOv11.



Como se observa en la Figura 12, el proceso inicia con la captura de un fotograma de la cámara, sobre el cual se ejecuta el modelo YOLOv11. A partir de la inferencia se

obtienen la clase identificada, el nivel de confianza y las coordenadas de cada caja delimitadora. Las detecciones son procesadas individualmente y clasificadas como macho, hembra o burbuja, permitiendo posteriormente dibujar sobre el fotograma los cuadros delimitadores y sus respectivas etiquetas.

Una vez procesadas las detecciones del fotograma, el sistema verifica si el macho y la hembra fueron identificados simultáneamente. Esta condición es necesaria antes de ejecutar el análisis de proximidad; por lo tanto, si alguno de los peces no se encuentra presente, no se realizan cálculos de distancia y el sistema continúa con el ciclo de procesamiento de los siguientes fotogramas.

Cuando ambas clases son detectadas, se calculan los centroides de sus cajas delimitadoras y, a partir de estos puntos, se determina la distancia euclidiana entre los peces. De manera complementaria, se calcula la separación horizontal (gap) entre las cajas, utilizada como un segundo criterio para estimar la cercanía entre los ejemplares. El sistema compara estos valores con los umbrales establecidos y, cuando se cumple la condición de proximidad, muestra el mensaje “PECES CERCA” y envía los datos correspondientes a Firebase.

De esta manera, el procesamiento se ejecuta de forma cíclica sobre los fotogramas obtenidos por la cámara, permitiendo mantener el análisis de los peces durante el funcionamiento del sistema. El código fuente utilizado para implementar esta lógica se presenta en el Anexo E, donde puede revisarse con mayor detalle el procedimiento de inferencia, procesamiento de las detecciones y cálculo de proximidad.

La principal diferencia entre este sistema y una detección estática tradicional radica en que la visión artificial no se limita únicamente a reconocer objetos dentro de una imagen. Las detecciones realizadas por YOLOv11 constituyen la entrada para una lógica posterior de análisis, mediante la cual se evalúa la interacción entre el macho y la hembra y se generan los datos utilizados por los demás componentes del sistema.

Además, cuando se identifican burbujas dentro de la región analizada, esta información puede emplearse como un indicador complementario relacionado con el comportamiento reproductivo del pez Betta. De esta forma, el subsistema de visión artificial integra la detección de objetos, el análisis espacial entre los peces y el registro de información, proporcionando una base automatizada para el seguimiento del proceso reproductivo.

4.1.3. Implementación de la lógica de avistamientos

Una de las funciones más relevantes del sistema es la generación automática de avistamientos, definidos como eventos en los que el macho y la hembra son detectados simultáneamente y además presentan una condición de proximidad dentro del mismo fotograma. Esta lógica permite transformar la detección visual en una variable funcional de monitoreo, más útil que una simple identificación individual de peces.

La proximidad entre ambos peces se calcula mediante la distancia euclidiana entre los centroides de las cajas delimitadoras; este cálculo se expresa en la Ec. 2 donde

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad \text{Ec. 2}$$

x_1, y_1 corresponden al centroide de la detección de la hembra y x_2, y_2 al centroide de la detección del macho. Cuando el valor de d es menor al umbral definido por el sistema, se genera automáticamente un nuevo avistamiento.

La Figura 13 ilustra esta lógica de forma clara. En ella se observa el registro del Avistamiento #79, generado cuando el sistema detectó simultáneamente a la hembra y al macho con valores de confianza de 0.82 y 0.89, respectivamente. Esta evidencia visual confirma que el avistamiento no depende únicamente de la presencia de ambos peces, sino de su relación espacial dentro del fotograma.

Figura 13

Ejemplo de avistamiento detectado por el sistema en tiempo real



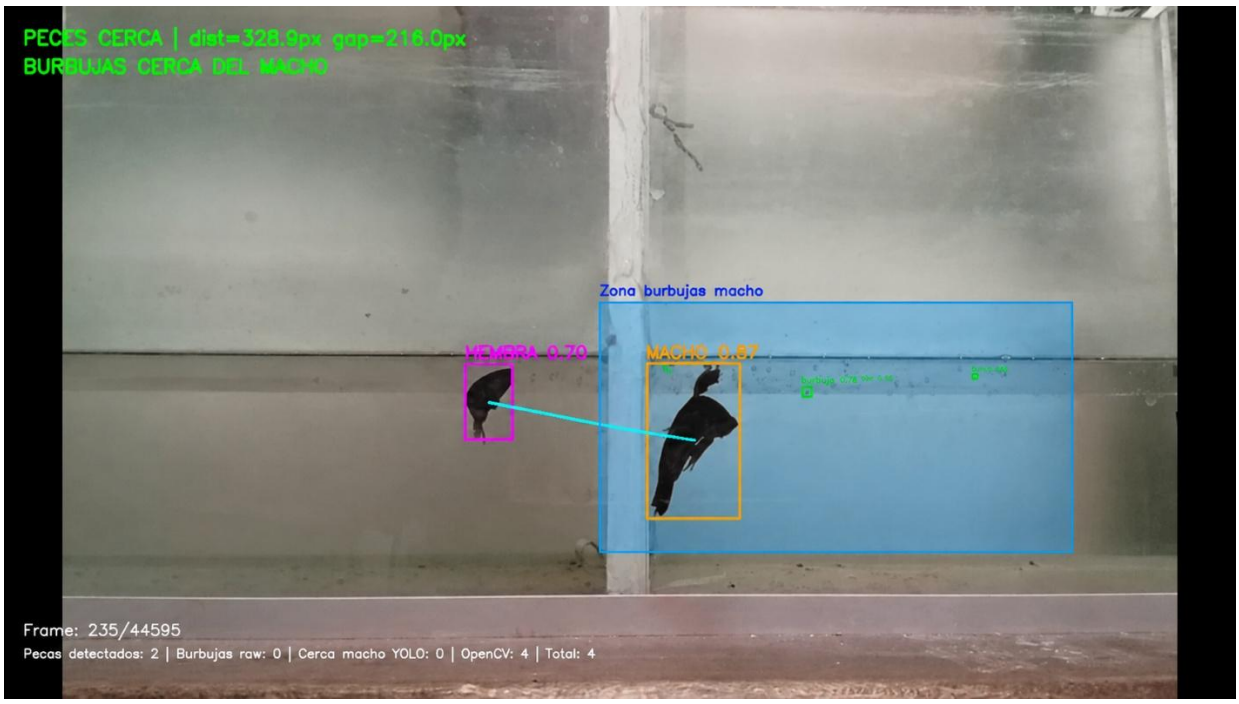
4.1.4. Implementación de la verificación de burbujas

Una vez generado un avistamiento, el sistema almacena la imagen correspondiente y ejecuta sobre ella una verificación adicional de burbujas. Esta etapa no se realiza sobre todos los fotogramas del video, sino únicamente sobre las imágenes previamente seleccionadas por el sistema, lo cual reduce la carga de procesamiento y aumenta la pertinencia del análisis.

La verificación de burbujas constituye un criterio complementario, ya que agrega un patrón visual adicional al análisis del comportamiento observado. En este contexto, la Figura 13 debe mostrar una imagen almacenada por el sistema y posteriormente analizada para identificar la presencia de burbujas sobre la superficie del agua. Si la verificación es positiva, el sistema registra este resultado dentro de Firebase, permitiendo conservar evidencia acumulada de estos eventos a lo largo del tiempo, tal como se muestra en la Figura 14.

