Universidad Nacional Autónoma de México

Posgrado en Ciencia e Ingeniería de la Computación

Sistema prototipo de monitoreo subacuático automático de peces por visión estereoscópica y aprendizaje profundo

T E S I S

QUE PARA OPTAR POR EL GRADO DE

Maestro en Ciencia e Ingeniería de la Computación

PRESENTA:
Ing. Héctor Carlos Aranda Martínez

Tutor Principal:
Dra. Nidiyare Hevia Montiel
Unidad Académica del Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas en el Estado de Yucatán

Mérida, Yucatán
Diciembre 2021

Agradecimientos

A la Universidad Nacional Autónoma de México y al Programa de Posgrado en Ciencia e Ingeniería de la Computación, por haberme provisto de una formación integral.

Al CONACyT por la beca proporcionada para realizar mis estudios de maestría en el periodo de agosto de 2019 a agosto de 2021.

Al apoyo económico brindado por el Programa de Apoyo a Proyectos de Investigación e Innovación Tecnológica (PAPIIT) del proyecto PAPIIT UNAM IT100220 otorgado a la Dra. Nidiyare Hevia Montiel.

Al Programa de Apoyo a los Estudios de Posgrado (PAEP) por el apoyo otorgado para la asistencia al XIII Congreso Mexicano de Inteligencia Artificial, donde se presentaron los resultados de este trabajo.

A mi directora de tesis, la Dra. Nidiyare Hevia Montiel, por el apoyo, los consejos y la motivación que me impulsaron para concluir este proyecto.

Al Dr. José Antonio Neme Castillo, el Dr. Jorge Luis Pérez González, el Dr. Julián Bravo Castillero y el Dr. Xavier Chiappa Carra por aceptar ser parte de mi jurado y por sus observaciones y aportaciones a este trabajo.

Al Campus Sisal por permitir la realización de experimentos en sus instalaciones; en particular al M. en C. Rafael Eduardo Pacheco Góngora por su interés y apoyo en mis experimentaciones.

Al Laboratorio Universitario de Cómputo de Alto Rendimiento (LUCAR), en particular al Ing. Adrian Durán Chavesti por el apoyo en el uso de los servidores.

A David Espinosa por siempre estar al pendiente y permitirme utilizar su infraestructura y sus peces para realizar mis experimentos.

A mis profesores del posgrado, por siempre hacer su mejor trabajo a pesar de las dificultades.

A papá, mamá, Fer y Fito, gracias por ser ese lugar al que siempre quiero regresar.

Para Chazita.
Eres ese impulso que necesito
para conquistar el mundo.

Resumen

El monitoreo subacuático se ha desarrollado desde que se tuvo la capacidad de "ver" debajo del agua, utilizando cámaras sumergibles. En las últimas dos décadas se ha potenciado su uso a nivel acuícola, para hacer más eficientes las actividades de pesca y cultivo a través del monitoreo. De entre las técnicas utilizadas más importantes destacan el sonar y la visión por computadora. Si bien, el sonar es útil para obtener información del entorno, es de muy baja resolución y por su naturaleza genera datos con mucho ruido. Utilizar cámaras que funcionen en el espectro visible ha sido una opción viable y, gracias a los avances tecnológicos, también una opción económica que permite obtener datos de alta calidad en cuestión de resolución espacial.

En este trabajo se presenta la propuesta de un prototipo de monitoreo subacuático utilizando visión estereoscópica, que permite obtener mayor información de los peces monitoreados, calculando tanto su posición en el espacio como su longitud. El prototipo propuesto está hecho de diferentes módulos que realizan tareas específicas, los cuales son: obtener datos estereoscópicos por medio de las cámaras (video e imagen); identificar y localizar objetos en cada imagen individual por aprendizaje profundo, utilizando una red neuronal convolucional tipo YOLO (*You Only Look Once*, una de las arquitecturas más utilizadas en tareas de detección de objetos); hacer la correspondencia de los objetos en cada par de imágenes utilizando información estereoscópica; calcular las posiciones tridimensionales de puntos de interés de los objetos mediante geometría epipolar; y finalmente hacer el cálculo de las longitudes de los peces detectados.

El prototipo se ha desempeñado satisfactoriamente en ambiente controlado, utilizando *phantoms* de peces (figuras rígidas) y controlando su posición espacial. Después de la realización y análisis de los resultados, se mostró que el prototipo logra hacer estimaciones de longitud haciendo uso de *phantoms* con un error porcentual de hasta el 2.5 % bajo condiciones controladas. Sin embargo, como se esperaría, el prototipo disminuyó su capacidad de detección en ambientes semicontrolados, aunque las veces en las que la detección era buena las estimaciones también lo fueron.

Este trabajo pretende ser el inicio del desarrollo de un sistema de monitoreo más robusto, confiable y fácil de utilizar. Los resultados obtenidos son satisfactorios, y permiten proponer acciones a desarrollar para mejorarlo, como la solución de los problemas encontrados en la detección y localización de boca y aleta; así, se espera que sea posible su uso en ambientes no controlados (como arrecife o petenes).

Abstract

Underwater monitoring has been developed since the ability to "see" underwater, using submersible cameras. In the last two decades, its use in aquaculture has increased to make fishing and farming activities more efficient through monitoring. Among the most important techniques used are sonar and computer

vision. Although sonar is useful for obtaining information from the environment, it is of very low resolution and by its nature generates data with a lot of noise. Using cameras that operate in the visible spectrum has been a viable option and, thanks to technological advances, also an economical option that allows obtaining high quality data in terms of spatial resolution.

This work presents the proposal of an underwater monitoring prototype using stereoscopic vision, which allows to obtain more information of the monitored fish, calculating both its position in space and its length. The proposed prototype is made of different modules that perform specific tasks, which are: obtaining stereoscopic data by means of the cameras (video and image); identifying and locating objects in each individual image by deep learning, using a convolutional neural network type YOLO (You Only Look Once, one of the most used architectures in object detection tasks); matching the objects in each pair of images using stereoscopic information; calculating the three-dimensional positions of points of interest of the objects by means of epipolar geometry; and finally calculating the lengths of the detected fish.

The prototype has performed satisfactorily in controlled environment, using fish *phantoms* (rigid figures) and controlling their spatial position. After the realization and analysis of the results, it was shown that the prototype is able to make length estimations using *phantoms* with a percentage error of up to 2.5 % under controlled conditions. However, as would be expected, the prototype decreased its detection capability in semi-controlled environments, although the times when detection was good the estimates were also good.

This work is intended to be the beginning of the development of a more robust, reliable and user-friendly monitoring system. The results obtained are satisfactory, and allow us to propose actions to be developed to improve it, such as the solution of the problems found in the detection and location of mouth and fin; thus, it is expected that its use in uncontrolled environments (such as reefs or petenes) will be possible.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Las ciencias de la computación son útiles, e inclusive necesarias, en casi todos los campos del conocimiento. Esta versatilidad deriva de la facilidad que tienen los expertos en este campo de modelar un problema, es decir, generar un algoritmo que lo describa para dar una posible solución. Actualmente, el aumento en la capacidad de cómputo y la facilidad económica y técnica de adquirir datos, ha permitido que áreas, como la inteligencia artificial, ahora sean tan activas y resuelvan eficientemente grandes problemas de la industria. Todo este desarrollo ha ido avanzando a través del tiempo, generando tecnologías cada vez más avanzadas que hasta cierto punto han desafiado las creencias de lo que podríamos hacer con una computadora.
A continuación se explicará de manera general los temas principales sobre los que está basado este trabajo. Iniciando con el concepto de visión computacional hasta el de aprendizaje computacional para la detección automática.

1.1. Monitoreo subacuático

Monitorear es una actividad que nos permite dar seguimiento a un evento, o un conjunto de eventos, para posteriormente generar análisis a partir de los datos obtenidos. La forma de realizar este monitoreo depende de: 1) las características del evento; y 2) las características del entorno. En un entorno subacuático existen diversos eventos de interés que pueden monitorearse, como los niveles de pH, la concentración de oxígeno, el crecimiento de un arrecife de coral, el comportamiento de un banco de peces en su temporada reproductiva, entre muchos más. Hablando de un entorno más controlado, como en el sector acuícola, la importancia del monitoreo aumenta todavía más. Aquí la importancia del monitoreo es mayor, pues existen diferentes variables que permiten hacer más eficiente la producción de peces, como el seguimiento del tamaño, masa, salud, pH, oxigenación, entre otros.
Existe una gran variedad de técnicas para dar seguimiento a las variables antes mencionadas, por supuesto que algunas se desempeñan mejor que otras, y su

implementación recae en el criterio del interesado. Además, podemos encontrar una gran industria que ha desarrollado productos especializados en resolver estor problemas. A medida que la tecnología va avanzando, nuevas y mejores técnicas y métodos son generados, permitiendo hacer más eficiente esta tarea y extender los límites de la misma. Por ejemplo, gracias al desarrollo de los que ahora se conoce como el Internet de las Cosas (o *Internet of Things*), se han diseñado redes de sensores subacuáticos que permiten conocer, con gran precisión, las condiciones fisico-químicas en una región específica [1]. De forma similar, nuevos desarrollos en el área de aprendizaje automático, así como en visión computacional, han permitido en la última década el desarrollo de nuevos métodos no invasivos para llevar el monitoreo de peces de una forma rápida y eficiente. A lo largo de este trabajo de investigación se explicará la propuesta de una nueva técnica basada en aprendizaje automático y visión computacional.

1.2. Visión Computacional

Existe la coincidencia en decir que el fin último de la visión computacional es que las computadoras puedan, al menos, realizar las tareas de visión que los humanos hacemos. No debemos confundir este concepto con el procesamiento de imágenes, donde el interés es principalmente hacer un procesamiento sobre los píxeles de una imagen, generalmente a niveles de intensidad, obteniendo una imagen con características derivadas de la original.
Para poder continuar, debe quedar claro que una imagen digital puede representarse como un arreglo de números que representan un nivel de intensidad luminosa. Coloquialmente se le conoce a cada elemento de este arreglo como "píxel". Así en una imagen $I(x, y)$, acceder a un píxel específico es tan fácil como indicar las coordenadas (x, y) en las que se encuentra.

Fig. 1.1. Filtro gaussiano aplicado a la imagen original. En cada una se considera un valor diferente del parámetro *sigma*, el cual permite controlar el grado de difuminación. Imagen original obtenida de: fuente propia.

Un claro ejemplo de procesamiento de imágenes es el uso de filtros para resaltar

características de estas. Algunos de estos filtros ya están definidos y son ampliamente utilizados, como el filtro de gaussiano (figura 1.1). Otros más pueden depender de la imagen, como la resolución espacial o el tipo de frecuencias sobre las que se deseen trabajar. Para trabajar con este tipo de filtros se utiliza la **convolución**, que se trata de una operación matemática descrita por la siguiente ecuación [2]

$$H(x,y) = \sum_{i=-m}^{m} \sum_{j=-m}^{m} k(i,j) I(x-i, y-j) \tag{1.1}$$

donde H es la nueva imagen, y k es el filtro de dimensiones $2m+1$. Con esta operación, utilizando diferentes filtros, podemos extraer información importante contenida en las imágenes originales. Por ejemplo, los filtros para extracción de bordes (como el de Prewitt, ecuación 1.2 o Sobel, ecuación 1.3) realizan una diferenciación entre los niveles de intensidad de píxeles vecinos, resaltando los puntos donde las diferencias son grandes. De esta manera, cuanto mayor sea la diferencia de intensidad, la evaluación de la convolución en dicha posición dará valores altos. Esto permite identificar bordes dentro de la imagen, como se puede apreciar en la figura 1.2. En estos casos en particular, los bordes son detectados en orientaciones en específico, ya sea en el eje x o en el eje y.

$$Prewitt_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \qquad Prewitt_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \tag{1.2}$$

$$Sobel_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -1 \\ 1 & 0 & -2 \end{bmatrix} \qquad Sobel_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{1.3}$$

Además de la aplicación de filtros, también es común desear trabajar con los niveles de intensidad de cada píxel. Una de las mejores herramientas para visualizar su distribución es el histograma de intensidades (ver figura 1.3). Este histograma tiene un gran número de aplicaciones, de entre las cuales destacaremos dos: el ajuste de los niveles de intensidad de la imagen, y la segmentación por niveles de intensidad. El primer caso es útil cuando se desee resaltar elementos que no son fácilmente visibles en la imagen, ya sea porque la imagen sea muy "clara" o muy "oscura". El segundo caso funciona muy bien cuando los elementos que se desean segmentar contrastan con el fondo, es decir, que hay una clara diferencia entre los niveles de intensidad entre ambas partes. Para trabajar con el histograma es común utilizar la información en escala de grises de una imagen, aunque también suele haber algunas aplicaciones de histogramas en diferentes espacios de color.

Además de las anteriores, hay un gran número de operaciones que se pueden usarse para procesar una imagen. Sin embargo, a pesar de transformar la información de la imagen, todavía no extraemos de ella lo que es relevante. Aquí es donde entra la visión computacional, que es definida por algunos como la capacidad de una máquina (computadora) para darle sentido a lo que hay en

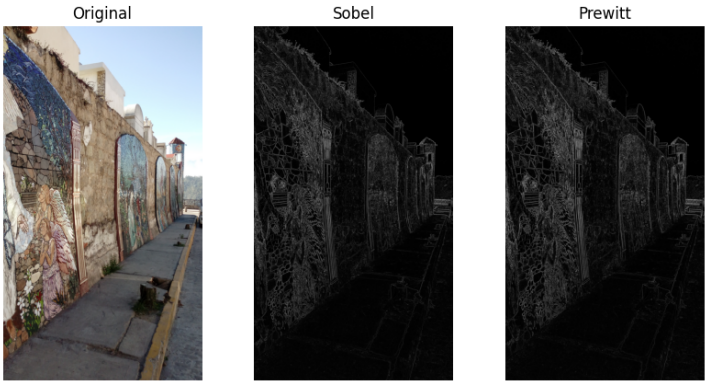


Fig. 1.2. Extracción de los bordes de la imagen a través de los filtros de Sobel y Prewitt. Para ello se trabaja con la versión de la imagen en escala de grises, después se realiza la convolución con cada uno de los filtros.

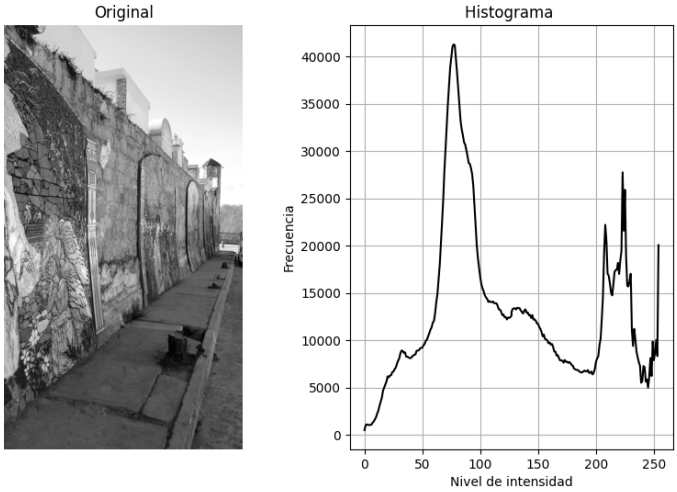


Fig. 1.3. Histograma de intensidades en la imagen en escala de grises. También es posible generar un histograma por cada canal de color, en caso de ser necesario.

una imagen, mediante la obtención de información de importancia de la misma. A medida que nuestro conocimiento sobre el funcionamiento de nuestra propia visión avanza, podemos diseñar algoritmos de visión computacional más complejos. A su vez también podemos entender mejor nuestra visión gracias al ver

cómo funcionan los algoritmos que hemos diseñado [3].
Esta rama de la inteligencia artificial es relativamente reciente [3], su desarrollo comenzó en la segunda mitad del siglo XX (y en la actualidad parece que todavía falta mucho por hacer). Por ejemplo, lo que podría ser algo tan simple para una persona, como identificar objetos en una imagen, es una tarea titánica para una computadora. Analizando este problema a profundidad nos damos cuenta de que no tiene una solución sencilla, pues existe una gran cantidad de factores que son necesarios conocer para siquiera poder identificar un solo objeto. Hace 55 años se creía que este problema podría ser resuelto en lo que se conoce como "*The Summer Vision Project*" [4], donde el objetivo era sencillo: crear un sistema de visión que identificara objetos en una imagen y los ligara a un vocabulario de objetos conocidos. Lamentablemente el resultado no fue el esperado.
Concretamente, ¿a qué tipo de operaciones se les puede considerar como parte de la visión por computadora? A continuación se enlistan algunas:

- **Clasificación**. La clasificación se encarga de corresponder la información de una imagen con una etiqueta. Estas etiquetas comúnmente son nombres de objetos conocidos, aunque también pueden ser conceptos más complejos como sentimientos, lugares, momentos, etc.