Figura 14

Verificación de burbujas sobre imagen por avistamiento



4.1.5. Implementación de la aplicación móvil

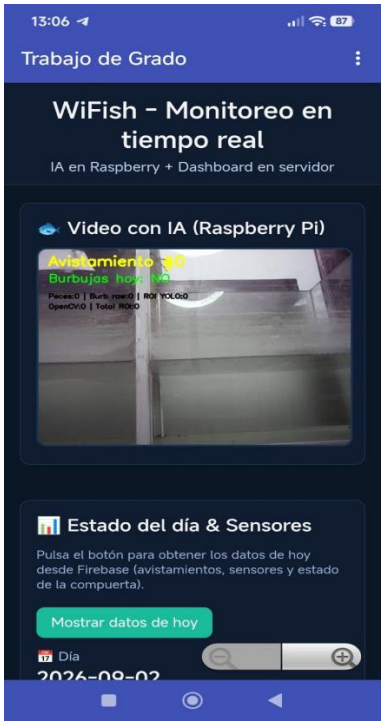
Para complementar el monitoreo del sistema embebido, se implementó una aplicación móvil mediante MIT App Inventor, con la finalidad de permitir el acceso a la información generada por el prototipo desde un dispositivo móvil. La aplicación se integra con los datos almacenados en Firebase Realtime Database, permitiendo consultar la información generada por los subsistemas de visión artificial y sensado ambiental.

Como se observa en la Figura 15, la aplicación dispone de una sección destinada a la visualización del video procesado por la Raspberry Pi, donde se muestra el funcionamiento del modelo de inteligencia artificial y la información relacionada con los avistamientos y la verificación de burbujas. De esta manera, el usuario puede observar de

forma remota el comportamiento registrado dentro del acuario sin necesidad de encontrarse físicamente junto al prototipo.

Figura 15

Interfaz de la aplicación móvil para el monitoreo del sistema



Además, la aplicación permite consultar los datos correspondientes al estado del sistema durante el día. Esta información es obtenida desde Firebase e incluye los avistamientos registrados, los valores de los sensores y el estado de la compuerta. La integración de estos elementos permite centralizar dentro de una misma interfaz móvil la información generada por el sistema embebido, facilitando el seguimiento de las condiciones ambientales y de los eventos identificados mediante visión artificial.

La comunicación con Firebase permite que la información presentada en la aplicación se mantenga relacionada con los registros generados por la Raspberry Pi y el ESP32. De esta forma, la aplicación constituye la interfaz de acceso remoto del prototipo y complementa los mecanismos de notificación implementados mediante Telegram, proporcionando al usuario una alternativa para consultar el estado del acuario y la información asociada al comportamiento reproductivo de los peces Betta.

4.2. Resultados del período de operación del sistema

Dado que el proyecto ha permanecido encendido y operando durante varios meses, los resultados no se presentan en este capítulo como observaciones diarias aisladas, sino como resultados acumulados y métricas globales del período de funcionamiento. Esta forma de presentación resulta más adecuada para un sistema IoT de operación prolongada, ya que permite visualizar de manera integral el volumen de información procesada y el comportamiento general del prototipo.

El resumen de este período de operación se presenta en la Tabla 10, donde se incluyen el tiempo total de funcionamiento del sistema, la cantidad de secuencias de video procesadas, el número total de avistamientos registrados, el número de imágenes almacenadas, el total de verificaciones positivas de burbujas y la cantidad de registros históricos generados por el subsistema de sensores.

Tabla 10

Resumen acumulado del período de operación del sistema

Parámetro	Valor
Tiempo total de operación	150 días
Número acumulado de avistamientos	25875
Número acumulado de imágenes almacenadas	10482
Número acumulado de verificaciones positivas de burbujas	2014
Número de registros históricos de sensores	1405903

4.3. Resultados del subsistema de visión artificial

Los resultados del subsistema de visión artificial deben interpretarse a partir de dos niveles: primero, la capacidad del sistema para detectar al macho y a la hembra en tiempo real; y segundo, su capacidad para generar avistamientos y almacenar imágenes relevantes. El resumen de estos resultados se presenta en la Tabla 11.

Tabla 11

Tabla de identificación.

Variable evaluada	Resultado acumulado
Avistamientos generados	25875
Imágenes almacenadas por avistamiento	10482
Imágenes con verificación positiva de burbujas	2014

La Tabla 11 evidencia que la detección visual no se limita a identificar objetos, sino que desencadena eventos funcionales dentro del sistema. Esta interpretación se encuentra respaldada por la Figura 11, la Figura 12 y la Figura 13, donde se muestran ejemplos concretos de detección, avistamiento y verificación de burbujas.

4.4. Resultados del monitoreo de variables fisicoquímicas

Como el sistema ha operado de forma continua, no resulta conveniente mostrar cada lectura individual de sensores. En su lugar, se presenta un resumen estadístico del comportamiento de las variables fisicoquímicas más relevantes durante el período de monitoreo. Estos resultados se muestran en la Tabla 12, donde se reportan los valores mínimo, máximo y promedio de temperatura representado por grados Celsius, pH y en partes por millón (ppm) en el total de solidos disueltos (TDS).

Tabla 12

Valores Censados

Variable	Valor mínimo	Valor máximo	Valor promedio
Temperatura (°C)	22	28.5	27.05°
pH	4.6	14.5	8.96
TDS (ppm)	32.23	361.9	235.16

La Tabla 12 proporciona una visión global del contexto ambiental en el que ocurrieron los avistamientos y la verificación de burbujas, los promedios obtenidos se mantuvieron dentro de rangos adecuados para la especie.

En el caso del monitoreo ambiental, las lecturas fueron tomadas con una frecuencia constante, permitiendo observar la evolución de las condiciones del agua durante el tiempo de ejecución del sistema. En conjunto, los datos recopilados durante la

implementación constituyeron el insumo necesario para la fase de pruebas y análisis de resultados.

5. Conclusiones y Recomendaciones

5.1. Conclusiones

En este proyecto se logró desarrollar un sistema de monitoreo acuático inteligente que integra exitosamente tecnologías de IoT y visión artificial para la supervisión de un acuario. Los resultados demuestran que el sistema mejora sustancialmente las capacidades de control y observación en comparación con métodos tradicionales. A continuación, se sintetizan las principales conclusiones:

La revisión bibliográfica permitió establecer los fundamentos biológicos y tecnológicos necesarios para el desarrollo del sistema, identificando factores relacionados con el proceso reproductivo de los peces Betta splendens, entre ellos las condiciones fisicoquímicas del agua, el acercamiento entre el macho y la hembra y la formación del nido de burbujas. De igual manera, el estudio de inteligencia artificial, visión artificial y redes neuronales permitió determinar las características necesarias para seleccionar una arquitectura de deep learning capaz de procesar información visual dentro de un sistema embebido, relacionando de esta manera el comportamiento biológico de la especie con los criterios utilizados posteriormente para el diseño del prototipo.

La implementación del modelo de visión artificial basado en YOLOv11 permitió identificar dentro de los fotogramas al pez Betta macho, la hembra y la presencia de burbujas, siendo integrado en una Raspberry Pi como unidad principal de procesamiento. A partir de estas detecciones se incorporó una lógica basada en la proximidad entre ambos peces, permitiendo que el sistema no se limite únicamente al reconocimiento visual de las

clases entrenadas, sino que genere automáticamente avistamientos cuando se presentan condiciones de acercamiento entre los ejemplares y posteriormente verifique la presencia de burbujas como otro indicador asociado al comportamiento reproductivo. Este procesamiento permitió convertir la información obtenida mediante la cámara en eventos que pueden ser almacenados y utilizados para el seguimiento del proceso dentro del acuario.

La integración de la Raspberry Pi, el ESP32, la cámara, los sensores y el mecanismo de accionamiento de la compuerta permitió estructurar un sistema embebido capaz de realizar simultáneamente el procesamiento de video y el monitoreo de las condiciones ambientales del acuario. La utilización de Firebase Realtime Database permitió centralizar los avistamientos, las mediciones de los sensores y los estados generados durante el funcionamiento, mientras que la aplicación móvil desarrollada proporcionó acceso remoto a esta información y complementó, junto con Telegram, los mecanismos de consulta y notificación del sistema. De esta manera, la pecera dejó de funcionar únicamente como un entorno físico de reproducción y pasó a constituir un entorno de monitoreo automatizado con acceso remoto a la información generada durante su operación.