- **Detección de objetos**. El objetivo principal es poder conocer las coordenadas donde está contenido un objeto de interés. Un ejemplo muy común es el de los detectores de rostros. Aquí el algoritmo se especializa en buscar rostros dentro de una imagen y devolver las coordenadas de la región que contiene cada una de las caras encontradas.

- **Reconstrucción de escena**. Obtener información tridimensional a partir de una imagen es el último paso para poder comprender la información de una escena [5]. Esto actualmente se puede hacer mediante técnicas de visión multivista, es decir, utilizando más de una fuente de información (cámara).

1.3. Aprendizaje automático

Una tarea relevante que la visión computacional intenta resolver, es la de la detección de objetos en imágenes. Para una persona, detectar un objeto y localizarlo en una imagen es una tarea relativamente sencilla, como se ha mencionado antes, gracias a nuestra gran capacidad de aprendizaje. Pongamos de ejemplo el problema de detección del pez de la imagen (ver figura 1.4), la pregunta principal es, ¿cómo identificar la región en la que se encuentra el pez? Para poder detectar un objeto es necesario comprender la información que contiene la imagen, tal como formas, texturas, intensidades, colores, etc.; entonces es natural que los primeros pasos para realizar esta tarea conlleven el análisis de algunas de estas características, donde nos podemos encontrar técnicas estadísticas (análisis de componentes principales o PCA, K-medias) o de aprendizaje automático (redes neuronales artificiales, bosques de decisión aleatorios).

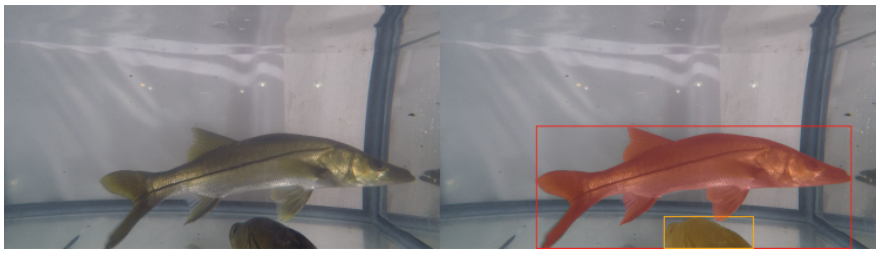


Fig. 1.4. Ejemplo del resultado esperado de la detección de peces en una imagen, donde se muestran las regiones que contienen completamente a los peces detectados. Imagen original obtenida en colaboración con el M. en C. Rafael Eduardo Pacheco Góngora de la Unidad Académica Sisal, en la UNAM Yucatán.

Un tema que tiene especial relevancia es el de la Inteligencia Artificial, pues una gran cantidad de dispositivos de uso cotidiano comienzan a tener algoritmos inteligentes, desde celulares hasta autos. Sin embargo, una gran parte de estos algoritmos pertenecen a una sola rama de lo que conocemos como inteligencia artificial: el aprendizaje automático. En palabra sencillas podríamos decir que el aprendizaje automático se encarga de asimilar información relevante en un conjunto de datos, pero eso sería simplificar injustamente la complejidad de procesos que pueden ser utilizados para llevar a cabo esta tarea.
Este aprendizaje puede ser representado de diferentes maneras, ya sea mediante pesos para un proceso de ponderación, o umbrales que permitan discriminar cierta información de otra. Lo más importante es poder generar un modelo que, habiendo aprendido adecuadamente de los datos, nos permita predecir a partir de datos nuevos. El ejemplo más sencillo es el de una regresión lineal de un conjunto de datos. Al graficar los datos como puntos en un plano cartesiano, podemos trazar una línea de tal forma que la distancia promedio de cada punto a la línea sea la más pequeña. Esta línea representa el modelo que podemos utilizar para estimar el valor resultante de un nuevo dato que el modelo no ha 'visto" (es decir que no ha formado parte del conjunto de datos para estimar los parámetros de la línea). La forma en la que se genera el modelo y los métodos estadísticos disponibles hacen que existan una gran cantidad de técnicas de aprendizaje automático, entre ellas se destacan:

- **Bosques de decisión aleatorios (*Random Forest*) [6].** Se basan en un conjunto de árboles de decisión que generan hiperplanos para separar los datos a partir de umbrales que aprenden por un proceso de entrenamiento. Lo interesante es que el resultado del procesamiento de nuevos datos no depende de un solo árbol, sino del conjunto de decisiones que tomaron todos los árboles, llegando a un consenso y devolviendo el resultado más probable. Estos algoritmos son muy buenos en tareas de clasificación, y aunque pueden utilizarse para hacer regresión, hay que realizar ciertas consideraciones para que funcionen adecuadamente.

- **Máquinas de soporte vectorial (*Support Vector Machines*) [7].** Se basan en un cambio del espacio de los datos (aumento de dimensionalidad), donde se espera encontrar un hiperplano que permita separar los mismos de manera eficiente. Esto permite que problemas no linealmente separables en su espacio, puedan serlo en un espacio de más dimensiones.
- **Redes de neuronas artificiales (*Artificial Neural Network*).** Actualmente es una de las técnicas más utilizadas tanto a nivel académico como no académico, gracias a su versatilidad y capacidad de aprendizaje [8]. Las redes de neuronas, o redes neuronales, están formadas de una entidad matemática conocida como perceptrón, que a pesar de realizar una operación relativamente sencilla, cuando funcionan en conjunto han demostrado una alta capacidad de ajustarse a datos altamente no lineales.

De entre los métodos mencionados, en este trabajo se utilizó una red neuronal convolucional, gracias a su gran velocidad de procesamiento y a su buen desempeño, cuyo funcionamiento se basa en las redes de perceptrones que serán explicadas a continuación.

1.3.1. Redes de perceptrones

A mediados del siglo pasado Rosenblatt [9] planteó lo que sería el fundamento de una entidad matemática que medio siglo más tarde dominaría el campo del aprendizaje automático: el perceptrón. Su propuesta se basó en el comportamiento de una neurona cerebral, donde la neurona recibe estímulos como entradas ($S_i \in \boldsymbol{S}$), son procesados ($P(\boldsymbol{S})$) y evaluados por una "función de activación" ($A(P(\boldsymbol{S}))$) que decidirá si la neurona se activa o no, ver figura 1.5).

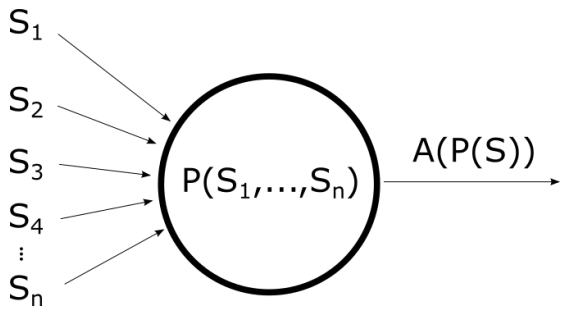


Fig. 1.5. Representación esquemática del perceptrón de Rosenblatt. Cada uno de los estímulos está ponderado por un "peso" de tal manera que $S_i = w_i x_i$, donde x_i es el valor de entrada y w_i el peso por el que se multiplicará tal valor.

Este modelo es relativamente sencillo, y ha demostrado poder aproximar funciones lineales simples, aunque no pueda hacerlo con funciones no lineales. El por qué pueden hacer estos ajustes recae directamente en los pesos con los que se ponderan los valores de entrada. Al cambiarlos, el comportamiento del perceptrón cambia, y es capaz de "aprender" a ajustar la función requerida. Poco

tiempo fue necesario para encontrar que usar una red de dos o más perceptrones, simulando la interacción que las neuronas tienen en el cerebro, permite ajustar funciones no lineales cada vez más complejas (a medida que aumentamos el número de perceptrones). Aumentar el número de neuronas traía grandes ventajas, aunque también requería ajustar un mayor número de parámetros (pesos), lo que necesita de mayor potencia de cómputo, así como algoritmos eficientes para esto. A la modificación de los pesos de una red se le llama entrenamiento. El entrenamiento de grandes redes de perceptrones limitaban su uso, hasta que en 1986 Rumelhart *et al* [10] describieron el algoritmo *Backpropagation* en el cual proponían usar un método de corrección de los parámetros de ponderación (pesos) por medio de la evaluación del error a la salida de la red y corrigiéndolos según su aportación al error global.
Estas redes de neuronas (redes neuronales de ahora en adelante) han sido de gran interés académico, principalmente por su sorprendente capacidad de generalización, su facilidad de implementación y su versatilidad. Este tipo de algoritmos son generalmente considerados como cajas negras, ya que cuentan con una cantidad enorme de parámetros, lo que no hace sencilla la tarea de saber qué es lo que pasa en su interior. A pesar de esto han demostrado resolver problemas computacionales que de otras formas no habían tenido una solución satisfactoria.
Trabajar con imágenes en una red de perceptrones es muy poco intuitivo, ya que, dada la estructura de la red, se debe convertir la matriz que contiene los elementos de niveles de intensidad de la imagen en un vector; cada elemento de este vector será la entrada de cada una de las neuronas de la primera capa. A diferencia de otros tipos de datos, las imágenes tienen características que dependen de la posición espacial de sus elementos, es decir, que es de importancia conocer la vecindad de los píxeles que se están analizando. Por ejemplo, para detectar una esquina es necesario hacer evaluaciones en píxeles adyacentes al que se está evaluando, así mediante la extracción de características en la vecindad se puede determinar si el píxel en cuestión representa una esquina o no. La forma más común de procesar imágenes para extraer estas características es mediante filtros (explicados en la sección anterior), y la operación realizada con estos filtros se conoce como convolución.

1.3.2. Redes neuronales convolucionales

En 1980 Fukushima [11] propuso una nueva arquitectura de red neuronal, inspirándose en el funcionamiento del sistema nervioso visual, donde el mismo sistema se autoorganiza y puede aprender patrones geométricos (tarea fundamental en el procesamiento de imágenes). La idea es la siguiente: se alimenta la imagen a la red que tiene como capa de entrada neuronas organizadas en un arreglo rectangular equivalente al tamaño de la imagen; cada capa es procesada por c filtros de convolución de tamaño $k \times k$, cuyos parámetros son calculados mediante el proceso de aprendizaje, y generan una nueva capa de c canales. A diferencia de otro tipo de redes neuronales, estas toman en cuenta la relación espacial de los elementos de entrada, es por esto que las capas son rectangulares y se utiliza la convolución, operación que trabaja a nivel local del píxel. A este

tipo de redes se les conoce como Redes Neuronales Convolucionales (o CNN por sus siglas en inglés).
Se ha encontrado que cada una de las capas de la CNN extrae diferentes características de la imagen original, tal como bordes, texturas, esquinas, etc. (figura 1.6). A medida que aumenta la profundidad (el número de capas de procesamiento), las capas se especializan en patrones específicos de la imagen original. La extracción de estas características se realiza a través de los filtros que no están predefinidos (o no suelen estarlo) y sus parámetros son aprendidos a través del entrenamiento.

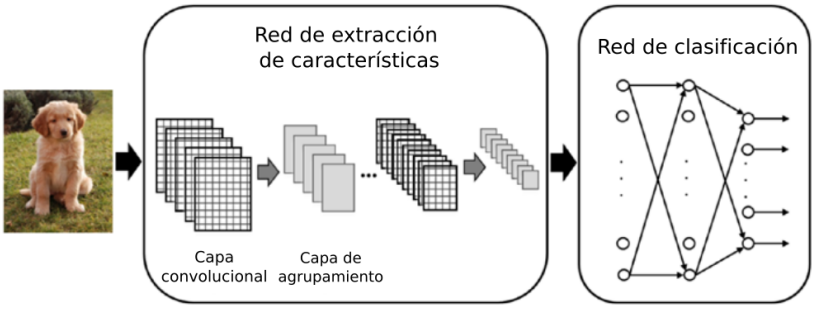


Fig. 1.6. Esquematización del funcionamiento de una CNN: obtenida y traducida de [12].

Esta es la base del funcionamiento de las redes neuronales convolucionales. Ya que se organizan en múltiples capas y que tienen una gran cantidad de parámetros entrenables, estos algoritmos son considerados como parte del paradigma del *deep learning*, o aprendizaje profundo. En el capítulo 3 se hablará en profundidad sobre las características y funcionamiento de una CNN.

1.4. La cámara

Todos conocemos lo que es una cámara. Actualmente es un dispositivo que forma parte de nuestra vida cotidiana; un aparato físico que podemos ver y tocar. En visión computacional nos es indiferente la forma física y características de las cámaras, lo que es importante es cómo se puede representar matemáticamente. Por la naturaleza del funcionamiento de las cámaras y sus características (como sensores o lentes), tenemos una gran cantidad de tipos. Lo que nos interesa es poder tener un modelo que nos permita, cambiando sus parámetros, representar cada una de ellas. Para esto hay que definir los elementos esenciales que las componen, como: distancia focal, eje óptico, centro óptico, etc.
Por la sencillez de su fundamento y funcionamiento, y además que ha demostrado funcionar para la gran mayoría de las cámaras actuales, el modelo de la cámara *pinhole* es el más utilizado. Su fundamento es muy sencillo: la luz captada de una escena pasa a través de un pequeño orificio (*pinhole*), estos rayos

chocan en una superficie proyectando una imagen invertida de la escena. Este efecto era utilizado en lo que anteriormente se conocía como *camera oscura*, que se trata de un cuarto completamente oscuro con un pequeño orificio en uno de sus lados que proyecta una imagen en una de las paredes. Algo similar sucede con las cámaras actuales, aunque también utilizan lentes para mejorar la forma en la que los rayos se proyectan, en este caso, en el sensor.

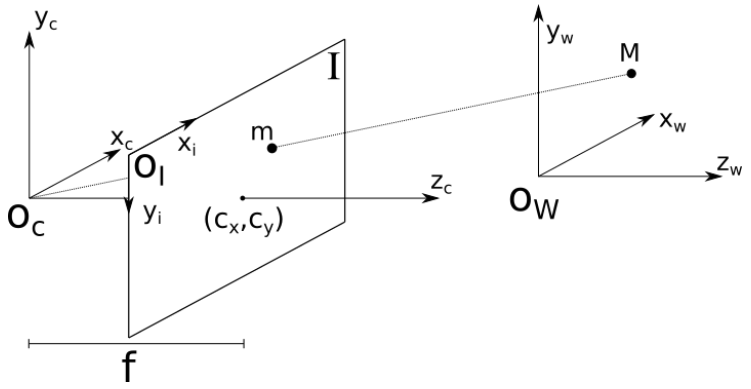


Fig. 1.7. Representación del modelo *pinhole* y la proyección del punto espacial M en el plano de la imagen **I**, identificado como m. La distancia focal está identificada por f, y el centro óptico por las coordenadas (c_x, c_y). Se especifican tres sistemas coordenados: O_I es el plano de la imagen proyectada, O_C es el sistema de la cámara y O_W es el sistema del mundo.