El funcionamiento prolongado del prototipo permitió comprobar la integración y operación conjunta de los diferentes subsistemas desarrollados. Durante 150 días de funcionamiento se registraron 25 875 avistamientos, se almacenaron 10 482 imágenes, se obtuvieron 2014 verificaciones positivas de burbujas y se generaron 1 405 903 registros históricos provenientes del monitoreo ambiental. Estos resultados evidencian que el

sistema fue capaz de capturar, procesar, almacenar y mantener información de manera continua durante el período experimental, demostrando la funcionalidad del prototipo en condiciones reales de operación y permitiendo disponer de información acumulada para el análisis del comportamiento observado en los peces.

En conjunto, el desarrollo realizado permitió obtener un sistema embebido que combina deep learning, visión artificial, sensado ambiental, almacenamiento en la nube y acceso remoto para apoyar la identificación y seguimiento de indicadores relacionados con el comportamiento reproductivo de los peces Betta splendens. La detección simultánea del macho y la hembra, el análisis de su proximidad y la posterior verificación de burbujas proporcionan al sistema criterios observables para reconocer eventos de interés sin depender únicamente de la supervisión permanente de una persona. Por lo tanto, el prototipo constituye una herramienta tecnológica de apoyo para automatizar el monitoreo del proceso reproductivo, manteniendo además un registro histórico que puede ser utilizado para posteriores análisis y mejoras del sistema.

5.2. Recomendaciones

A pesar de los logros obtenidos, el desarrollo de este sistema abre nuevas posibilidades de mejora y expansión. A continuación, se enumeran recomendaciones puntuales y consideraciones para trabajos futuros que podrían complementar y optimizar el sistema:

Se recomienda ampliar el conjunto de datos visuales utilizado por el sistema, incorporando imágenes y secuencias obtenidas bajo distintas condiciones de iluminación,

reflejos, ángulos de captura y variaciones del comportamiento de los peces. Esto permitiría mejorar la capacidad de generalización del modelo YOLOv11 durante su operación en tiempo real.

Se sugiere fortalecer la lógica de generación de avistamientos mediante mecanismos de validación temporal, por ejemplo confirmando la proximidad entre macho y hembra durante varios fotogramas consecutivos o aplicando una ventana de exclusión entre eventos. Esta mejora ayudaría a evitar conteos repetitivos de un mismo acercamiento y aumentaría la precisión conductual del sistema.

Se recomienda complementar el análisis visual con un estudio más profundo de la relación entre las variables fisicoquímicas del acuario y los eventos detectados por el sistema. En particular, correlacionar temperatura, pH y TDS con los avistamientos y la presencia de burbujas permitiría obtener conclusiones más robustas sobre las condiciones favorables para la reproducción de la especie.

Se aconseja mejorar las condiciones físicas del entorno de captura, especialmente la iluminación y el control de reflejos en el acuario, ya que estos factores pueden influir directamente en la calidad de los fotogramas procesados y, por tanto, en el rendimiento del modelo de visión artificial. Una escena más controlada favorecería detecciones más estables y confiables.

Se recomienda incorporar mecanismos adicionales de respaldo y gestión de datos, de modo que la información generada por el sistema pueda conservarse de forma segura y utilizarse posteriormente para análisis históricos, generación de reportes o

entrenamiento de versiones mejoradas del modelo. Esta recomendación es coherente con la naturaleza IoT y de monitoreo continuo del sistema planteado.

Referencias Bibliográficas

[1] R. Acosta and V. Riveros, “Las tecnologías de la información y comunicación como mediadoras en el aprendizaje de la biología. Algunas consideraciones,” Omnia, vol. 18, no. 1, pp. 25–44, 2012.

[2] C. D. Arauz Jovel, “Experiencia en la reproducción de la carpa Koi (Cyprinus carpio) y el pez Betta (Betta splendens) bajo condiciones de Zamorano, Honduras,” Proyecto Especial del Programa de Ingeniero Agrónomo, Escuela Agrícola Panamericana, Zamorano, Honduras, 2000.

[3] D. A. Arboleda Obregón, “Crianza y producción del Betta (Betta splendens) para acuaristas no profesionales,” REDVET. Revista Electrónica de Veterinaria, vol. 7, no. 4, pp. 1–4, 2006.

[4] M. C. Arboleda Tabares, “Evaluación de la reversión sexual del pez Betta (Betta splendens) en etapa de alevinaje,” Tesina, Programa de Tecnología en Acuicultura, Universidad del Pacífico, Buenaventura, Colombia, 2022.

[5] D. Bueno and J. Mantilla, Sistemas embebidos: fundamentos y aplicaciones. Editorial Académica Española, 2017.

[6] J. A. Castro, “Reproducción del pez ornamental betta ‘Betta splendens’ en cautiverio en el cantón Huaquillas,” 2021. [Online]. Available: https://www.academia.edu/48414748/Reproducci%C3%B3n_del_pez_ornamental_betta_Betta_splendens_en_cautiverio_en_el_cant%C3%B3n_Huaquillas

[7] C. D. Andrade Chimba, “Implementación de herramientas de seguridad web en una tienda de ventas por internet desarrollada en Prestashop,” Trabajo de titulación, Universidad de las Américas, Quito, Ecuador, 2016.

[8] Comisión Nacional para el Conocimiento y Uso de la Biodiversidad (CONABIO), “Evaluación rápida de invasividad de Betta splendens,” Sistema de Información sobre Especies Invasoras en México, México, 2017.

[9] B. H. Curtin and S. J. Matthews, “Deep Learning for Inexpensive Image Classification of Wildlife on the Raspberry Pi,” in 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), 2019, pp. 82–87, doi: 10.1109/UEMCON47517.2019.8993061.

[10] L. Deng and D. Yu, “Deep Learning: Methods and Applications,” Foundations and Trends in Signal Processing, vol. 7, nos. 3–4, pp. 197–387, 2014, doi: 10.1561/2000000039.

[11] J. Díaz Delgado, “Implementación de modelos de deep learning en sistemas embebidos para aplicaciones móviles,” Revista de Ingeniería Electrónica, vol. 15, no. 2, pp. 45–60, 2022.

[12] T. Diwan, G. Anirudh, and J. V. Tembhurne, “Object detection using YOLO: Challenges, architectural successors, datasets and applications,” Multimedia Tools and Applications, vol. 82, no. 6, pp. 9243–9275, 2023.

[13] L. E. B. Enríquez, F. L. García Gil de Muñoz, J. P. Orduña Villalobos, D. Villeda Cuevas, E. R. Fernández, and D. R. R. Martínez, “Identificación de componentes de las burbujas protectoras producidas por el macho del pez betta (Betta splendens),” Investigación Universitaria Multidisciplinaria, no. 12, pp. 145–149, 2013.

[14] G. Galeano, Programación de sistemas embebidos en C: teoría y prácticas aplicadas a cualquier microcontrolador. Alfaomega, 2009.

[15] A. M. Gallo Encalada, “Estudio comparativo para la obtención de machos en la especie Betta splendens (Charles Take Regan, 1910), ‘Pez Betta’ aplicando el método de reproducción por control de temperatura y método natural,” Tesis de grado, Universidad Nacional de Piura, Piura, Perú, 2017.

[16] J. García, “Diseño de sistemas embebidos para aplicaciones IoT,” Journal of Embedded Systems, vol. 12, no. 3, pp. 78–92, 2019.

[17] C. García Bermejo, “Classification of medical images using Deep Learning techniques,” Universidad de Cantabria, 2020. [Online]. Available: https://repositorio.unican.es/xmlui/handle/10902/20624

[18] J. L. González-Andújar, R. Ferrero, M. Lima, P. A. Marquet, F. Valladares, and S. A. Navarrete, “Población humana y uso de recursos,” in Cambio Global. Una mirada desde Iberoamérica, 2018, pp. 31–50.

[19] E. R. Gutiérrez Callisaya, “Evaluación del crecimiento de peces Carpa Koi platino y Ogon chica ornamentales (Cyprinus carpio) en agua fría bajo condiciones controladas,” Revista Estudiantil AGRO-VET, vol. 3, no. 1, pp. 312–320, 2019.

[20] A. Herrera Arango, J. A. Rojas Collazos, and J. L. Toscano Castro, “Conservación del hábitat de peceras y/o estanques de peces ornamentales con tecnologías IoT,” 2023.

[21] IEEE Standards Association, “IEEE 1012-2024 — Standard for System, Software, and Hardware Verification and Validation,” 2024.

[22] IEEE Standards Association, “IEEE 1016-2009 — Systems Design — Software Design Descriptions,” 2009.

[23] IEEE Standards Association, “IEEE/ISO/IEC 29119-1-2021 — Software and systems engineering — Software testing — Part 1: General concepts,” 2021.

[24] IEEE Standards Association, “IEEE/ISO/IEC 29148-2018 — Systems and software engineering — Life cycle processes — Requirements engineering,” 2018.

[25] IEEE Standards Association, “IEEE/ISO/IEC 42010-2022 — Systems and software engineering — Architecture description,” 2022.

[26] IONOS, “¿Qué es un píxel?,” 2021.

[27] R. Khanam and M. Hussain, “YOLOv11: An Overview of the Key Architectural Enhancements,” arXiv:2410.17725, 2024, doi: 10.48550/arXiv.2410.17725.

[28] J. M. Laborde, C. Carbone, M. Á. Ayala, and M. del P. L. Cagliada, Peces como animales de experimentación. Libros de Cátedra, 2021.

[29] M. Á. Landines Parra, Producción de peces ornamentales de la Orinoquía colombiana, vol. 2, pp. 154–157, 2007.

[30] C. L. Macoretta, "Desarrollo e implementación de técnicas para el enfriamiento de embriones de dos especies de peces de interés comercial y científico: el pejerrey (Odontesthes bonariensis) y el pez beta (Betta splendens)," Tesis doctoral, Universidad de Buenos Aires, Buenos Aires, Argentina, 2022. [Online]. Available: https://hdl.handle.net/20.500.12110/tesis_n7280_Macoretta

[31] S. Madakam, "Internet of Things: Smart Things," International Journal of Future Computer and Communication, vol. 4, no. 4, pp. 250–253, 2015, doi: 10.7763/IJFCC.2015.V4.395.

[32] B. Mahesh, "Machine Learning Algorithms — A Review," International Journal of Science and Research, vol. 9, no. 1, pp. 381–386, 2020, doi: 10.21275/ART20203995.

[33] W. F. Manrique Chalco, "Detección y segmentación de peces utilizando técnicas de deep learning," 2023.

[34] A. Marín Castejón, "Aplicación de Deep Learning para problemas de clasificación de imágenes," 2019. [Online]. Available: https://hdl.handle.net/20.500.14352/14349

[35] M. Massiris, C. Delrieux, and J. Á. Fernández Muñoz, “Detección de equipos de protección personal mediante red neuronal convolucional YOLO,” in XXXIX Jornadas de Automática, 2018, pp. 1022–1029.

[36] R. Miranda, “Opérculo,” Universidad de Navarra, 2002.

[37] J. A. C. Moreno and V. A. Álvarez, “Correlación entre el sistema vasotonérgico y comportamientos agonistas de Betta splendens,” 2019.

[38] N. H. Motlagh, M. Mohammadrezaei, J. Hunt, and B. Zakeri, “Internet of Things (IoT) and the Energy Sector,” Energies, vol. 13, no. 2, Art. no. 494, 2020, doi: 10.3390/en13020494.

[39] J. G. Navas Aquim, P. Quizhpe Cordero, W. G. Galarza Mora, and M. M. Quezada Morocho, “Reproducción normal y asistida del Betta Splendens en condiciones de acuario en la estación de maricultura FCA-UTMACH,” Dominio de las Ciencias, vol. 9, no. 3, pp. 1334–1393, 2023, doi: 10.23857/dc.v9i3.3502.

[40] G. Parisse, El gran libro de los peces tropicales. Parkstone International, 2017.

[41] J. C. F. Pérez, “Acuariofilia: enfermedades y tratamientos de peces de acuario,” Panorama Actual del Medicamento, vol. 41, no. 403, pp. 470–476, 2017.

[42] J. Portilla, G. Mujica, J.-S. Lee, and T. Riesgo, “The Extreme Edge at the Bottom of the Internet of Things: A Review,” IEEE Sensors Journal, vol. 19, no. 9, pp. 3179–3190, 2019, doi: 10.1109/JSEN.2019.2891911.

[43] G. Y. Prado Riaño, M. Arredondo Castro, and R. Martínez-Yáñez, “Presencia de parásitos en peces de ornato,” Jóvenes en la Ciencia, vol. 4, no. 1, pp. 61–65, 2018.

[44] R. S. Pressman, Software Engineering: A Practitioner’s Approach, 8th ed. McGraw-Hill Education, 2014.

[45] D. Ratnasari, R. Mardhiyyah, and A. Pramudwiatmoko, “IoT Prototype Development of Automatic Fish Feeder and Water Replacement,” International Journal of Engineering, Technology and Natural Sciences, vol. 2, no. 2, 2020.

[46] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[47] L. Rincón Camacho, “Biología reproductiva de dos especies de peces ornamentales: el neón cardenal Paracheirodon axelrodi y la tetra cola roja Aphyocharax anisitsi (Characiformes, Characidae),” 2019.

[48] A. A. Rodríguez Tellez, “Manual de manejo básico del pez Betta Splendens,” Universidad Autónoma Metropolitana, Unidad Xochimilco, México, 2022. [Online]. Available: https://repositorio.xoc.uam.mx/jspui/handle/123456789/37014

[49] P. P. Shinde and S. Shah, “A Review of Machine Learning and Deep Learning Applications,” in 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018, pp. 1–6, doi: 10.1109/ICCUBEA.2018.8697857.

[50] E. R. Silva Guzmán, “Entrenamiento de la Red Neuronal Convolucional YOLO para objetos,” Universidad Nacional Autónoma de México, Ciudad de México, 2020.

[51] Ultralytics, “Ultralytics YOLO11,” 2024. [Online]. Available: https://github.com/ultralytics/ultralytics

[52] J. J. Vásquez Agudelo, “Sistema de detección de presencia de personas en emergencias para la Universidad Técnica del Norte,” Universidad Técnica del Norte, Ibarra, Ecuador, 2024.

[53] I. A. Vergara Luviano, “Mantenimiento y Producción de Pez Betta (Betta splendes) en un Programa de Mejoramiento Genético,” Universidad Autónoma Metropolitana, Unidad Xochimilco, México, 2023. [Online]. Available: https://repositorio.xoc.uam.mx/jspui/handle/123456789/40844

[54] J. L. Vivas Delgado, “Comercio internacional de peces ornamentales,” Trabajo Fin de Grado, Facultad de Comercio, Universidad de Valladolid, Valladolid, España, 2019. [Online]. Available: https://uvadoc.uva.es/handle/10324/38339

[55] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detection,” arXiv:2207.02696, 2022, doi: 10.48550/arXiv.2207.02696.

6. Anexos

6.1. Anexo A. Evidencia fotográfica del prototipo implementado

En este anexo se presenta la evidencia fotográfica del prototipo desarrollado para el monitoreo del comportamiento reproductivo de los peces *Betta splendens*. Las imágenes incluidas muestran el montaje físico del sistema, compuesto por la pecera experimental, la Raspberry Pi como unidad central de procesamiento, el microcontrolador ESP32, la cámara de captura de video, los sensores ambientales y los elementos complementarios de conexión y alimentación.

La finalidad de este anexo es evidenciar la implementación real del prototipo y mostrar la disposición física de sus componentes dentro del entorno de experimentación. De esta manera, se respalda visualmente la integración descrita en el capítulo metodológico y en el capítulo de implementación y resultados, confirmando que el sistema no se limitó al diseño teórico, sino que fue construido y puesto en funcionamiento en un entorno controlado.