El modelo *pinhole* está esquematizado en la figura 1.7. A primera vista podemos apreciar que contamos con tres sistemas coordenados: el del espacio O_W (o mundo), el de la cámara O_C y el de la imagen O_I. En el mundo real, los objetos se encontrarán en un sistema coordenado que puede ser el mismo o diferente al de la cámara. El centro de la cámara (o el orificio de una cámara oscura) está representado por el origen del sistema coordenado de la cámara O_C; es por este punto donde todos los rayos de luz pasarán para generar la imagen. En esta representación, el plano de proyección **I** se encuentra entre el centro de la cámara y el punto, lo cual es equivalente a hacer una proyección posterior invertida. Es importante resaltar que el centro de la imagen (c_x, c_y) se encuentra sobre el eje z_c (que corresponde al centro óptico) y consideraremos que el plano **I** es coplanar al plano $x_c y_c$ (lo cual es válido para las cámaras más recientes).
El modelo matemático de una cámara *pinhole* es relativamente simple, toma en consideración los siguientes elementos: la distancia focal (f), tamaño real del píxel rectangular ($\mu_x \times \mu_y$) y el punto principal (c_x, c_y). Estos parámetros son integrados en una matriz de la forma:

$$\boldsymbol{A} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1.4}$$

donde $f_x = f\mu_x$ y $f_y = f\mu_y$. Sin embargo estamos solamente considerando que el punto se encuentra dentro del sistema de la cámara, no del espacio, por lo que este modelo funciona "localmente".
Dado que la cámara también se encuentra en el espacio, el modelo puede extenderse realizando una transformación lineal entre ambos sistemas. Para esto se utiliza una matriz de rotación $\boldsymbol{R}$ y un vector de translación $\boldsymbol{t}$, con lo que podemos hacer que el sistema de la cámara concuerde con el sistema espacial. Para este proceso se utiliza la siguiente matriz de transformación rígida:

$$\boldsymbol{D}_{3x4} = \begin{bmatrix} \boldsymbol{R}_{3\times 3} & \boldsymbol{t}_{3\times 1} \end{bmatrix} \tag{1.5}$$

Así, formamos lo que se conoce como la matriz de proyección

$$\boldsymbol{P} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} = \boldsymbol{AD} \tag{1.6}$$

con la que podemos conocer las coordenadas en la imagen (m) de un punto en el espacio M mediante el producto matricial siguiente:

$$\boldsymbol{m} = \boldsymbol{PM} \tag{1.7}$$

Conocer los valores de los parámetros antes mencionados se le conoce como **calibración** y podemos dividirla en dos: la calibración interna encuentra los valores de A, que representan a la cámara; y la calibración externa encuentra los de D, que ubican a la cámara en el espacio. La calibración es necesaria para poder utilizar este modelo computacionalmente, por lo que se ha desarrollado diferentes técnicas para realizarla; además, de forma general es relativamente sencilla de hacer. Más adelante en este trabajo se describe la técnica utilizada y el proceso de solución de la misma.

1.5. Visión estereoscópica

Los seres humanos vemos el mundo en tres dimensiones; es decir que podemos ver la profundidad de una escena y estimar la posición de los objetos en ella. Gracias a esto estimamos en qué lugar del espacio colocar nuestra mano para atrapar una pelota, qué tanto debe uno mover la mano para tomar un vaso de agua, o con qué fuerza lanzar un balón para que entre en una canasta. Dicha ventaja evolutiva ha sido posible gracias a que disponemos de mayor información que lo que puede dar una sola imagen; es decir, gracias a que tenemos dos ojos. La disposición de nuestro sistema visual permite tener dos vistas para una misma escena. Este hecho, aunado a la habilidad de nuestro cerebro de aprender, ha calibrado perfectamente nuestro cerebro para que a través de lo que percibimos por los ojos podamos estimar la posición espacial de un objeto en relación con nuestro cuerpo, inclusive estimar sus dimensiones. Dicha estimación no es obvia cuando se tiene una sola vista y ningún elemento de referencia.

De una forma computacional, obtener información estereoscópica de una escena puede ser posible utilizando dos cámaras. Casi desde la creación de las primeras cámaras, se comenzó a utilizar esto para tener imágenes desde dos vistas, como la mostrada en la figura 1.8. Trabajar con más de una vista tiene grandes ventajas, así como desventajas. Por ejemplo, podemos reconstruir la trayectoria tridimensional de un objeto en la escena, pero tenemos problemas cuando solo una de estas vistas "ve" al objeto (es decir que en una de las vistas está ocluido). Este es uno de los problemas principales al trabajar con imágenes estereoscópicas [14, p. 140]. De entre estos, los más importantes son: 1) encontrar la correspondencia de la proyección de un objeto en ambas imágenes, y 2) cómo reconstruir la escena que se está visualizando.
El problema de correspondencia plantea, en pocas palabras, identificar las partes de la imagen de una vista que corresponden con la imagen de la segunda vista. Existen diferentes propuestas para resolver este problema, aun así nos encontramos con algunas dificultades [15], principalmente (ver figura 1.9):

1. Cuando un objeto es ocluido en una imagen y en otra no, la información de dicho objeto está incompleta.

2. Las imágenes, al no ser obtenidas por la misma cámara, pueden tener diferencias en intensidades de píxeles sobre el mismo objeto.

Actualmente existen diversos algoritmos que intentan resolver este problema, entre ellos encontramos: correlación por ventanas (*window-based correlation)*, métodos cooperativos (*cooperative methods*), programación dinámica (*Dynamic programming*) y métodos basados en gráficas (*graph-Based methods*).
Una de las maneras de extraer información de las capturas estereoscópicas es mediante geometría epipolar. Esta geometría está diseñada para trabajar con datos de proyecciones en múltiples vistas y sus fundamentos son relativamente sencillos (ver figura 1.10): un punto en el espacio (W) es proyectado en un punto bidimensional en una de las vistas (W es proyectado en I_1 para obtener w_1), la posición de la proyección del mismo punto en otra vista (I_2) forma parte de

Fig. 1.8. Imágenes de la misma escena capturadas por dos cámaras separadas entre sí. Las diferencias son sutiles, aunque suficientes para estimar la profundidad a la que se encuentran los objetos: obtenido de [13].

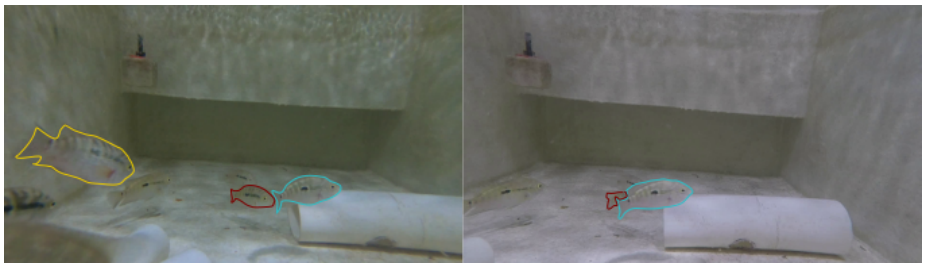


Fig. 1.9. Captura estereoscópica de una escena subacuática. En azul podemos ver un pez que ocluye al pez rojo solamente en una de las vistas. En amarillo vemos un pez que solamente aparece en una vista. Se hace evidente la diferencia en el espectro de colores de ambas cámaras, debido a los tipos de sensores de las cámaras.

un conjunto de puntos formado por todos los posibles puntos W (representados por ¿W?) que se proyecten en w_1. Este conjunto de puntos forma una línea l en la segunda vista, llamada línea epipolar. Esto facilita la forma en la que se busca información en más de una vista, ya que no es necesario buscar en toda la imagen, sino solamente sobre la línea epipolar correspondiente.

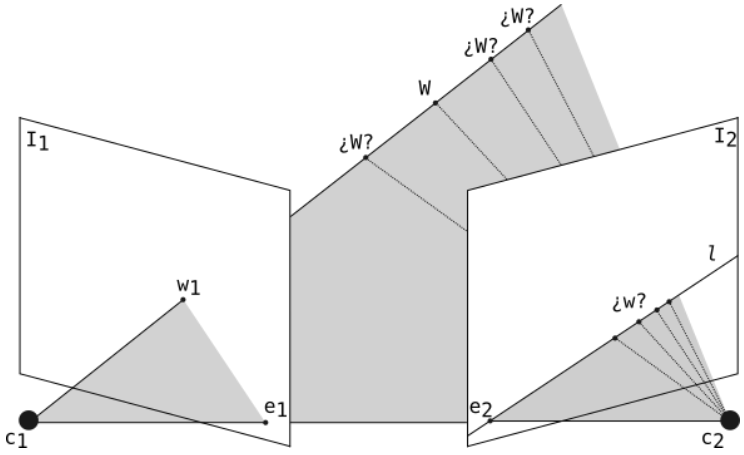


Fig. 1.10. Esquematización de los fundamentos de la geometría epipolar. El punto W es proyectado en un punto conocido en la imagen 1 (I_1), pero el mismo punto se puede encontrar proyectado sobre una línea, conocida como línea epipolar (l), en la imagen 2 (I_2). Por lo tanto, la correspondencia entre ambas proyecciones se limita a la búsqueda del punto W sobre la línea epipolar. e_1 y e_2 son los puntos epipolares de cada vista.

Para el caso de tener tres vistas o más se tienen casos particulares y generales de esta geometría, aunque el fundamento es el mismo. Además, la precisión de las estimaciones depende directamente de la calibración de las cámaras, por lo

que es sumamente importante realizarla adecuadamente, ya que se utiliza tanto la matriz fundamental (F) como la matriz esencial (E).
Además de este método de estimación (paramétrica), existen otros métodos (no paramétricos) basados en aprendizaje profundo para extraer información tridimensional de la escena, aunque tienen la gran desventaja de depender de datos de entrenamiento, lo contrario al método antes mencionado. En este trabajo se utilizó geometría epipolar para extraer información relevante de las escenas, por lo que se explicarán los fundamentos en capítulos posteriores.

Capítulo 2

Antecedentes

2.1. Revisión de trabajos relacionados al monitoreo de peces

Obtener información del mundo real a través de imágenes es un problema que se ha intentado solucionar desde el siglo pasado y que sigue siendo activamente investigado. Problemas de la vida real pueden ser resueltos utilizando métodos y técnicas de visión computacional como: reconstrucción 3D, segmentación y detección de objetos, clasificación, seguimiento de movimiento, extracción de características, entre otros. Uno de los problemas para los cuales, por su naturaleza, la visión computacional tiene grandes ventajas es el monitoreo de forma continua y no invasiva. Técnicas de visión por ordenador han demostrado ser objetivas y efectivas para estas tareas en el monitoreo subacuático, tanto en cautiverio como en ambientes naturales [16]. Ruff *et al* [17] describen algunas de estas técnicas utilizadas desde hace un par de décadas:

- **Ondas ultrasónicas (sondas de ultrasonido)**. Para esta técnica se utilizan sensores ultrasónicos, los cuales no dependen de la luz, por lo que pueden ser utilizados en cualquier momento del día. Sin embargo, su resolución es muy baja (en comparación con una imagen u otras técnicas como *Light Detection and Ranging*) y contiene, por su naturaleza, gran cantidad de ruido.
- **Implantes de rastreo**. Se basa en colocar implantes electrónicos a los peces con capacidad de monitorear desde su estado físico hasta el de su entorno de manera remota. Esta técnica es costosa y en algunos escenarios poco viables.
- **Captura de video**. Con cámaras capaces de funcionar bajo el agua, se pueden obtener datos de imágenes y video de manera no invasiva. Estos datos son utilizados para el estudio del comportamiento de peces [18], obtención de características morfológicas [19], índices de crecimiento [20], tamaños y edad [21, 22, 23] y conteo [22].

- **Monitoreo manual**. Esta técnica invasiva requiere manipular al pez utilizando algún medio de captura, lo cual puede lastimarlos y estresarlos [18, 21].

Es de gran interés en la industria acuícola poder estimar el tamaño de peces, además de automatizar este proceso [24]. Algunos métodos simples y económicos utilizan una cámara solamente. Hao *et al* [25] desarrolló un análisis del tipo de técnica más conveniente dependiendo en dependencia con la especie y el ambiente de monitoreo. Hablando precisamente de sistemas de una sola cámara, en [26], se utilizaron trampas en criaderos por donde se hacían pasar los peces y así poder medir sus dimensiones. De forma similar en [27], se utilizó una objeto de dimensiones conocidas, captado en la misma imagen, para estimar la longitud de atunes fuera del agua. Existen desventajas evidentes: se requiere que todos los peces sean similares en forma y tamaño para disminuir el error; se asume que la distancia del pez a la cámara no cambia y es conocida; y la posición del pez puede afectar directamente las mediciones.
Usar una sola cámara conlleva trabajar con ciertas limitaciones, a costa de la dificultad de implementación; recordemos que una imagen es una proyección de una escena tridimensional a un espacio de 2 dimensiones. En cambio, trabajar con dos cámaras, creando un sistema de visión esteresocópica, permite trabajar en un plano tridimensional a costa de una implementación más costosa y complicada. A pesar de esto, y gracias a la reducción de costos en sistemas de captura y el desarrollo de algoritmos eficientes, cada vez es más común preferir un sistema estéreo a uno mono.
Trabajar con imágenes subacuáticas implica enfrentarse a otros retos adicionales [28, 29]:

- Variables físicas del agua como refracción y absorción de la luz, que generan fenómenos como iluminación no uniforme y atenuación de luz.
- Fenómenos ópticos no lineales al trabajar con la carcasa donde se encuentra contenida la cámara (*housing*).
- Partículas sólidas suspendidas o burbujas que generen ruido en las imágenes capturadas. Además de interferencia por otros organismos.

Existen diferentes técnicas y métodos de visión computacional aplicados a la acuicultura desarrollados en los últimos 20 años. Shortis *et al* en [30] hace una revisión de algunos de ellos y propone un flujo de trabajo general que podría funcionar como una plantilla. Podríamos sintetizar todos los pasos propuestos en los siguientes [31]:

1. Extracción del fondo.
2. Identificación de los candidatos a peces. En [32] se detalla un modelo de identificación y clasificación de peces, utilizando la base de datos Image CLEF 2014. De manera similar, en [33] se desarrolló un modelo para la detección y segmentación de peces en ambientes no controlados.

3. Rastreo de los candidatos, para descartar ruido y objetos no deseados.

4. Realizar las mediciones (forma, volumen).

Para el trabajo desarrollado en esta tesis, se propone que el proceso sea completamente automático, es decir que ninguna persona intervenga en ningún momento. En la actualidad todavía hay sistemas que no han automatizado por completo el proceso, sobre todo en la detección de la boca y la aleta caudal (necesarias para hacer la medición de la longitud) que se realizan de forma manual [34, 35], o de forma semiautomática [36]. De entre los trabajos encontrados se destacan dos: el de Rodriguez *et al* [37], y el de Shi *et al* [38]. Ambos presentan resultados cuantitativos de una detección individual de peces, mientras que el resto de autores reportados basan sus análisis en técnicas estadísticas para mediciones de conjuntos de peces, sin conocer sus dimensiones individuales. Para Rodriguez *et al* el error mínimo relativo alcanzado fue del 10 %, mientras que para Shi *et al* fue del 1.22 %. Ambos son errores cuadráticos promedio de la longitud estimada de cada pez, respecto a la real.

2.2. Planteamiento del problema

El conocimiento de las dimensiones de peces en cautiverio permiten generar datos estadísticos que son de interés económico en la planificación de la alimentación en el sector acuícola, o en el seguimiento de bancos de peces en ambientes naturales. Actualmente, a pesar de la facilidad de acceder a recursos tecnológicos para el monitoreo mediante cámaras, la dificultad de procesar todos los datos obtenidos es un obstáculo que sigue limitando su uso. Para el monitoreo subacuático es aún más complicado el procesamiento de estos datos, así como el desarrollo y mantenimiento de un sistema automático, lo que lleva a los interesados a continuar con métodos manuales. He aquí la importancia del desarrollo de métodos computacionales que automaticen este proceso. En la UNAM, Campus Sisal en Yucatán, se llevan a cabo trabajos de investigación donde la automatización de tareas, como el monitoreo, hará más eficiente la obtención de resultados.

2.3. Hipótesis

En el monitoreo subacuático, mediante técnicas de visión estereoscópica computacional y aprendizaje automático, será posible determinar la posición espacial y la longitud de peces de forma automática.

2.4. Objetivos

2.4.1. Objetivo general

Diseñar un sistema prototipo de visión computacional para obtener información de un arreglo estereoscópico que permita identificar y obtener las características espaciales de peces de manera automatizada.

2.4.2. Objetivos particulares

1. Llevar a cabo una revisión bibliográfica referente al uso de visión estereoscópica computacional para el monitoreo de especies acuáticas.
2. Diseñar un arreglo de cámaras para visión estereoscópica.
3. Realizar la calibración de las cámaras del arreglo estereoscópico.
4. Diseñar un ambiente y/o escena controlado(a) (luz, ruido, dimensiones) para la captura de los videos.
5. Implementar un sistema de aprendizaje automático (usando redes neuronales convolucionales) para la detección de peces a partir de adquisición estereoscópica de las imágenes.
6. Implementar algoritmos de correspondencia en regiones de interés a partir de la extracción de características particulares del objeto de interés.
7. Diseño e implementación de un sistema de referencia espacial basado en visión estereoscópica y geometría epipolar.
8. Realizar pruebas con el prototipo en un ambiente controlado (luz, profundidad de captura, dimensiones de los objetos).
9. Realizar pruebas con el prototipo en un ambiente semi-controlado con peces vivos (dimensiones, número de peces).
10. Analizar y validar los resultados generados por el sistema prototipo.

2.5. Aportación

Este proyecto toma diferentes técnicas de visión computacional, aprendizaje automático y estereoscopía para generar un prototipo que sea fácil de utilizar, eficaz y confiable en la detección y medición de peces en ambientes subacuáticos. Si bien ya existen diferentes trabajos que realizan tareas similares, en este se proponen técnicas de aprendizaje profundo que no han sido utilizadas con este fin, mejorando su desempeño y la rapidez de procesamiento de los datos. Como se mostrará a lo largo de este escrito, los resultados son satisfactorios para su uso en ambientes controlados, lo que significa un gran paso hacia el desarrollo de un sistema robusto para su uso en ambientes semi o no controlados.

Capítulo 3

Marco teórico

3.1. Red neuronal convolucional

El algoritmo más utilizado para realizar tareas de aprendizaje automático en imágenes es el de una red neuronal convolucional (CNN por sus siglas en inglés), gracias a su capacidad de procesar información de una forma similar a la del sistema nervioso visual animal [39] y a los excelentes resultados que ha demostrado tener [40]. Las redes neuronales convolucionales se componen de elementos esenciales, [40, 41] las clasifican en: capa de convolución, capa de *pooling*, capa totalmente conectada, y capa de pérdida o de clasificación. Con estos elementos se puede diseñar una CNN para la realización de tareas específicas.

3.1.1. Arquitectura de una CNN

La secuencia de elementos que son utilizados en la CNN, y cómo estos se conectan entre ellos, componen la arquitectura de la red. Esta arquitectura define en gran medida cómo esta se desempeñará el algoritmo. Su diseño depende de la tarea a realizar, así como de los datos a procesar, por lo que no hay una red general para resolver cualquier problema de visión computacional, sino familias de arquitecturas que pueden tener desempeños diferentes ante problemas similares. A continuación se explicará el funcionamiento de algunos de estos elementos.

3.1.1.1. Capa de convolución

Es la capa encargada de realizar la convolución de la imagen de entrada con un conjunto de filtros, los cuales son aprendidos por la red en el proceso de aprendizaje. Tal como en las operaciones de convolución en 2 dimensiones, se utilizan filtros con la diferencia de que estos son de tres dimensiones. La tercera dimensión nos dice cuántos filtros bidimensionales son utilizados, y posteriormente entrenados. Como en operaciones de convolución en 2D, el resultado puede mantener, disminuir o extender su resolución espacial, aunque lo más común es mantenerla a lo largo de las capas. Generalmente, para la disminución

en la resolución espacial se utilizan las capas de *pooling*, que se explicarán más adelante.

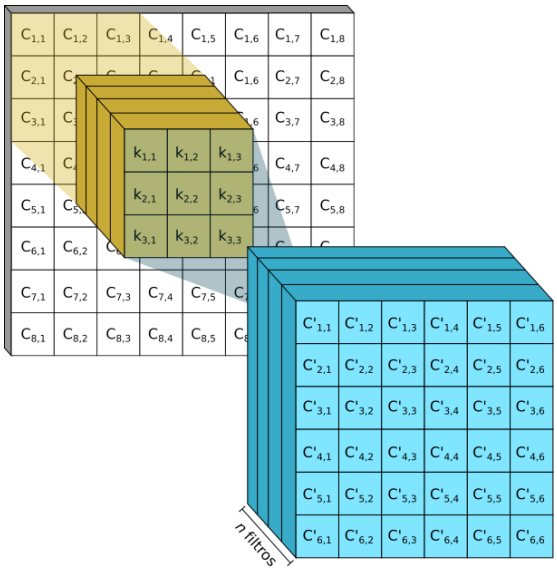


Fig. 3.1. Operación de la capa de convolución. Se va desplazando el filtro tridimensional, comunmente de $3 \times 3 \times n$, donde n es el número de filtros bidimensionales que lo componen y que determina el número de canales de la siguiente capa.

Uno de los hiperparámetros que se deben considerar son las dimensiones de los filtros ($k \times k$) y la cantidad n_k. Es común que se utilicen filtros de 3×3, aunque la única recomendación (o requisito) es que las dimensiones sean impares, para que el centro del filtro coincida con un solo píxel. La tercera dimensión dependerá del número de canales que se le alimentan. Cuando se considera más de un solo filtro, el resultado de cada uno se apila con el resto, generando un arreglo de $M \times N \times n_k$, donde $M \times N$ es la resolución de la capa anterior. Al final de esta convolución, el resultado numérico se hace pasar por una función de activación. Por lo general se utilizan funciones no lineales como la función logística, tanh (tangente hiperbólica), ReLU (*Rectified Linear Unit*), SiLU (*Sigmoid Linear Unit*), etc.

3.1.1.2. Capa de agrupación (*pooling*)

Estas capas se han diseñado de tal forma que se pueda disminuir la resolución espacial de la información que pasa por la red minimizando la pérdida de información relevante. Existen muchas maneras de hacer estas agrupaciones, por ejemplo, tomar 4 píxeles vecinos y considerar el valor más grande, o el más pequeño, o el promedio, o la mediana, etc. Es decir, se hace pasar esta información

a una función que determina un único valor que representaría al conjunto que se está considerando.

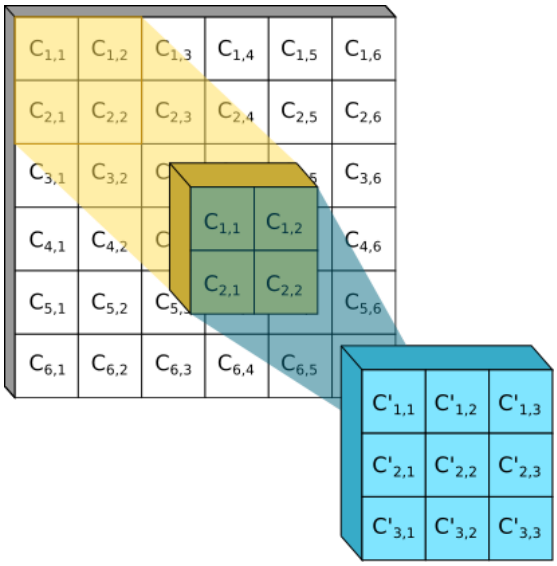


Fig. 3.2. Funcionamiento de la capa de agrupación. Se toman pequeños grupos de píxeles y se procesan para generar los píxeles de la nueva capa. Este procesamiento puede implicar extraer el valor máximo, el mínimo, el promedio, etc.

Se ha acostumbrado a hacer agrupaciones de 4×4 píxeles considerando como resultado el valor más grande del grupo, a este proceso se le llama *max pooling*. Independientemente de cómo se haga el *pooling*, hay dos beneficios que conlleva su uso: la disminución del número de operaciones al trabajar con la CNN y la prevención de un sobre ajuste de la misma. El uso de estas capas generalmente se hace después de una secuencia de convoluciones, disminuyendo la resolución del resultado a la mitad.

3.1.1.3. Capa completamente conectada

Se le llama así a la capa compuesta por una red multicapa de perceptrones, comúnmente utilizada al final de la CNN. Esta capa recibe la información que ya ha sido procesada y codificada por las capas de convolución y agrupación, por lo que su tarea es tomar una decisión respecto a los datos recibidos. Dependiendo del tipo de función de activación seleccionada, pueden ser utilizadas para tareas de clasificación o regresión. Es en estas capas donde, según [41] se lleva a cabo el "razonamiento de alto nivel", haciendo referencia a que, con base en la información de relevancia obtenida por las capas anteriores, realiza la toma de decisiones sobre la información procesada.

3.1.1.4. Capa de pérdida

Se le conoce así a la última capa de una CNN, la cual se encarga de evaluar la respuesta de la red ante cierta entrada, calculando un error (también conocido como pérdida, o *loss*). De esta capa dependen directamente los resultados del entrenamiento, pues la evaluación de estos errores será utilizada para el ajuste de los pesos. Existe una gran cantidad de funciones de activación para esta capa, donde las más comunes son las *softmax* (función exponencial normalizada, usada para vectores n-dimensionales) y entropía cruzada, para tareas de clasificación.

3.1.2. Entrenamiento de una CNN

Como se mencionó anteriormente, entrenar una red neuronal significa modificar los valores de sus pesos internos, de tal manera que la salida de la red genere los resultados esperados. Esta tarea es sumamente importante, pues de ella depende el desempeño de las inferencias que hará el sistema. La manera más común de entrenar una CNN es mediante aprendizaje supervisado (ver figura 3.3). Para poder llevarlo a cabo se necesitan datos etiquetados, que se refiere a que cada dato (en este caso de imagen) tenga un identificador. Es mediante este identificador que se "supervisa" el entrenamiento, de manera que se ajustan los pesos para que se minimice el error de inferencia. Por ejemplo, para un proceso de detección de perros y gatos en imágenes, se requiere que cada imagen tenga el identificador "perro", o "gato", el cual es pasado a la red para hacer la evaluación de cómo hace la tarea de clasificación. Este tipo de entrenamiento permite que la red infiera más rápidamente cuáles son las características más importantes de los datos para cumplir con su objetivo.
Sin duda alguna, uno de los componentes más importantes en desempeño de un algoritmo de aprendizaje automático son los datos con los que se entrena. La calidad y cantidad de estos debe estar garantizada para obtener los mejores resultados posibles. En el caso de imágenes, siempre es clave tener imágenes con la menor cantidad de ruido posible, de dimensiones similares (para no perder precisión al momento de ajustar los tamaños en caso de usarse para clasificación a nivel de píxel o localización) y en la mayor cantidad disponible.
El entrenamiento no en más que un problema de optimización, donde los parámetros son los pesos y la función a minimizar está dada por la capa de pérdida. Así, como cualquier otro problema de optimización, se puede utilizar una gran cantidad de métodos, desde descenso de gradiente hasta metaheurísticos evolutivos. El primero ha sido uno de los más utilizados al formar parte del algoritmo *Backpropagation*, que ha sido utilizado desde el siglo pasado como optimizador en el entrenamiento de redes neuronales.

3.2. Visión estereoscópica

Como se mencionó anteriormente, podemos tener sistemas de una sola vista (usando una sola cámara) y que son los que solemos utilizar. En estos casos no nos interesa más que plasmar en un plano (imagen) una proyección de lo

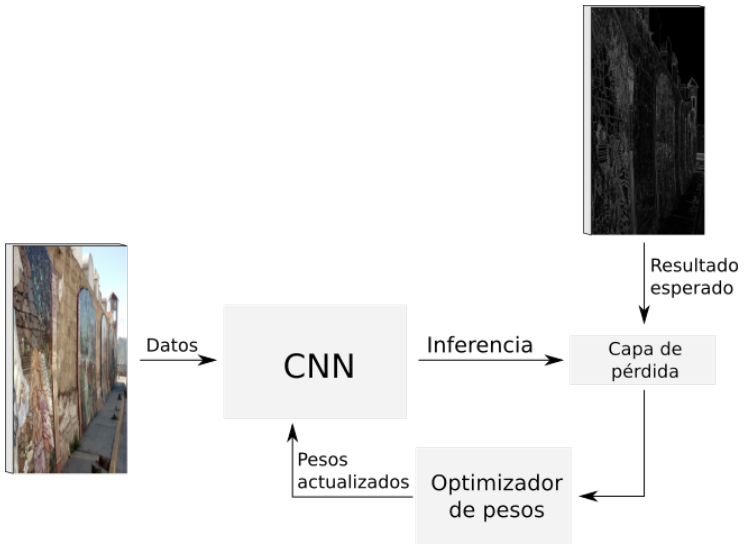


Fig. 3.3. Esquematización del proceso de entrenamiento supervisado de una CNN. Se procesan los datos por la CNN, la inferencia se evalúa en la capa de pérdida, esta evaluación es utilizada por el optimizador de pesos para generar una actualización que será aplicada a la CNN. Este proceso es iterativo hasta llegar a un valor mínimo de la pérdida.

que vemos en el mundo real, con sus colores, sus formas y sus características. Este proceso fue ya descrito en la introducción, donde se explicó el modelo de cámaras y la generación de imágenes. Si quisiéramos plantear el problema inverso (recrear una escena tridimensional a partir de los datos de una imagen), nos encontraríamos con la imposibilidad de realizar esta acción con un proceso tan sencillo como el de plasmar una imagen.
Recordemos que la proyección de puntos tridimensionales en un plano es fácil de hacer, conociendo la posición de los puntos en el espacio y el modelo matemático de la cámara, se puede saber la posición que le correspondería a cada uno de ellos en la imagen. Sin embargo, hacer el proceso inverso no es así de sencillo, ya que es posible que más de uno de los puntos en el espacio se proyecte en un solo punto de la imagen. Esto hace que intentar recuperar la información de profundidad a partir de una sola imagen sea prácticamente imposible, y ha permitido generar grandes efectos visuales en el cine del siglo pasado. Entonces, ¿cómo recuperar esta información? Ya sea que contemos con elementos de referencia, de los que conozcamos tanto la posición como las dimensiones, podríamos hacer mediciones sobre otros objetos que se encuentren en el mismo plano en el espacio. Contar con objetos conocidos en todas las escenas no siempre es posible o conveniente, por lo que es importante utilizar otros métodos, como la estereoscopia.
Utilizar más de una vista en una escena tiene grandes beneficios, principalmente porque dos puntos en el espacio que se proyecten sobre el mismo punto en

una imagen no necesariamente lo harán en otra, tomada al mismo tiempo. Esta información adicional puede ser utilizada para poder calcular la posición tridimensional de un punto según su posición en las imágenes, tal y como se explicará a continuación.