Figura 16

Foto general del prototipo

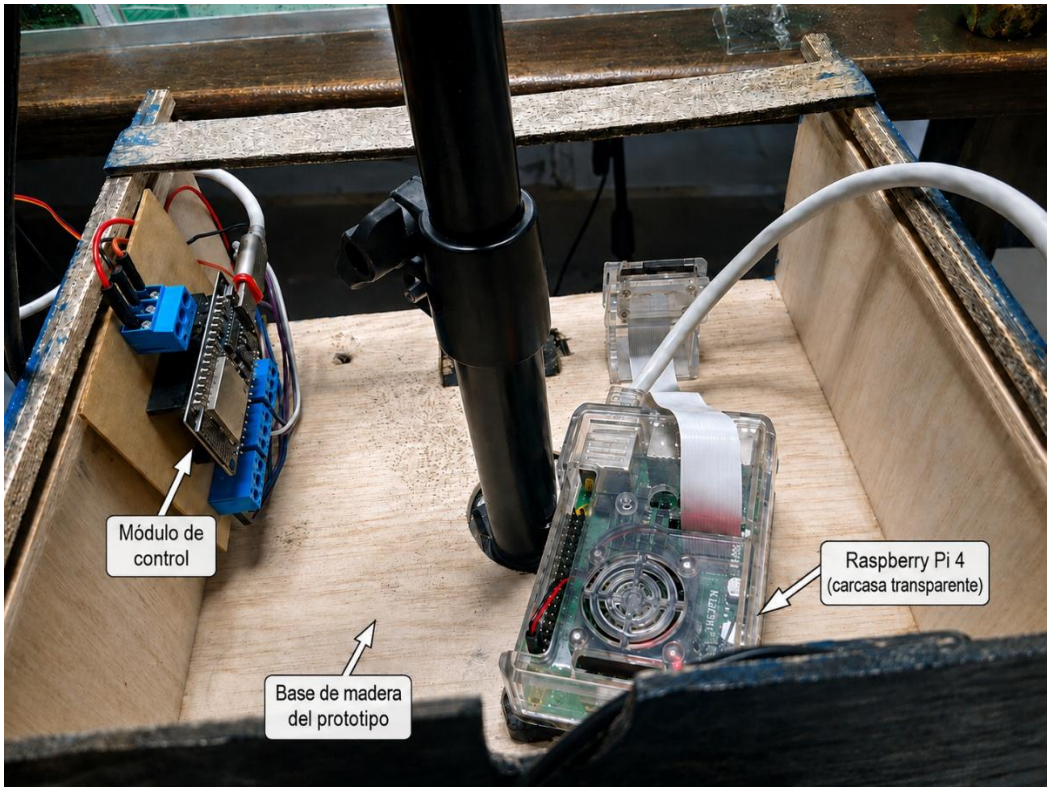


Figura 17

Foto de la pecera.



Figura 18

Foto de la Raspberry Pi



Figura 19

Foto del ESP32.

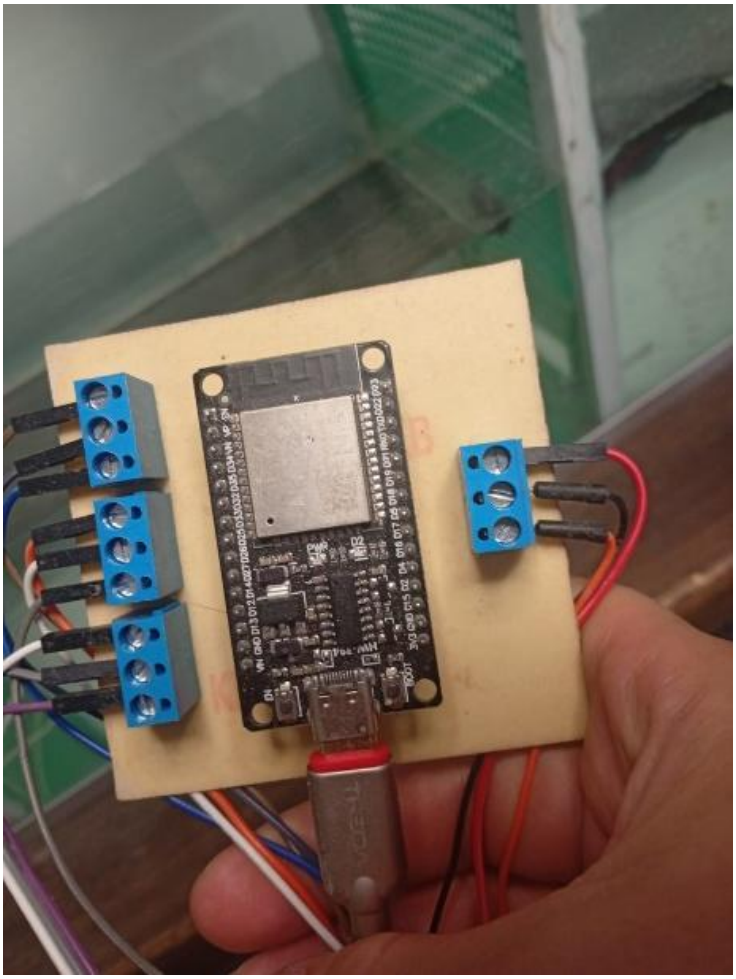


Figura 20

Foto de la cámara

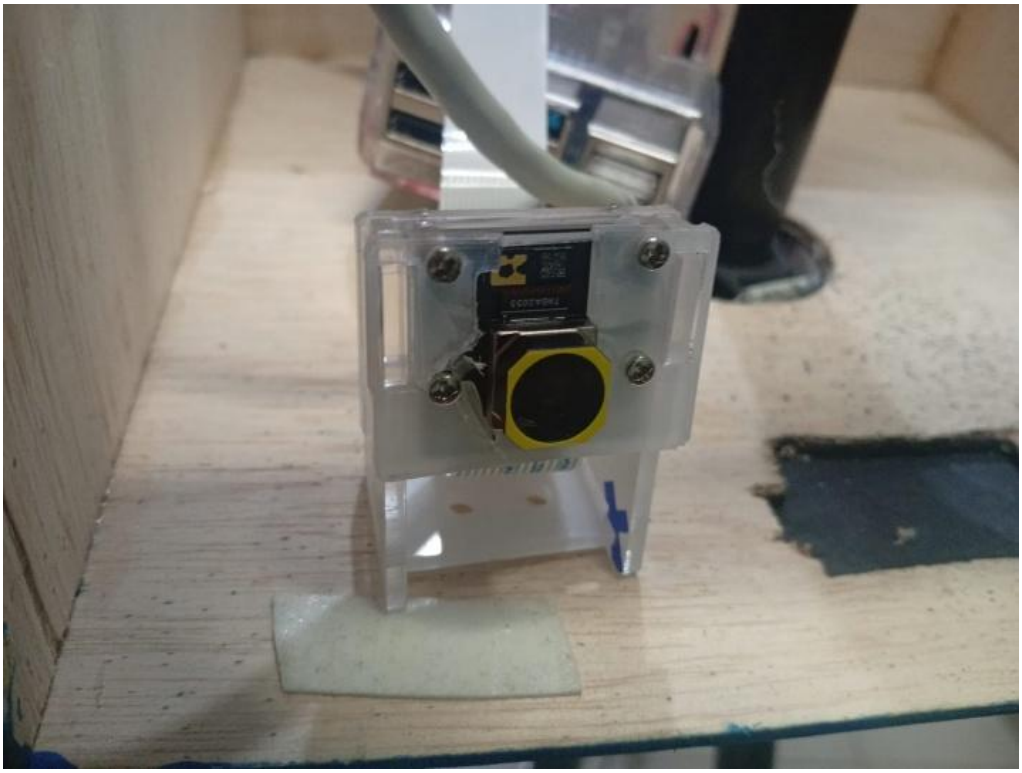


Figura 21

Foto de los sensores instalados.



Figura 22

Foto del montaje completo.



6.2. Anexo B. Evidencia del sistema de visión artificial en funcionamiento

En este anexo se presentan capturas del sistema de visión artificial durante su operación en tiempo real. Las imágenes muestran el funcionamiento del modelo YOLOv11 al detectar la presencia del pez macho y de la hembra dentro del acuario, así

como la representación gráfica de las cajas delimitadoras y los valores de confianza asignados por el modelo a cada detección.

Este anexo tiene como objetivo respaldar visualmente la capacidad del sistema para procesar video en tiempo real y reconocer los elementos de interés definidos en el proyecto. Asimismo, permite evidenciar la manera en que el modelo de visión artificial fue integrado al sistema embebido, constituyéndose en la base para la generación de avistamientos y el análisis posterior del comportamiento reproductivo.