3.2.1. Modelo de cámara y calibración individual

Anteriormente se explicó el funcionamiento del modelo de una cámara *pinhole*, la cual puede ser descrita por la siguiente ecuación:

$$\boldsymbol{P} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} = \boldsymbol{AD} \tag{3.1}$$

donde A es la matriz de calibración interna y D la de calibración externa. Como se explicó en la introducción, cada punto ($\boldsymbol{M} = \begin{bmatrix} U & V & W \end{bmatrix}^T$) que se encuentra en el espacio debe ser transformado para trabajar en el sistema coordenado de la cámara mediante una translación ($\boldsymbol{t}$) y una rotación ($\boldsymbol{R}$).

$$\tilde{\boldsymbol{M}}' = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} \tilde{\boldsymbol{M}} \tag{3.2}$$

donde $\tilde{\boldsymbol{M}}$ y $\tilde{\boldsymbol{M}}'$ son las coordenadas homogéneas del punto $\boldsymbol{M}$ en el sistema coordenado del espacio y de la cámara, respectivamente. Una vez teniendo expresado en coordenadas de la cámara, se realiza la proyección hacia el plano de la imagen de la siguiente forma

$$s\tilde{\boldsymbol{m}} = \boldsymbol{A}\tilde{\boldsymbol{M}}' \tag{3.3}$$

donde s es un factor de escalamiento y $\tilde{\boldsymbol{m}}$ es el punto en la imagen $\boldsymbol{m} = \begin{bmatrix} u & v \end{bmatrix}^T$ expresado en coordenadas homogéneas. Simplificando esta expresión y considerando la matriz de transformación proyectiva $\boldsymbol{P}$, podemos conocer la posición de un punto en la imagen ($\tilde{\boldsymbol{m}}$) correspondiente a uno en el espacio ($\tilde{\boldsymbol{M}}$), mediante el método de Transformación Linear Directa, descrito en [42, p. 88, 178-179], partiendo de la ecuación

$$s\tilde{\boldsymbol{m}} = \boldsymbol{AD}\tilde{\boldsymbol{M}} = \boldsymbol{P}\tilde{\boldsymbol{M}} \tag{3.4}$$

todo expresado en coordenadas homogéneas. Para eliminar esta dependencia de la escala, ya que $\tilde{\boldsymbol{m}}$ y $\boldsymbol{P}\tilde{\boldsymbol{M}}$ son vectores que tienen la misma dirección, pero diferente magnitud, se expresa la ecuación 3.4 como el producto vectorial

$$\tilde{\boldsymbol{m}} \times \boldsymbol{P}\tilde{\boldsymbol{M}} = 0 \tag{3.5}$$

Expresando este producto vectorial como una operación matricial, obtenemos

$$\begin{bmatrix} \boldsymbol{0}^T & -w\tilde{\boldsymbol{M}}^T & v\tilde{\boldsymbol{M}}^T \\ w\tilde{\boldsymbol{M}}^T & \boldsymbol{0}^T & -u\tilde{\boldsymbol{M}}^T \\ -v\tilde{\boldsymbol{M}}^T & u\tilde{\boldsymbol{M}}^T & \boldsymbol{0}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{P}^1 \\ \boldsymbol{P}^2 \\ \boldsymbol{P}^3 \end{bmatrix} = \boldsymbol{0} \tag{3.6}$$

donde $\tilde{\boldsymbol{m}} = \begin{bmatrix} u & v & w \end{bmatrix}^T$ y $\boldsymbol{P}^{iT} = \begin{bmatrix} P^{i,1} & P^{i,2} & P^{i,3} \end{bmatrix}^T$ es el vector correspondiente a la i-ésima fila de la matriz $\boldsymbol{P}$. Ya que la matriz homogénea $\boldsymbol{P}$ tiene 11 grados de libertad, es necesario conocer al menos $5\frac{1}{2}$ correspondencias $\tilde{\boldsymbol{M}} \rightarrow \tilde{\boldsymbol{m}}$ para que el sistema de ecuaciones esté completamente definido. Esta última media correspondencia corresponde a una sola de las coordenadas (x, y) del último punto considerado.
Para encontrar los valores correspondientes a la matriz $\boldsymbol{P}$ se sigue el proceso de calibración propuesto por Zhang [43], el cual es uno de los algoritmos más utilizados para la calibración individual de cámaras; y para el caso de la calibración subacuática se siguió lo propuesto por [44], utilizando un tablero de ajedrez como patrón de dimensiones conocidas (ver figura 3.4). Para encontrar los valores de $\boldsymbol{P}$ se toma un conjunto de puntos $\boldsymbol{m}_i = \begin{bmatrix} u_i & v_i \end{bmatrix}^T$ conocidos del patrón de calibración. Encontrar este patrón en la imagen es fácil y rápido, además se conocen las dimensiones y posiciones relativas de los cuadros, por lo que es muy fácil encontrar los puntos correspondientes a las esquinas de los cuadros. En técnicas de calibración de cámaras, este tipo de patrones son muy utilizados, aunque existe una gran variedad patrones diferentes.

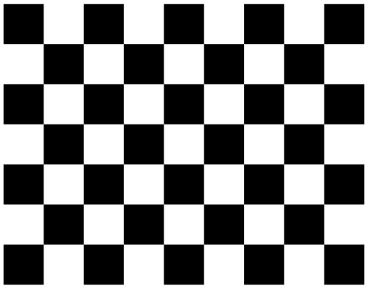


Fig. 3.4. Patrón de calibración estilo tablero de ajedrez, con 48 puntos de posición conocida y fácilmente detectables para llevar a cabo la calibración.

Mediante un rápido procesamiento de la imagen se pueden encontrar todos los puntos de interés antes mencionados. Estableciendo una de las esquinas del patrón como punto de referencia espacial (es decir, el origen del sistema coordenado del espacio) se procede a calcular las coordenadas de los puntos encontrados en el espacio. Entonces ya tenemos puntos correspondientes $\tilde{\boldsymbol{M}}_i \rightarrow \tilde{\boldsymbol{m}}_i$ con los que podemos armar nuestro sistema de ecuaciones basándonos en la expresión 3.7. Es fácil notar que hay dependencia lineal entre estas ecuaciones, por lo que podemos reescribirla de la siguiente forma

$$\begin{bmatrix} \mathbf{0}^T & -w_i\tilde{\boldsymbol{M}}_i^T & v\tilde{\boldsymbol{M}}_i^T \\ w_i\tilde{\boldsymbol{M}}_i^T & \mathbf{0}^T & -u_i\tilde{\boldsymbol{M}}_i^T \end{bmatrix} \begin{bmatrix} \boldsymbol{P}^1 \\ \boldsymbol{P}^2 \\ \boldsymbol{P}^3 \end{bmatrix} = \boldsymbol{B}\boldsymbol{P'} = \mathbf{0} \tag{3.7}$$

Así, resolviendo el conjunto de ecuaciones lineales $\boldsymbol{B}\boldsymbol{P'} = 0$, se puede encontrar los valores que conforman la matriz de transformación proyectiva $\boldsymbol{P}$.

En este primer paso podemos encontrar una solución exacta al problema, pero que depende de la calidad de la correspondencia de los puntos seleccionados. Es decir, que si esta se hace mal, o existe ruido en las imágenes, la solución no será la esperada. Sin embargo, podemos utilizar esta solución como una primera aproximación a la solución óptima. Considerando que se cuenta con más de 6 puntos de correspondencia se puede plantear una función de error a minimizar, en este caso se plantea un error basado en la distancia algebraica que considera al j-ésimo punto de la imagen i y su respectiva proyección, expresado por la magnitud de la diferencia entre dos vectores, tal que

$$\sum_{i=1}^{m}\sum_{j=i}^{n}||\boldsymbol{m}_{ij}-\hat{\boldsymbol{m}}(\boldsymbol{A},\boldsymbol{R}_i,\boldsymbol{t}_i,\boldsymbol{M}_j)||^2 \tag{3.8}$$

donde $\hat{\boldsymbol{m}}(\boldsymbol{A},\boldsymbol{R}_i,\boldsymbol{t}_i,\boldsymbol{M}_j)$ equivale al punto $\boldsymbol{M}_j$ proyectado en la imagen i. Entonces este problema se vuelve uno de optimización, donde las matrices de calibración son los parámetros que se varían para minimizar la función 3.8. Para ello existen diferentes tipos de optimización, aunque uno de los más utilizados es el de Levenberg–Marquardt, que funciona muy bien para resolver problemas de mínimos cuadrados no lineales.

3.2.2. Calibración estereoscópica

Una vez hecha la calibración de cada cámara, se procede a calibrar el sistema estereoscópico. Para esto se deben encontrar los siguientes parámetros:

- **La matriz de rotación $\boldsymbol{R}$.** Rota el sistema coordenado de la segunda cámara (C_2) para que corresponda con la primera (C_1) para que los planos de imagen en ambas cámaras sea coplanar.
- **El vector de traslación ($\boldsymbol{T}$).** Traslada el sistema coordenado de la segunda cámara (C_2) al de la primera (C_1).
- **Matriz Esencial ($\boldsymbol{E}$).** Describe la relación entre puntos en los sistemas coordenados de ambas cámaras.
- **Matriz Fundamental ($\boldsymbol{F}$).** Describe la relación entre los puntos en los planos de la imagen de ambas cámaras.

Los parámetros de rotación ($\boldsymbol{R}$) y traslación ($\boldsymbol{T}$) se pueden conocer resolviendo la siguiente expresión

$$\boldsymbol{R}=\boldsymbol{R}_i(\boldsymbol{R}_j)^T \tag{3.9}$$

$$\boldsymbol{T}=\boldsymbol{T}_i-\boldsymbol{R}\boldsymbol{T}_j \tag{3.10}$$

donde $\boldsymbol{R}_i$ y $\boldsymbol{T}_i$ representan la matriz de rotación y el vector de traslación de la cámara i a la cámara j y viceversa.

Por otro lado, la matriz esencial ($\boldsymbol{E}$) puede calcularse a partir de la matriz fundamental ($\boldsymbol{F}$). Recordemos que la matriz fundamental puede ser considerada como una homografía entre los planos de las dos imágenes, donde se cumple:

$$\tilde{\boldsymbol{m}}'^T \boldsymbol{F} \tilde{\boldsymbol{m}} = \boldsymbol{0} \tag{3.11}$$

donde $\tilde{\boldsymbol{m}}'$ y $\tilde{\boldsymbol{m}}$ con puntos que corresponden a un punto en el espacio proyectado sobre la segunda y primera imagen, respectivamente. El cálculo de esta matriz es primero analítico, de la siguiente forma:

$$\boldsymbol{F} = \boldsymbol{A}_2^{-T} \cdot \boldsymbol{E} \cdot \boldsymbol{A}_1^{-1} \tag{3.12}$$

donde la matriz esencial está definida por

$$\boldsymbol{E} = \begin{bmatrix} 0 & -T_3 & T_1 \\ T_3 & 0 & -T_2 \\ -T_1 & T_2 & 0 \end{bmatrix} \boldsymbol{R} \tag{3.13}$$

donde T_i es el i-ésimo elemento del vector de traslación. El resultado se afina minimizando el error de reproyección. Con este método podemos también conocer el error promedio del proceso para evaluar la calidad de la calibración.

3.2.3. Reconstrucción tridimensional

3.2.3.1. Rectificación estereoscópica

Cada punto en la imagen izquierda (I_1) genera toda una línea epipolar en la imagen derecha (I_2), como se muestra en la figura 3.5, descrita por las relaciones

$$\boldsymbol{l}' = \boldsymbol{F}\boldsymbol{m} \tag{3.14}$$

$$\boldsymbol{l} = \boldsymbol{F}^T \boldsymbol{m}' \tag{3.15}$$

donde $\boldsymbol{F}$ es la matriz fundamental, $\boldsymbol{m}$ y $\boldsymbol{l}$ son el punto y la línea epipolar de la imagen izquierda, y $\boldsymbol{m}'$ y $\boldsymbol{l}'$ son el punto y la línea epipolar de la imagen derecha. Lo que hacemos al rectificar es alinear las líneas epipolares de ambas vistas, de tal manera que todas sean paralelas a la línea base (la línea que una ambos centros de cámara).

3.2.3.2. Corrección de puntos correspondientes

Poniendo como ejemplo la selección de dos puntos característicos (por ejemplo, la boca y aleta caudal, como será considerado más adelante en este trabajo) en dos imágenes diferentes, es muy probable que no se cumpla que

$$\boldsymbol{m}'^T \boldsymbol{F} \boldsymbol{m} = 0 \tag{3.16}$$

Esto puede ser debido al propio método de selección de dichos puntos (ya sea manual o automático), y puede llevar a generar errores en el proceso de triangulación de los mismos. Es por esto que se corrigen los puntos seleccionados de la siguiente manera:

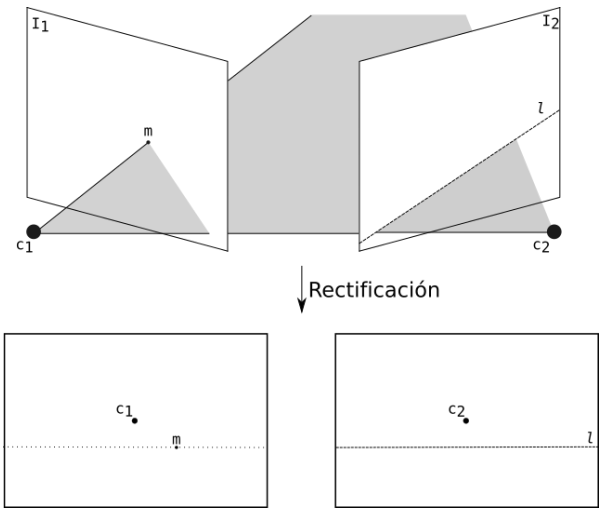


Fig. 3.5. Proceso de rectificación estereoscópica. Se busca la transformación afín entre las imágenes que permita alinear las líneas epipolares $\boldsymbol{l}$ de cada punto m, así como alinear los centros de cámara C_1 y C_2.

1. Considerar que existen los puntos $\bar{\boldsymbol{m}}$ y $\bar{\boldsymbol{m}}'$ tal que $\bar{\boldsymbol{m}}'^T \boldsymbol{F} \bar{\boldsymbol{m}} = 0$, y que se encuentran cercanos a $\boldsymbol{m}$ y $\boldsymbol{m}'$.
2. La distancia euclideana entre dos puntos está expresada por $d(p_1, p_2)$.
3. Definimos una función error que sirva para minimizar la distancia entre los puntos
$$\boldsymbol{E}(\boldsymbol{m}, \boldsymbol{m}') = d(\boldsymbol{m}, \bar{\boldsymbol{m}})^2 + d(\boldsymbol{m}', \bar{\boldsymbol{m}}')^2 \tag{3.17}$$
siempre que se cumpla $\bar{\boldsymbol{m}}'^T F \bar{\boldsymbol{m}} = 0$.

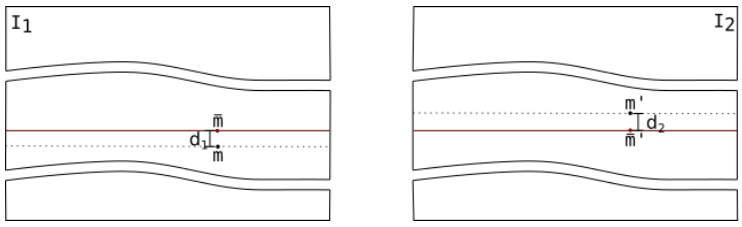


Fig. 3.6. Corrección en la correspondencia de puntos. La línea roja corresponde a la línea epipolar que minimiza $\boldsymbol{E}(\boldsymbol{m}, \boldsymbol{m}')$. Los puntos corregidos serán $\bar{\boldsymbol{m}}$ y $\bar{\boldsymbol{m}}'$.

En la figura 3.6 se muestra la línea roja, que representa la línea epipolar que minimiza la suma de las distancias d_1 y d_2, que corresponden a las distancias entre los puntos originales y los corregidos. Para la solución de este problema

se implementó un algoritmo de mínimos cuadrados para encontrar los valores óptimos.

3.2.3.3. Reconstrucción por triangulación

La triangulación nos permitirá reconstruir el punto en el espacio que se proyecta a los puntos correspondientes en cada vista. Para llevar esto a cabo se utiliza el algoritmo de Transformación Lineal Directa (o *DLT*, por sus siglas en inglés). Este algoritmo (descrito en [42]) genera una ecuación lineal de la forma $\boldsymbol{AM} = 0$ a partir del conocimiento de las matrices de proyección de ambas cámaras ($\boldsymbol{P}$ y $\boldsymbol{P'}$) y los puntos corregidos (como se describió anteriormente).
El desarrollo de este sistema de ecuaciones considera la eliminación del factor de escala mediante el producto cruz $\boldsymbol{m} \times (\boldsymbol{PM}) = 0$. Esto genera las ecuaciones

$$m_x(\boldsymbol{p}^{3T}\boldsymbol{M}) - (\boldsymbol{p}^{1T}\boldsymbol{M}) = 0 \tag{3.18}$$

$$m_y(\boldsymbol{p}^{3T}\boldsymbol{M}) - (\boldsymbol{p}^{2T}\boldsymbol{M}) = 0 \tag{3.19}$$

$$m_x(\boldsymbol{p}^{2T}\boldsymbol{M}) - m_y(\boldsymbol{p}^{1T}\boldsymbol{M}) = 0 \tag{3.20}$$

donde $\boldsymbol{p}^i$ es la i-ésima fila de la matriz de proyección $\boldsymbol{P}$, y $\boldsymbol{m} = \begin{bmatrix} m_x & m_y \end{bmatrix}$. De esto observamos que solamente las ecuaciones 3.18 y 3.19 son linealmente independientes. Además, estas ecuaciones son lineales en términos de $\boldsymbol{M}$, por lo que la ecuación $\boldsymbol{AM} = 0$ tiene como valor de A

$$\boldsymbol{A} = \begin{bmatrix} m_x\boldsymbol{p}^{3T} - \boldsymbol{p}^{1T} \\ m_y\boldsymbol{p}^{3T} - \boldsymbol{p}^{2T} \\ m'_x\boldsymbol{p}'^{3T} - \boldsymbol{p}'^{1T} \\ m'_y\boldsymbol{p}'^{3T} - \boldsymbol{p}'^{1T} \end{bmatrix} \tag{3.21}$$

La solución de dicha ecuación dará como resultado el punto tridimensional en coordenadas homogéneas.