Figura 23

Captura detectando macho.

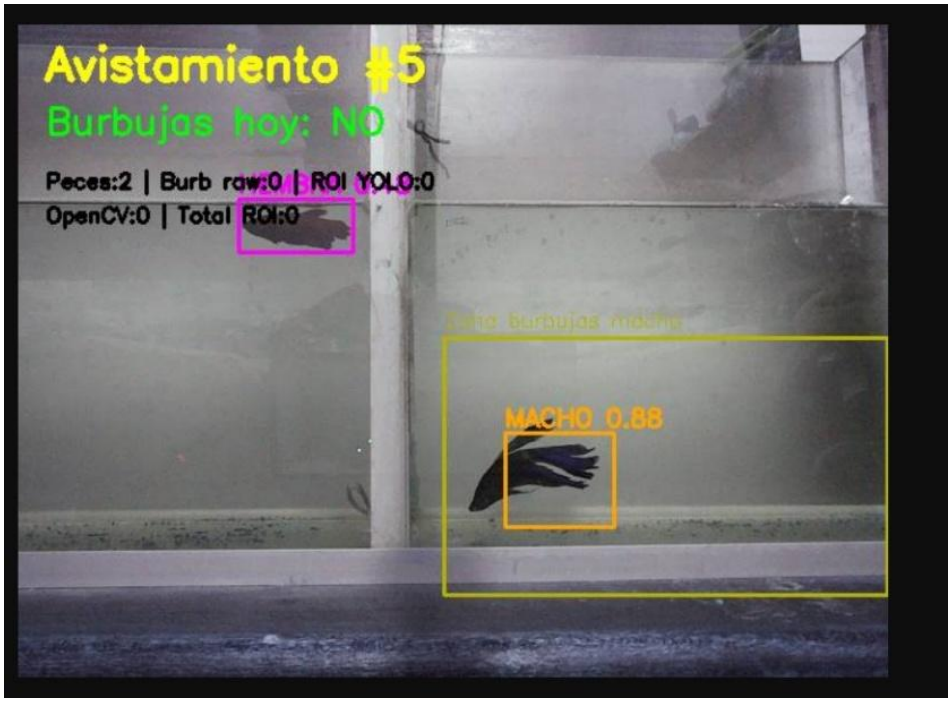


Figura 24

Captura detectando hembra.

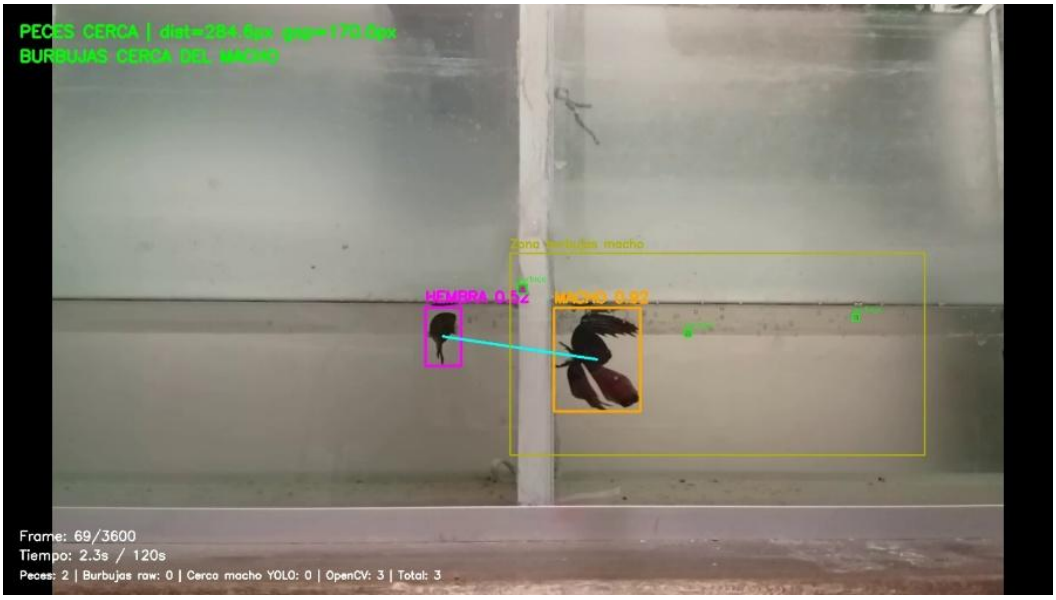


Figura 25

Captura con ambos peces detectados.

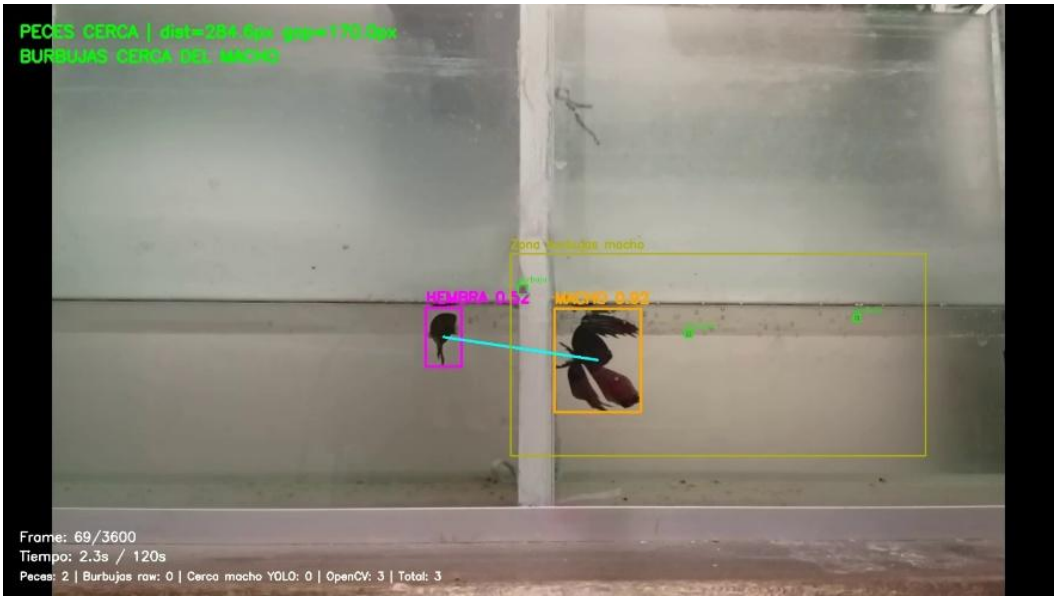


Figura 26

Captura de la interfaz en tiempo real.



6.3. Anexo C. Evidencia de generación de avistamientos y verificación de burbujas

En este anexo se incluyen capturas correspondientes a la generación automática de avistamientos y al proceso de verificación de burbujas realizado por el sistema. En el caso de los avistamientos, se muestran ejemplos en los que el sistema detecta simultáneamente al macho y a la hembra en condición de proximidad, generando un evento identificado con un contador acumulativo. En el caso de la verificación de burbujas, se presentan imágenes almacenadas por el sistema y posteriormente procesadas para determinar la presencia de este patrón visual en la superficie del agua.

La finalidad de este anexo es demostrar que el sistema no solo detecta objetos en video en tiempo real, sino que además interpreta la relación espacial entre los peces y ejecuta un análisis complementario sobre las imágenes almacenadas. Esto aporta evidencia directa del funcionamiento de dos de las funciones más importantes del prototipo: la generación de avistamientos y la identificación de burbujas como criterio asociado al comportamiento reproductivo.

Figura 27

Captura de un avistamiento real, por ejemplo “Avistamiento #161”.



Figura 28

Captura de otra escena de acercamiento.