Capítulo 4

Materiales y Métodos

En esta sección se explicarán las propuestas de métodos utilizados para desarrollar el proyecto, así como el diseño de los componentes físicos para la realización de pruebas de validación. De manera general se puede resumir la metodología en los pasos mostrados en el siguiente diagrama:

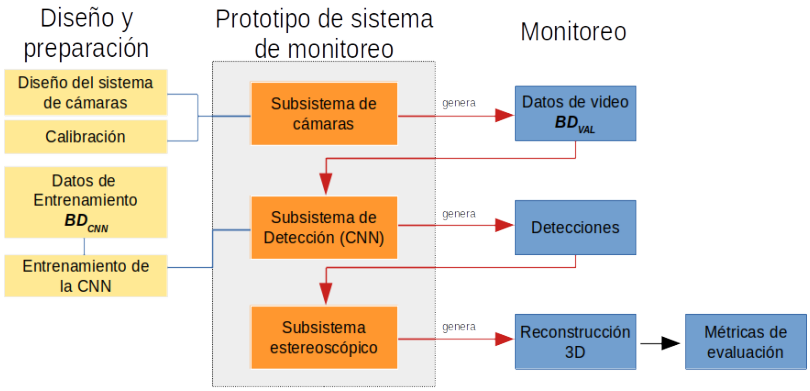


Fig. 4.1. Diagrama del proceso metodológico para el desarrollo y validación del sistema prototipo de monitoreo.

En las siguientes páginas se profundizará en cada una de las etapas que lo conforman.

4.1. Materiales

En esta sección se explicará cómo se obtuvieron los datos para el proceso de entrenamiento y calibración de las cámaras, así como los elementos que conforman al sistema de cámaras y el montaje utilizado para las pruebas y validación.

4.1.1. Bases de datos

Se utilizaron dos tipos de bases de datos: la primera se empleó para el entrenamiento del algoritmo de detección y la segunda para la validación del sistema. A continuación se explicará cómo se generaron ambas.

4.1.1.1. Base de datos para CNN (BD_{CNN})

La base de datos utilizada para el entrenamiento de la red neuronal convolucional (BD_{CNN}) proviene de dos fuentes: $BD_{CNN,1}$ que contiene imágenes variadas de peces provenientes del conjunto de datos *Open IMages Dataset* [1]; y $BD_{CNN,2}$, compuesta por un conjunto de datos de imágenes de peces subacuáticas en petenes del estado de Yucatán en condiciones no controladas. De la primera fuente se tiene una gran variedad de imágenes de peces (ver figura 4.2), tanto subacuáticas como fuera del agua (inclusive ilustraciones y fósiles). Esto da una gran variedad de características a la red que pueden servir para detectar peces de diferentes cualidades (colores, forma y posición).
La base de datos $BD_{CNN,2}$ está formada por imágenes de peces provenientes de archivos de video subacúatico en petenes (figura 4.3) anotadas por [33] provenientes del trabajo de [18]. Estas imágenes presentan una gran cantidad de ruido e inhomogeneidad de iluminación. La información que otorgan a la red permitirían trabajar en estos ambientes donde no se tiene control sobre dichas variables.
Contando el total de la base de datos en su conjunto se tienen 2767 imágenes (1883 de entrenamiento y 884 de prueba). Como es posible observar, en cada una de las imágenes podemos encontrar al menos un pez, por lo que el número de objetos para el entrenamiento es mucho mayor a los 2767. Las dimensiones de las imágenes del *Open Images Dataset* son variadas, sin embargo esto no es un problema ya que en el proceso de entrenamiento e inferencia se hace un ajuste del tamaño sin pérdida de las proporciones (como se explicará más adelante). En el caso de las imágenes de petenes tienen dimensiones de 1900×1080.

4.1.1.2. Base de datos para validación BD_{VAL}

Como se ha mencionado anteriormente, los datos serán generados por el sistema de cámaras en formato de video. Se tienen considerados dos tipos de ambientes para la obtención y creación de la base de datos: 1) $BD_{VAL,1}$ para el desarrollo de los algoritmos y la validación de estos, se obtuvieron datos en un ambiente controlado; 2) $BD_{VAL,2}$ con datos de pruebas en un ambiente semi-controlado para corroborar su robustez. A continuación se explica en qué consiste cada uno.
Ambiente controlado ($BD_{VAL,1}$)
El diseño de este ambiente implica tener control sobre las siguientes variables:

1. **Luminosidad.** Utilizando diversas fuentes de luz (asegurando una dispersión adecuada de esta), se logró contar con una escena bien iluminada, lo

[1] `https://storage.googleapis.com/openimages/web/index.html`

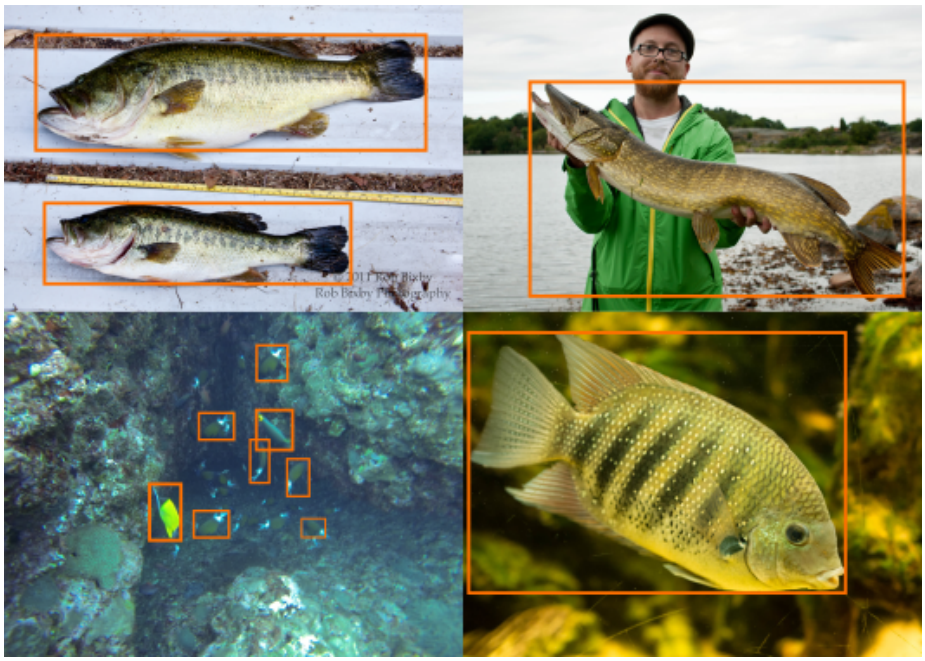


Fig. 4.2. Ejemplos de imágenes de la base de datos $BD_{CNN,1}$.

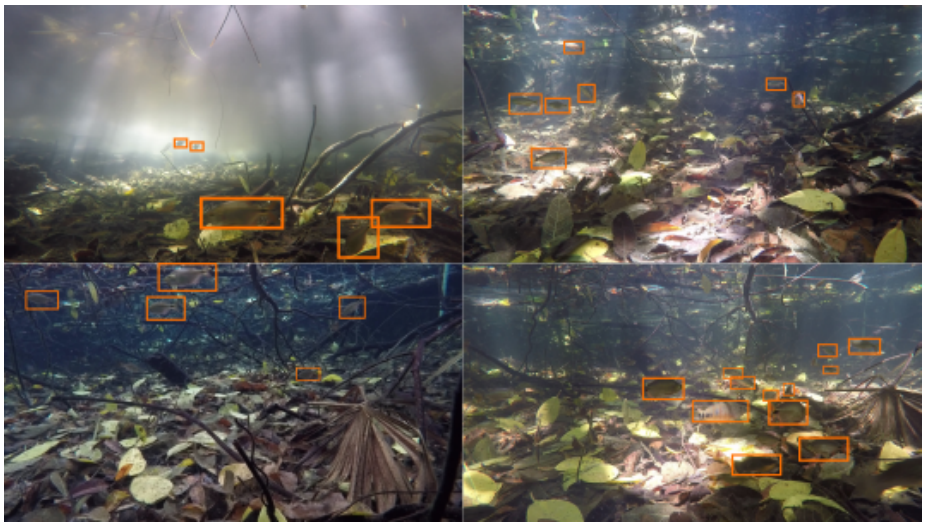


Fig. 4.3. Ejemplos de imágenes de la base de datos $BD_{CNN,2}$

que permite una captura de imágenes con el menor ruido lumínico posible y homogeneidad de iluminación.

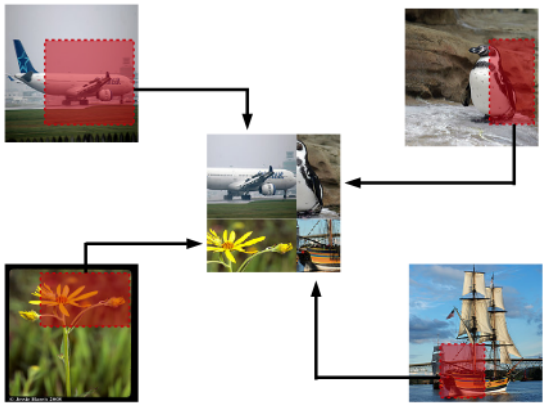


Fig. 4.4. Explicación visual de la creación de un mosaico a partir de 4 imágenes diferentes: obtenido de [45]

2. **Uso de *phantoms*.** Se utilizaron figuras de peces de dimensiones y características conocidas para la primera parte de pruebas de los algoritmos (ver figura 4.9c).

3. **Dimensiones del espacio.** Se contó con el mapeo espacial de la escena completa.

Ambiente semi-controlado ($BD_{VAL,2}$)
Las pruebas en ambiente semi-controlado se hicieron en la Unidad Académica de la UNAM ubicada en Sisal, Yucatán. En estas pruebas los peces fueron libres de moverse en su entorno y no se tuvo control alguno de su trayectoria, además la iluminación del entorno no pudo ser perfectamente garantizada. De lo que se tuvo control fue de:

1. **Dimensiones de los peces.** Los peces utilizados fueron medidos manualmente, por lo que sus dimensiones son conocidas.

2. **Dimensiones del entorno.** Se conocen las dimensiones del entorno de las pruebas y la posición del sistema de cámaras dentro del mismo.

4.1.2. Diseño experimental

A continuación se explicará cómo se diseñaron los experimentos, desde el sistema de cámaras hasta la adquisición de los datos de prueba y validación.

4.1.2.1. Sistema estereoscópico de cámaras

Para este prototipo se utilizaron dos cámaras sumergibles GoPro®: Hero 4 y Hero7. Aunque estas cámaras pueden trabajar con diferentes tipos de campos de visión (FOV), el que se utilizó en este trabajo es lineal. Los videos obtenidos con este tipo de FOV pueden tener las siguientes especificaciones:

Propiedad	Hero4	Hero7
Resolución	1080p, 960p, 720p	2.7K, 1080p
Cuadros por segundo	60, 30fps	60, 30fps
Tamaño del píxel	$1.55\mu m$	$1.55\mu m$

Tabla 4.1. Especificaciones técnicas de las cámaras para la adquisición de video.

Para que la calidad y formato de los videos obtenidos sean los mismos, se trabajó a una resolución de 1080p a 30 cuadros por segundo. Una cualidad que permite utilizarlas para este trabajo es que el tamaño de píxel es el mismo para ambas. Por otro lado, la información disponible sobre la fabricación de las cámaras no menciona las características ópticas de importancia como la distancia focal o el centro óptico; sin embargo, estas características pueden ser obtenidas como resultado del proceso de calibración de las cámaras (explicado en el capítulo 3).
Para el diseño del sistema estereoscópico se tomó en cuenta el tamaño del objeto a identificar, la distancia a la que se espera que se encuentre y la distancia focal de las cámaras. Como en todo sistema estereoscópico existe un área de sobreposición de los campos de visión (ver 4.5), todo objeto que se encuentre en esta región podrá ser visto desde ambas cámaras.
Uno de los parámetros más importantes es la línea base (o *baseline*, **B**) que está definida como la distancia entre las dos cámaras C_1 y C_2 (ver figura 4.5). Intuitivamente podemos observar que, para esta configuración, a medida que aumenta **B** el área de visión estereoscópica disminuye; sin embargo, los valores calculados de disparidad aumentan, por lo que se pueden calcular distancias de objetos a profundidades mayores. Este comportamiento se invierte al disminuir **B**.
Dado que la distancia focal y los ángulos de visión (horizontal, vertical y diagonal) no pueden ser cambiados para las cámaras que se utilizarán, la selección de la distancia entre estas determinará el área efectiva que se dispondrá para la adquisición de los datos.
Según Rovira-Más *et. al.* [46] la línea base deberá ser seleccionada con base en las dimensiones de los objetos y la profundidad a la que se encuentran. De esta manera, mientras más lejos se encuentren se requiere de una distancia focal y una línea base mayor. Utilizando la figura 4.6 podemos encontrar la relación entre número de píxeles requeridos para capturar el objeto en cuestión y la distancia a la que éste se encuentra. Esta relación está dada por:

$$\frac{L_1}{D} = \frac{L_2}{f} = \frac{mp}{f} \tag{4.1}$$

donde p es la longitud de un píxel en el sensor de la cámara (considerando un píxel cuadrado). Sabemos que p y f son parámetros de la cámara, además que m depende directamente de L_2.
Ya que se trata de cámaras con las mismas características, considerando que $\alpha_1 = \alpha_2$, podemos calcular el área de visión estereoscópica de la siguiente ma-

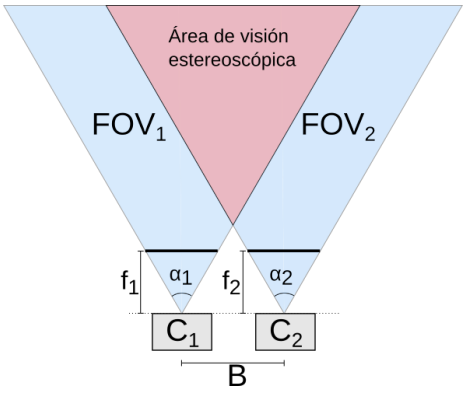


Fig. 4.5. Visualización del sistema estereoscópico. C_i representa las cámaras, α_i el ángulo de visión horizontal, f_i la distancia focal, FOV_i el campo de visión y B la distancia horizontal entre cámaras.

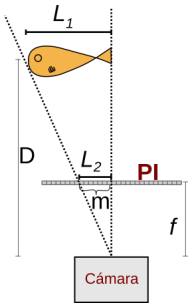


Fig. 4.6. Representación de las relaciones existentes entre el objeto y el plano de la imagen (**PI**). L_1 es la longitud del objeto, L_2 es la distancia del objeto proyectado en el PI, m es la cantidad de píxeles que ocupan L_2, f es la distancia focal y D es la distancia de la cámara al objeto.

nera:

$$A_{estereo} = \frac{\left(2\tan(\frac{\alpha}{2})D - B\right)^2}{2\tan(\frac{\alpha}{2})} \tag{4.2}$$

Es claro que cuando B es igual a cero el área será la mayor posible, sin embargo por cuestiones físicas de las cámaras, esta distancia no puede ser menor a los 10cm. Además, como se mencionará a continuación, el ambiente controlado tiene una profundidad máxima de 51cm. Con estas limitantes no es posible escoger una mejor distancia de la línea base, pero queda ya un modelo para poder utilizarse en ambientes no controlados y que los objetos se encuentren más lejanos al sistema de cámaras.