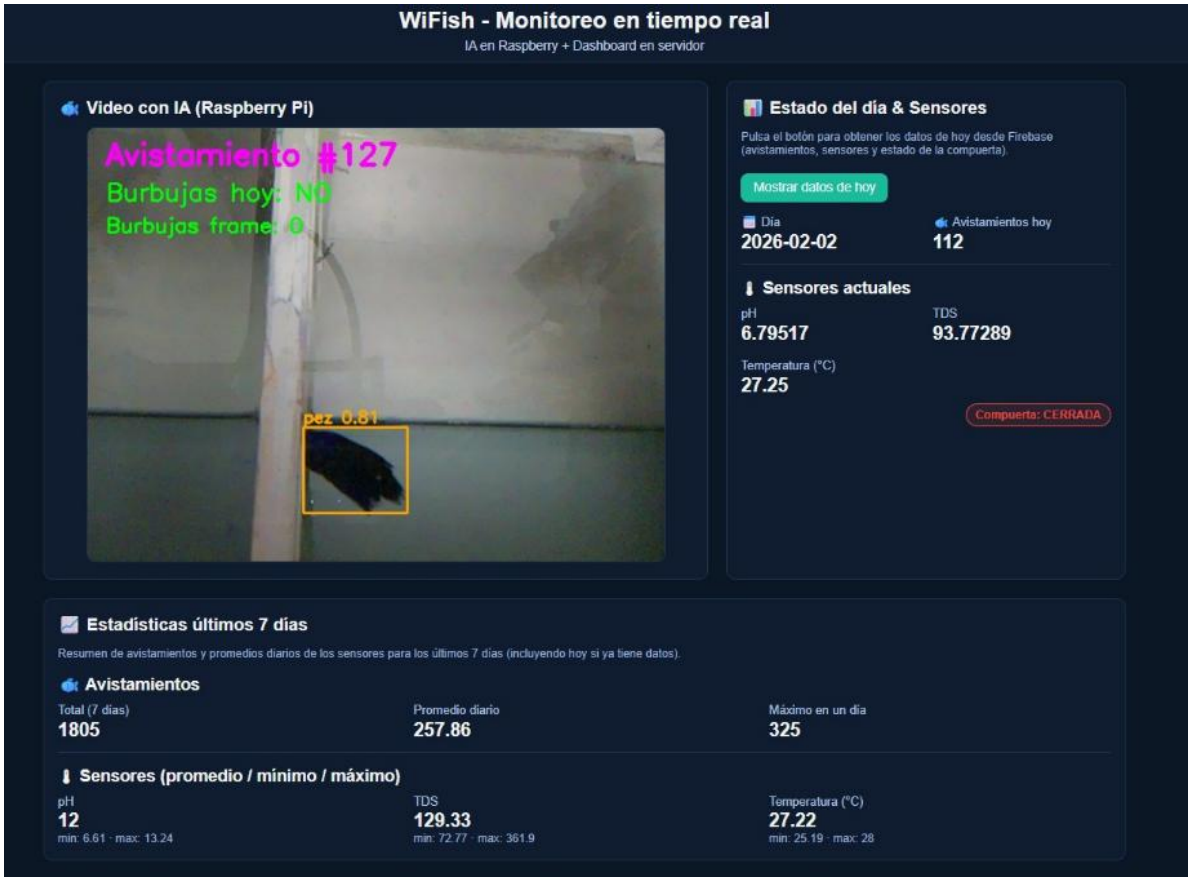
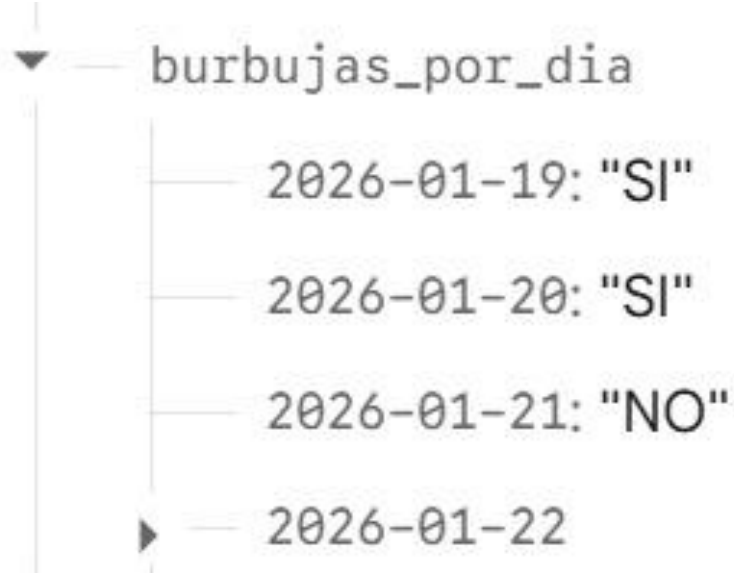


Figura 29

Imagen almacenada con burbujas.



6.4. Anexo D. Evidencia del almacenamiento en Firebase Realtime Database

En este anexo se presenta la evidencia del almacenamiento de datos en Firebase Realtime Database. Las capturas incluidas muestran la estructura de la base de datos utilizada en el sistema, donde se registran los avistamientos acumulados, la verificación de burbujas, el estado de la compuerta y las variables ambientales adquiridas por los sensores, diferenciando entre valores actuales e históricos.

Este anexo tiene como propósito demostrar que el sistema implementado posee una arquitectura IoT funcional, capaz de conservar de forma persistente la información generada durante su operación. La organización de los datos en Firebase respalda la

trazabilidad del prototipo, ya que permite consultar eventos visuales y variables fisicoquímicas a lo largo del tiempo. De esta manera, el proyecto se consolida no solo como una herramienta de detección en tiempo real, sino también como una plataforma de monitoreo con almacenamiento histórico.

Figura 30

Captura de la estructura general de Firebase.



Figura 31

Captura del nodo avistamientos.



Figura 32

Captura del nodo sensores.



Figura 33

Captura de histórico.

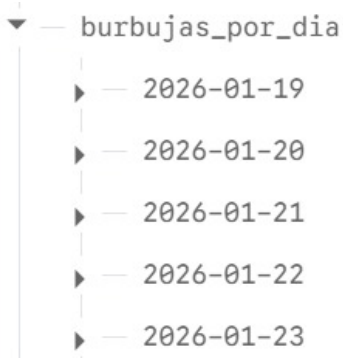
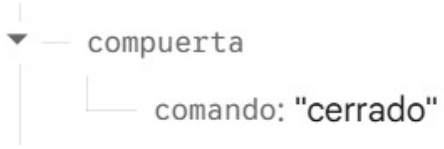


Figura 34

Captura de compuerta.



6.5. Anexo E. Fragmentos relevantes del código fuente

En este anexo se presentan los fragmentos de código más representativos del sistema desarrollado. Se incluyen únicamente las secciones de programación directamente relacionadas con las funciones principales del proyecto, tales como la lectura de sensores, la recepción de datos desde el ESP32, la ejecución del modelo de visión artificial, la lógica de generación de avistamientos, la verificación de burbujas y el almacenamiento de información en Firebase.

El objetivo de este anexo es complementar la descripción técnica del sistema con evidencia del desarrollo de software implementado. No se busca incorporar la totalidad

del código fuente, sino únicamente aquellas partes que resultan más relevantes para comprender el funcionamiento del prototipo. Esta selección permite respaldar la originalidad del desarrollo y facilitar la revisión técnica del proyecto por parte del lector o del tribunal evaluador.