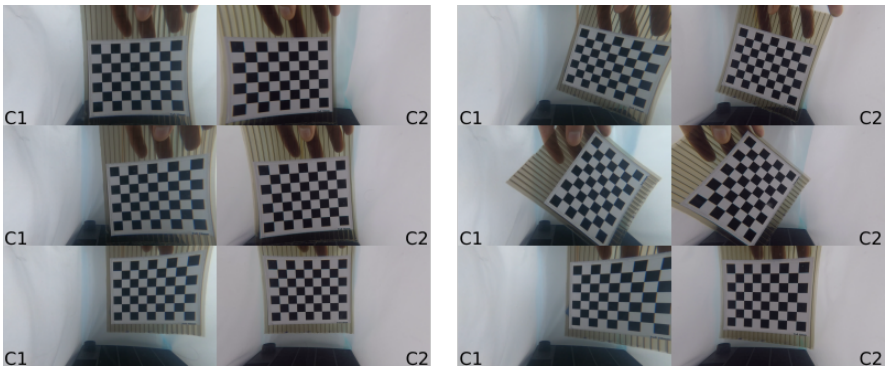


Fig. 4.7. Ejemplos de pares de capturas de las cámaras C_1 y C_2 utilizadas para la calibración del sistema estéreo.

4.1.2.2. Calibración

Para las calibraciones se utilizó un patrón de ajedrez impreso, con una lámina plastificada de terminado mate (para evitar reflejos) y colocado en una placa rígida de policarbonato. Se consideraron capturas ideales del patrón aquellas que tuvieran las siguientes características:

- El patrón debe ser captado perfectamente por ambas cámaras.
- El patrón debe abarcar cerca del 20 % del total de la imagen o más.
- El patrón debe ser lo más nítido posible y tener una iluminación homogénea (que permita la distinción de las esquinas en la imagen).

Es importante recordar que se trabaja con datos en video, por lo que la obtención de dichas capturas requiere contar con videos sincronizados (explicado en la siguiente sección). También se debe mencionar que la configuración física del arreglo de cámaras, más precisamente las posiciones relativas entre ellas, no deberán cambiar durante ni después de la calibración, de lo contrario ésta sería inválida (por la naturaleza de la misma). Es por ello que, cada vez que ocurra una modificación en las posiciones relativas de las cámaras, se debe realizar una nueva calibración.
Se considera que con 15 imágenes de buena calidad (bajo ruido, buena iluminación, patrón fácilmente detectable), es suficiente para hacer la calibración. Algunos ejemplos de capturas válidas se muestran en la imagen 4.7.
Actualmente existe software que es ampliamente utilizado para realizar la calibración de las cámaras: *Computer Vision Toolbox* de Matlab [47], y OpenCV [48]. El primero tiene desarrollada una interfaz de usuario optimizada para una calibración sencilla, rápida y amigable; ambas implementaciones se basan en el algoritmo de Zhang [43] explicado en la sección anterior. Aunque ambas implementaciones generan resultados similares, se utilizó la versión de Matlab para realizar la calibración.

4.1.2.3. Sincronización de los videos

Se tomó la decisión de trabajar con video por diversas razones, entre ellas destacan las siguientes:

1. Se desea monitorear peces que pueden moverse libremente en su medio. Esto quiere decir que trabajar con fotografías representa un gran reto dado que no podemos asegurarnos de tener una captura de buena calidad, sin elementos borrosos o incompletos (solo haber capturado una sección del pez en la imagen).

2. La estimación de las dimensiones de un pez pueden ser poco certeras con una sola imagen. Si trabajamos con video, podemos hacer un estimado de dichas dimensiones con un conjunto de imágenes que correspondan al mismo pez y mejorar la precisión de las mediciones.

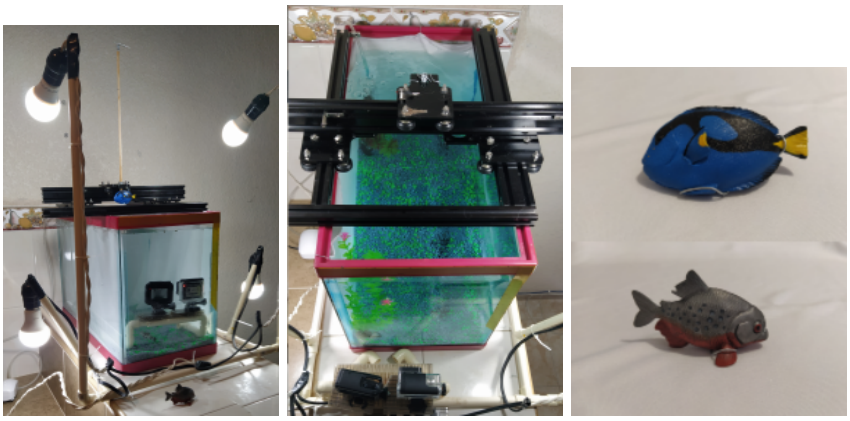
3. Trabajar con video puede permitir recuperar las trayectorias de los peces identificados, información que pudiera ser de utilidad, aunque no forma parte principal de este trabajo.

Si bien trabajar con video nos otorga algunas ventajas, también implica otras dificultades más. En nuestro caso los datos son obtenidos a partir de archivos de video, no de capturas de video en tiempo real. Esto implica tener que hacer la sincronización de los videos para poder hacer las mediciones. Para ello, al inicio de cada grabación se genera un *evento de sincronización* visual, asegurándonos que sea captado por ambas cámaras, fácilmente identificable y puntual (que pueda diferenciarse entre diferentes cuadros del video). Un ejemplo de este evento puede ser apreciado en la figura 4.8. Con este método puede fácilmente identificarse el cuadro del video en el que esto sucede para ambas cámaras y entonces sincronizarlas.

Fig. 4.8. Evento de sincronización utilizando un estímulo visual (luz) frente al sistema de cámaras.

4.1.2.4. Montaje en ambiente controlado

El ambiente controlado (figura 4.9b) se compone de una pecera de 26cm de ancho, 35.5cm de alto y 51 cm de largo, 4 fuentes de luz y un sistema de rieles. Con los últimos dos elementos se controló la uniformidad lumínica y la posición de los *phantoms* usados en las pruebas, respectivamente. Cada eje de rieles está graduado para poder conocer la posición exacta del *phantom* en cada momento y garantizar la repetibilidad de los experimentos.



(a) **(b)** **(c)**

Fig. 4.9. a) Montaje del ambiente controlado, se aprecia en la parte superior el sistema de rieles y la iluminación. b) Sistema de rieles que permiten controlar la posición de los *phantoms* c) *Phantoms* utilizadas para las pruebas, se cuentan con dos diferentes: pez cirujano (arriba) y piraña (abajo).

Tomando como referencia el diagrama mostrado en la figura 4.10, se trabajó sobre una sola posición en el eje Y (210mm) la cual coincide con la altura a la que se encuentran los centros ópticos de las cámaras; con esto se asegura la menor distorsión posible por efectos de las lentes. Al mantener fija la altura a la que se colocarán los *phantoms*, se genera un plano de posiciones (representado por el plano azul), por lo que no se tendrán datos de todos los lugares disponibles en el ambiente.
Para las pruebas el objeto se posicionó en 30 locaciones distintas, marcadas por las **X**. Recordemos que la calibración del sistema estereoscópico toma como referencia a la cámara izquierda (C_1), por lo que se utilizarán sus coordenadas para encontrar la posición relativa de cada punto al sistema estéreo.

4.1.2.5. Montaje en ambiente semi-controlado

Se utilizaron peceras con paredes opacas, de dimensiones de 35cm de ancho, 30cm de alto y 52cm de largo. Ya que las condiciones de iluminación no eran suficientes, se colocaron dos focos en la parte superior para iluminar directamente a los peces. En este montaje no fueron necesarios los rieles, ni la tela

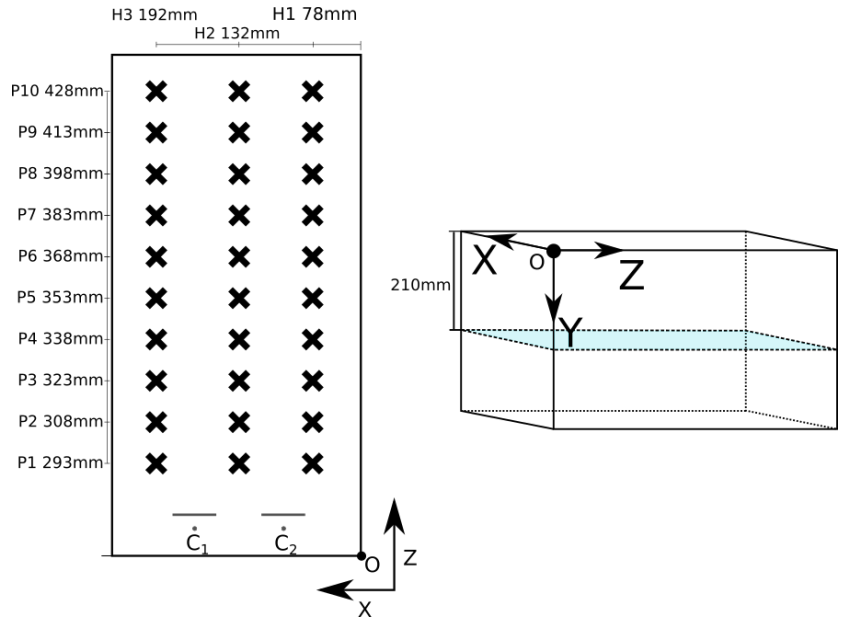


Fig. 4.10. Posiciones utilizadas durante las pruebas. El origen del sistema de referencia se encuentra en una de las esquinas superiores de la pecera. El *phantom* se colocó a una posición en el eje Y de 210mm. C_1 y C_2 representan las cámaras izquierda y derecha respectivamente.

en las paredes (ver figura 4.11). Dentro de la pecera se colocaron elementos, como trozos de tubos de plástico, que utilizaban los peces algunas veces como escondite y que les generaba confianza para nadar cerca de ellos.

4.2. Metodología

El procesamiento y análisis de los datos se hará de manera computacional, y se puede dividir en dos partes: la detección automática del pez y la reconstrucción tridimensional de la escena. Con esto último nos referimos a la ubicación espacial de puntos de interés en un espacio tridimensional. A partir de estos se calculará tanto la longitud del pez como su distancia al sistema de cámaras. A continuación se detalla cada una de las partes.

4.2.1. Procesamiento de datos

El procesamiento depende de las características de los datos obtenidos, pues en algunas ocasiones la iluminación o el ruido ambiental (partículas suspendidas, reflejos, rayos de luz, etc.) impiden realizar el proceso de detección satisfactoriamente. Para procesar los datos es común utilizar:

(a)

(b)

Fig. 4.11. Montaje de luces en el sistema semi-controlado. Se aprecia que desde una vista superior los peces son perfectamente visibles.

- Normalización de los histogramas para ajustar las intensidades de luminosidad.
- Eliminación de ruido por morfología matemática o filtros pasa-bajas.
- Uso del filtro *unsharp-masking* para hacer más nítidos los objetos.

Para el caso del ambiente controlado no fue necesario preprocesar los datos, ya que las condiciones de luminosidad y ruido fueron ideales para que las detecciones se llevaran a cabo satisfactoriamente.
Por otro lado, para el ambiente semi-controlado tanto la luminosidad como el ruido afectaron notablemente los resultados de las detecciones. Para corregirlo se hizo el siguiente preprocesamiento:

- Ajuste por reescalamiento sigmoidal [49]. Esto permite aumentar el contraste de las imágenes equilibrando la intensidad de las mismas. La ecuación que se utilizó fue

$$I'(x,y) = \frac{1}{1 + e^{g(c - I(x,y))}} \tag{4.3}$$

  donde I' es la nueva imagen, g es la ganancia y c es el valor de corte. En este paso se utilizaron los parámetros $g = 10$ y $c = 0{,}5$, pues fueron los que mejores resultados mostraron para las características de estos datos.

- Emparejamiento de histogramas [2]. Se ajusta la intensidad de los píxeles de la imagen de la cámara derecha para que empate con la cámara izquierda, de esta manera ambas vistas tendrán niveles de intensidades para los píxeles correspondientes muy parecidas.
- Filtrado pasa bajas. Se utiliza un filtro gaussiano de 3×3 con $\sigma = 2$ para eliminar parte del ruido que dejan los pasos anteriores.

Con esto se perciben de mejor manera los peces en los videos, permitiendo que su detección automática sea mucho más precisa.

4.2.2. Detección de objetos por aprendizaje automático

La detección automática es fundamental para que el prototipo funcione sin intervención humana, por lo que es de suma importancia que la detección sea rápida y precisa. Ernesto Castillo hizo el entrenamiento de una red neuronal convolucional tipo Mask R-CNN [33], la cual es capaz de localizar y segmentar peces encontrados en imágenes subacuáticas de petenes. Sin embargo, el tiempo de inferencia es prohibitivo para utilizarse en un sistema en tiempo real.
Existen otros tipos de arquitecturas de redes neuronales artificiales que pueden localizar objetos tan rápidamente como para usarse en el prototipo propuesto. Para este proceso se optó por utilizar una red neuronal artificial tipo YOLO [50], precisamente un modelo tipo YOLOv5 [51]. Estas arquitecturas de redes han demostrado tener desempeños tan buenos como para ser implementadas en sistemas de detección en tiempo real. En la figura 4.12 podemos ver un comparativo sobre el desempeño de esta familia de arquitecturas con respecto a otra familia de arquitecturas llamada *EfficientDet*. Esta familia de redes tiene como objetivo "*construir una arquitectura de detección escalable con una alta exactitud así como eficiencia*" [52], y demostraron poder superar en desempeño a otras redes para detección tal como *RetinaNet*, *Mask R-CNN* o *YOLOv3*.
Con estos resultados, se tiene evidencia de que los tiempos de inferencia para las arquitecturas YOLOv5 permiten su uso en sistemas de detección en tiempo real sin sacrificar su exactitud; y aunque en este trabajo no se planea hacer el procesamiento en tiempo real, queda fundamentado el proceso, junto con el algoritmo de detección, para escalar este prototipo a uno de análisis en tiempo real.

4.2.2.1. Arquitectura de la red YOLOv5

La familia de redes YOLOv5 cuenta con 4 arquitecturas predeterminadas: **s**, **m**, **l** y **x**. Cada una de ellas se diferencia por su profundidad, siendo la **s** la de menor cantidad de capas y la **x** la más profunda. Una de las principales ventajas de esta implementación, es que el usuario puede modificar libremente la arquitectura de la red, de una manera fácil e intuitiva. En este trabajo se utilizó la configuración **m**, ya que es evaluada con una alta puntuación de precisión promedio (AP) sin sacrificar la velocidad de detección (ver figura 4.12).

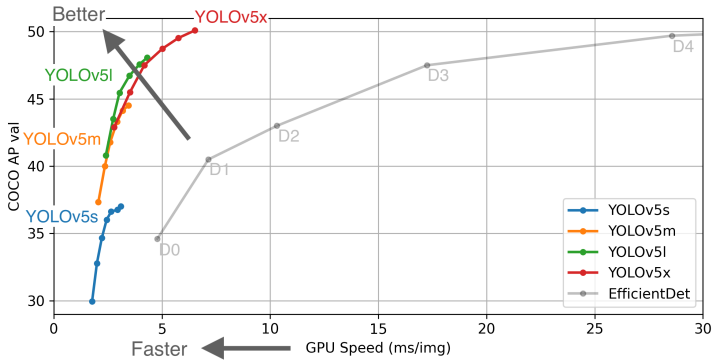


Fig. 4.12. Desempeño de las arquitecturas YOLOv5 contra la familia de arquitecturas *EfficientDet* [52], sobre 5000 imágenes del conjunto de evaluación COCO val2017: obtenida de `https://github.com/ultralytics/yolov5`

La arquitectura utilizada es la mostrada en la figura 4.13. En esencia se compone de ciertos bloques que se reutilizan y que se explican a continuación.

- **Focus.** Esta primer capa hace una reducción tipo *SpaceToDepth*, tal como se menciona en [53]. Esta reducción se lleva a cabo dividiendo la imagen en 4 bloques, los cuales son concatenados y pasados a una capa convolucional que devuelve los canales correctos (en este caso 64) para la siguiente capa. Este primer proceso es más rápido y tiene menor pérdida de información que otras capas donde solamente se hace una convolución con reducción de resolución.

- **Conv.** Se compone de una capa de convolución seguida de una normalización por lote (para el entrenamiento, según [54]), y una capa de activación SiLU [55] expresada por

$$silu(x) = x * \sigma(x) \tag{4.4}$$

  donde σ es la función sigmoide logística.