Código ESP32

```
// Lectura de sensores pH/TDS y control de compuerta con L298N

#define PIN_PH   34
#define PIN_TDS  35

#define IN1 26
#define IN2 27
#define ENA 25

unsigned long tiempoAnterior = 0;
const unsigned long intervalo = 300000; // 5 minutos

void setup() {
  Serial.begin(115200);

  pinMode(PIN_PH, INPUT);
  pinMode(PIN_TDS, INPUT);

  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENA, OUTPUT);

  cerrarCompuerta();
}

void loop() {
  if (millis() - tiempoAnterior >= intervalo) {
    tiempoAnterior = millis();

    float ph = leerPH();
    float tds = leerTDS();

    // Envío de datos hacia Raspberry Pi por UART
    Serial.print("{\"ph\":");
```

```
    Serial.print(ph);
    Serial.print(",\"tds\":");
    Serial.print(tds);
    Serial.println("}");
  }

  // Recepción de comandos desde Raspberry Pi
  if (Serial.available()) {
    String comando = Serial.readStringUntil('\n');
    comando.trim();

    if (comando == "ABRIR") {
      abrirCompuerta();
    }

    if (comando == "CERRAR") {
      cerrarCompuerta();
    }
  }
}

float leerPH() {
  int valor = analogRead(PIN_PH);
  float voltaje = valor * (3.3 / 4095.0);

  // Fórmula aproximada de conversión
  float ph = 7 + ((2.5 - voltaje) / 0.18);

  return ph;
}

float leerTDS() {
  int valor = analogRead(PIN_TDS);
  float voltaje = valor * (3.3 / 4095.0);

  // Conversión aproximada a ppm
  float tds = voltaje * 500;

  return tds;
}

void abrirCompuerta() {
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  analogWrite(ENA, 180);
```

```
  delay(3000);

  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
}

void cerrarCompuerta() {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  analogWrite(ENA, 180);
  delay(3000);

  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
}
```

Código E2. Código simplificado de Raspberry Pi

```
# Código simplificado Raspberry Pi
# Captura de video, detección YOLOv11, avistamientos y Firebase

import cv2
import time
import math
import serial
import json
from ultralytics import YOLO
import firebase_admin
from firebase_admin import credentials, db

# Inicializar Firebase
cred = credentials.Certificate("clave_firebase.json")
firebase_admin.initialize_app(cred, {
    "databaseURL": "https://tesis-peces-default-rtdb.firebaseio.com/"
})

# Inicializar modelo YOLOv11
modelo = YOLO("best.pt")

# Inicializar cámara
camara = cv2.VideoCapture(0)
```

```
# Comunicación con ESP32
esp32 = serial.Serial("/dev/ttyUSB0", 115200, timeout=1)

contador_avistamientos = 0
umbral_distancia = 180
ultimo_avistamiento = 0
tiempo_espera = 10   # segundos para evitar conteos repetidos

def distancia(p1, p2):
    return math.sqrt((p2[0] - p1[0])**2 + (p2[1] - p1[1])**2)

def centro_caja(caja):
    x1, y1, x2, y2 = caja
    return ((x1 + x2) / 2, (y1 + y2) / 2)

while True:
    ret, frame = camara.read()

    if not ret:
        continue

    resultados = modelo(frame, conf=0.50)

    macho = None
    hembra = None
    burbujas = False

    for resultado in resultados:
        for box in resultado.boxes:
            clase = int(box.cls[0])
            confianza = float(box.conf[0])
            x1, y1, x2, y2 = box.xyxy[0].tolist()

            nombre_clase = modelo.names[clase]

            if nombre_clase == "macho":
                macho = centro_caja((x1, y1, x2, y2))

            if nombre_clase == "hembra":
                hembra = centro_caja((x1, y1, x2, y2))

            if nombre_clase == "burbujas":
                burbujas = True

            # Dibujar detección en pantalla
```

```
        cv2.rectangle(frame, (int(x1), int(y1)), (int(x2), int(y2)), (0, 255, 0), 2)
        cv2.putText(frame, f"{nombre_clase} {confianza:.2f}",
                    (int(x1), int(y1) - 10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0, 255, 0), 2)

# Generación de avistamiento
if macho is not None and hembra is not None:
    d = distancia(macho, hembra)

    if d < umbral_distancia and time.time() - ultimo_avistamiento > tiempo_espera:
        contador_avistamientos += 1
        ultimo_avistamiento = time.time()

        nombre_imagen = f"avistamiento_{contador_avistamientos}.jpg"
        cv2.imwrite(nombre_imagen, frame)

        db.reference("avistamientos").set(contador_avistamientos)

        print(f"Avistamiento #{contador_avistamientos}")

# Registro de burbujas
if burbujas:
    db.reference("burbujas").set(True)

# Lectura de datos enviados por ESP32
if esp32.in_waiting > 0:
    linea = esp32.readline().decode("utf-8").strip()

    try:
        datos = json.loads(linea)

        db.reference("sensores/actual").set({
            "ph": datos["ph"],
            "tds": datos["tds"],
            "timestamp": time.strftime("%Y-%m-%d %H:%M:%S")
        })

    except:
        pass

cv2.imshow("Video con IA - Raspberry Pi", frame)

if cv2.waitKey(1) & 0xFF == ord("q"):
    break
```

```
camara.release()
cv2.destroyAllWindows()
```

Figura 35

Almacenamiento de imágenes.



Figura 36

Bot de Telegram.



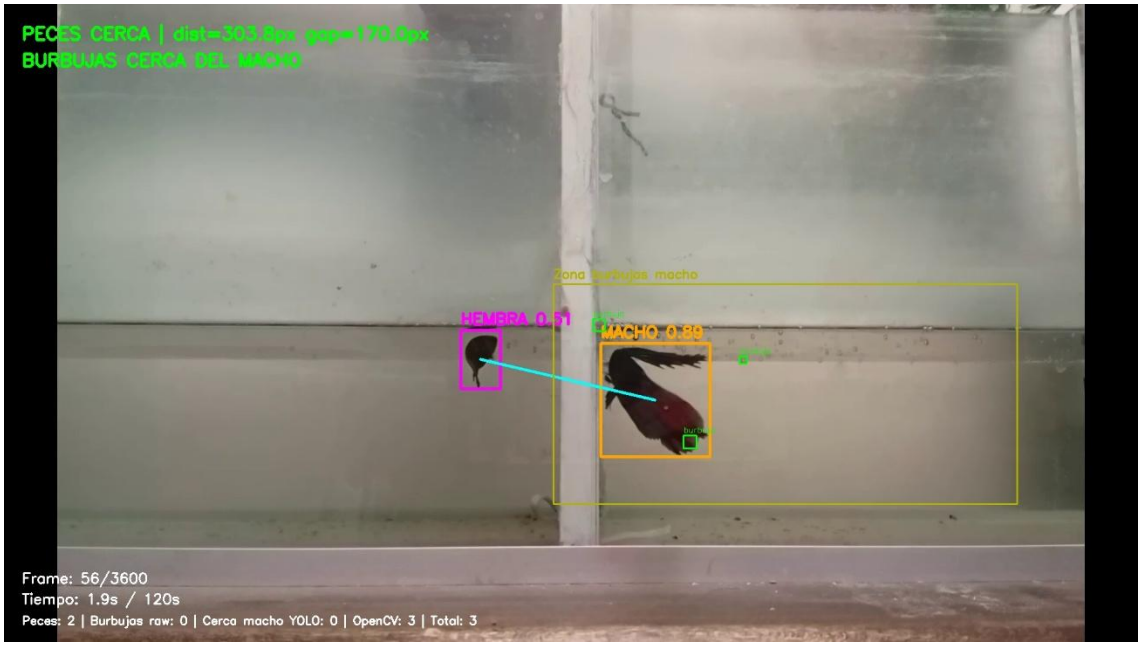
La lógica de proximidad entre el pez Betta macho y la hembra se calculó mediante dos criterios: la distancia euclidiana entre los centroides de las cajas delimitadoras y la

separación horizontal entre ambas detecciones. La distancia entre centroides se expresa mediante la Ec. 2, en la siguiente imagen:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Figura 37

Imagen de avistamiento



Ecuación 2. Distancia euclidiana entre los centroides del macho y la hembra.

Donde x_1, y_1corresponden al centroide de la detección de la hembra, x_2, y_2corresponden al centroide de la detección del macho, y $d$representa la distancia entre ambos peces en píxeles. Para el ejemplo mostrado, se consideran los centroides aproximados:

$$C_h = (738{,}552)$$

$$C_m = (998{,}709)$$

Aplicando la Ecuación 2:

$$d = \sqrt{(998 - 738)^2 + (709 - 552)^2}$$

$$d = \sqrt{260^2 + 157^2}$$

$$d = \sqrt{67600 + 24649}$$

$$d = \sqrt{92249}$$

$$d \approx 303.8\, px$$

Por lo tanto, la distancia entre el pez Betta macho y la hembra fue de aproximadamente 303.8píxeles, valor que coincide con el mostrado por el sistema en la imagen.