- **CSP 3 (*Cross Stage Partial*).** Este bloque está basado en la arquitectura propuesta por [56]. Se compone de una capa de cuello de botella (*Bottleneck*) en conjunto con tres operaciones de convolución. Estos bloques han demostrado disminuir el costo computacional sin sacrificar el desempeño de la red (incluso mejorando la precisión de tareas como la clasificación).

- **SSP (*Spatial Pyramid Pooling*).** Esta capa permite que la red acepte imágenes de cualquier dimensión, pudiendo así manejar cualquier tipo de escalas y razones de aspectos (*aspect ratios*). Su implementación está basada en [57].

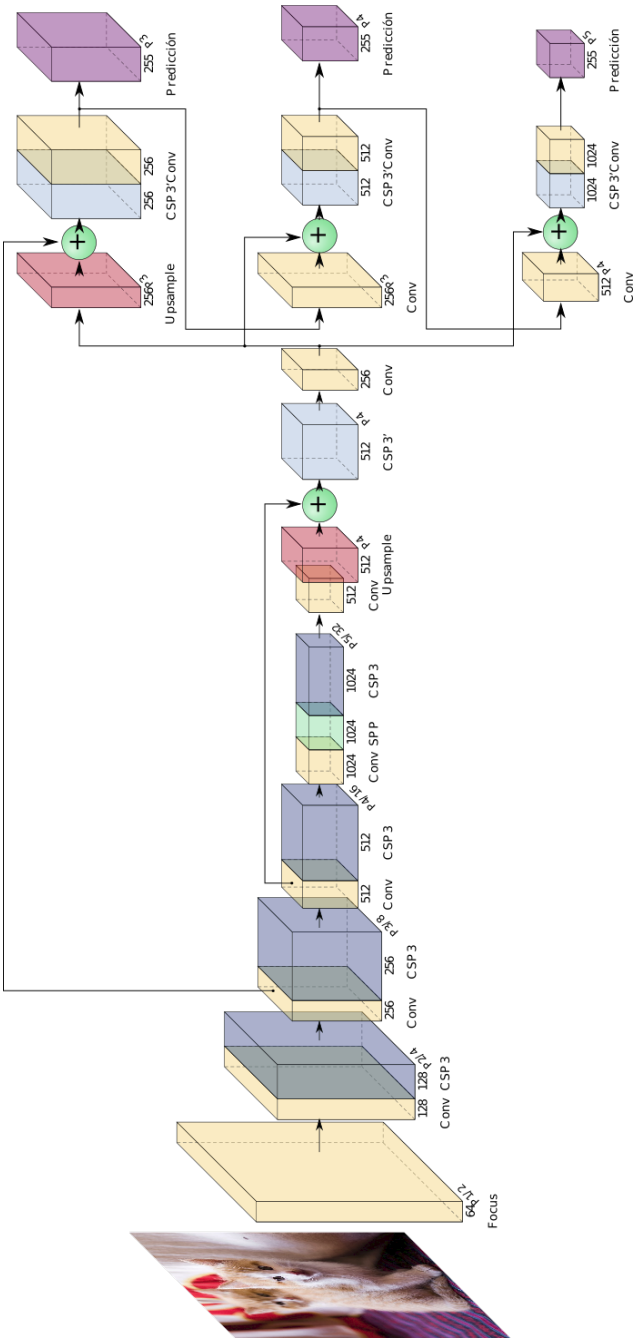


Fig. 4.13. Arquitectura *Yolov5l*.

Finalmente, se tienen 3 capas de salida, cada una a diferentes resoluciones (P3, P4 y P5). Esto le permite a la red hacer detecciones de objetos muy pequeños (P3), medianos (P4) y grandes (P5). Cada celda (vector) de estos tensores se compone por 3 secciones formadas por el siguiente vector $(t_x, t_y, t_w, t_h, P_0, p^T)^T$, donde t_x y t_y son las predicciones del centro de la caja predicha, t_w y t_h representan la altura y el ancho de la caja, P_0 es la probabilidad que la caja contenga un objeto y p es el vector de clasificación (que predice la clase a la que pertenece el objeto identificado). Son 3 secciones, ya que se trabajan con 3 tamaños de cajas predefinidas (*anchor boxes*), y cada celda se encarga de ajustarlas todas. Para la localización se considera la sección que contenga la mayor puntuación en p^T.
Posteriormente se calculan las coordenadas relativas de la detección en la imagen original, como se muestra en la figura 4.14. Finalmente, los resultados son filtrados por el método de supresión no máxima (*Non-Maximum Suppresion*), para eliminar las regiones propuestas que puedan representar el mismo objeto y estén superpuestas. Este proceso termina generando un arreglo de $n \times (5 + c)$, donde n es el número de detecciones válidas y c el número de clases (para el prototipo propuesto se consideró 1).
Para este trabajo se utilizó un proceso de transferencia de aprendizaje para no entrenar la red desde cero. Para ello se utilizaron los pesos pre-entrenados del modelo YOLOm (entrenado con la base de datos COCO val2017), el cual consta de 21,037,638 parámetros que se afinarán para trabajar en la detección de peces (la red pre-entrenada no tiene esa capacidad).

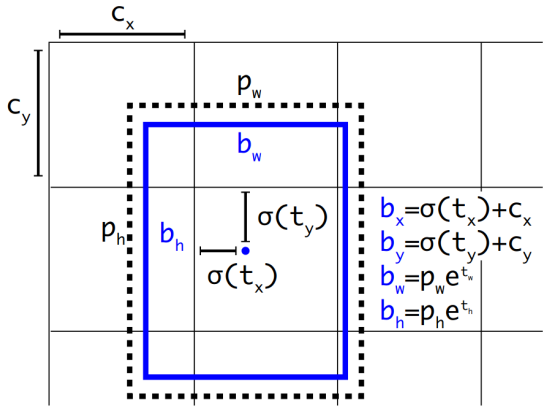


Fig. 4.14. Cálculo del centro y dimensiones de la caja de detección del objeto. c_x y c_y corresponden a las coordenadas de la celda responsable de la detección. p_w y p_h son las dimensiones predefinidas del *anchor box* utilizado: obtenida de [50]

4.2.2.2. Entrenamiento

El entrenamiento se llevó a cabo en un equipo con tarjeta gráfica NVIDIA Testa T4, con 2 procesadores Intel Xeon a 2.2GHz y 13 GB de RAM. El entrenamiento involucró 21,037,638 parámetros y tomó 1000 épocas. Durante este se ajusta el tamaño de las imágenes a dimensiones de 448x448 píxeles. En estas imágenes se hacen dos tipos de aumento de datos: en el espacio de color HSV, y en mosaico. El primero modifica el espacio de color de la imagen, creando una variación de la original que mantiene sus características espaciales. El segundo está basado en [45], donde se propone la creación de una nueva imagen a partir de 4 diferentes (ver figura 4.4). El proceso de selección del conjunto de imágenes, así como de las dimensiones de los cortes es completamente aleatorio. La figura mosaico obtenida cumple con las dimensiones para entrenamiento antes mencionadas.
El entrenamiento se lleva a cabo utilizando como optimizador el método de Descenso por el Gradiente Estocástico (SGD por sus siglas en inglés); como función de pérdida utiliza una función Binaria de Entropía Cruzada Sigmoidal

$$l(x,y)_n = -w_n \left(y_n \log \sigma(x_n) + (1-y_n)\log(1-\sigma(x_n))\right) \tag{4.5}$$

donde w_n es el peso de la neurona, y_n es el valor esperado, x_n es el valor predicho y $\sigma(v)$ es la función sigmoidal.
Al ser una tarea de localización, se evalúa el resultado de la red utilizando la intersección sobre la unión (*Intersection over Union*, o IoU). Este índice se calcula de la siguiente manera:

$$IoU = \frac{\text{Área de la Intersección}}{\text{Área de la Unión}} \tag{4.6}$$

Así, mientras la detección sea más cercana a la esperada, el valor será más cercano a 1. Ya que se está evaluando qué tan bien la red aproxima una región en toda la imagen, suele pasar que las predicciones no se ajusten perfectamente a los datos, por lo que es común tener valores menores a 1, pero mayores a 0.5. Con esta métrica, y definiendo un umbral, se puede definir cuando una detección es positiva o negativa. En este caso se utilizó un umbral de 0.6, por lo que detecciones con un IoU menor son consideradas negativas.
Así, se pudo calcular tanto la precisión como la sensitividad (*recall*) de la siguiente forma:

$$\text{precisión} = \frac{Verdaderos\ Positivos}{Verdaderos\ Positivos + Falsos\ Positivos} \tag{4.7}$$

$$\text{sensitividad} = \frac{Verdaderos\ Positivos}{Verdaderos\ Positivos + Falsos\ Negativos} \tag{4.8}$$

4.2.3. Selección de puntos de interés

Si bien existen diferentes maneras de medir la longitud de un pez, se utilizará una aproximación a la longitud total del mismo. Para ello se parte de la suposición

de que el localizador de peces devuelve una región rectangular que contiene completamente al pez, por lo que el contorno de este toca en al menos un punto a cada uno de los lados de dicha región. En la figura 4.15 se muestran estos puntos de contacto en color amarillo. Para el prototipo propuesto se utilizarán, como primera aproximación, los puntos rojos, que corresponden a los puntos medios de los lados verticales de la región de interés.
Esta selección de puntos disminuye el costo computacional de una búsqueda precisa de los puntos correspondientes a la boca y la aleta caudal. Además, como se verá más adelante, ha demostrado generar buenos resultados cuando el cuerpo del pez está completamente horizontal y es paralelo al plano de la imagen. Es importante agregar que en ciertas ocasiones, cuando el pez no está perfectamente alineado, se llegan a presentar pequeños errores en la medición, que en todo caso pueden ser minimizados.

4.2.4. Reconstrucción tridimensional

Con las cámaras calibradas y los puntos de interés localizados, el siguiente paso es reconstruir el punto en el espacio que corresponde a dichas proyecciones (ver figura 4.16). En esta figura se muestra cómo cada punto del pez (boca y aleta) es proyectado en un punto específico de cada imagen. Son estos puntos los que son utilizados para reconstruir su correspondiente punto en el espacio.
Para realizar esta reconstrucción, primero se debe hacer una corrección de los puntos en las imágenes, ya que es muy probable que no sean perfectamente correspondientes (que no estén sobre la línea epipolar creada por el punto en la otra vista). Para lograr este primer paso, se sigue el procedimiento descrito en el capítulo 3, donde se encuentran dos puntos cercanos a los propuestos, pero que minimicen la función error 3.17.
El segundo paso requiere hacer un ajuste de los puntos, corrigiendo el efecto de distorsión radial creado por la lente. Para esto se utiliza el modelo de distorsión

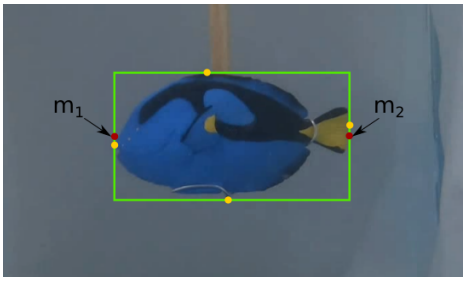


Fig. 4.15. En verde se muestra la región de interés que contiene al *phantom* de pez cirujano. Los puntos amarillos representan puntos de contacto del contorno del pez y la región. Los puntos rojos m_1 y m_2 son los puntos de interés utilizados.

de la cámara (según [42, p. 190])

$$\begin{bmatrix} m_{x,dist} \\ m_{y,dist} \end{bmatrix} = L(r) \begin{bmatrix} m_x \\ m_y \end{bmatrix} \tag{4.9}$$

donde $\begin{bmatrix} m_x & m_y \end{bmatrix}^T$ son los puntos idealmente proyectados (sin considerar efectos no lineales de distorsión), $\begin{bmatrix} m_{x,dist} & m_{y,dist} \end{bmatrix}^T$ son los puntos distorsionados, $L(r)$ es un factor de distorsión y $r = \sqrt{m_x^2 + m_y^2}$ es la distancia radial desde el centro de la distorsión. El factor de distorsión es una función arbitraria, aunque comúnmente se utiliza una expansión de Taylor en potencias de r alrededor del punto $r = 0$, de la forma

$$L(r) = 1 + k_1 r + k_2 r^2 + k_3 r^3 + ... \tag{4.10}$$

donde $(k_1, k_2, ..., k_n)$ son las constantes de distorsión obtenidas de la calibración. En la calibración propuesta en este trabajo, se utilizan las primeras 3 constantes, ya que la cámara no distorsiona en gran medida a la imagen.
Con los puntos ajustados, se procede a realizar la triangulación como se describe en el capítulo 3. En resumen, podemos ver en el algoritmo 1 los pasos para realizar este proceso.

4.2.5. Estabilización de las mediciones

Durante el proceso de reconstrucción de los puntos espaciales, una gran cantidad de errores pueden suceder. Estos errores pueden pasar desapercibidos para el sistema o alterar gravemente sus mediciones. Por ejemplo, el movimiento de los objetos en la escena no es lo suficientemente rápido para cambiar drásticamente

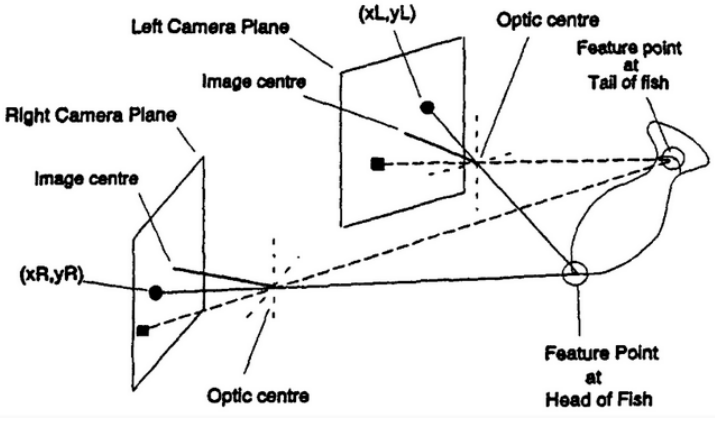


Fig. 4.16. Reconstrucción tridimensional de los puntos característicos encontrados en el pez. Utilizando esta información es posible conocer la posición espacial de cada punto: obtenida de [17].

Algoritmo 1 Algoritmo de reconstrucción tridimensional. Toma como entrada, por cada cámara i, los siguientes parámetros: el punto en la imagen m_i, la matriz de calibración P_i y los coeficientes de distorsión D_i. Devuelve el punto M.

Entrada: m_1, m_2, P_1, P_2, D_1, D_2
Salida: M
1: $(\dot{m}_1, \dot{m}_2) \leftarrow C(m_1, m_2)$
2: $\dot{m}_1 \leftarrow ajusteDistorsion(\dot{m}_1, P_1, D_1)$
3: $\dot{m}_2 \leftarrow ajusteDistorsion(\dot{m}_2, P_2, D_2)$
4: $M \leftarrow trangular(\dot{m}_1, \dot{m}_2, P_1, P_2)$

entre cuadros de video, sin embargo, en ocasiones puede ser suficiente para que las regiones detectadas entre dos cuadros consecutivos sean muy diferentes. Además, recordemos que el prototipo se diseña para utilizarse con peces cuya posición frente al sistema de cámaras no será siempre la deseada (completamente paralela al plano de la imagen), lo que naturalmente generará errores en las mediciones.
El método RANSAC (*Random Sample Consensus*) ha sido un método ampliamente utilizado gracias a que encuentra los parámetros de un modelo de manera rápida y robusta, tomando en consideración que una parte de estos datos son anómalos (lo cual define satisfactoriamente nuestro problema). En nuestro caso deseamos utilizar este método sobre el modelo representado por la ecuación (4.11). Este modelo considera solamente el promedio de las mediciones hechas, así, aquellas que sobrepasen un umbral serán consideradas como mediciones anómalas. Estas mediciones son entonces eliminadas, lo que permite estabilizar el promedio total y acercarnos al valor real de la longitud.

$$\bar{q} = \frac{1}{n} \sum_{i=1}^{n} q_i \quad , \quad q_i \in Q \tag{4.11}$$

donde $\bar{q}$ es el promedio de las longitudes, q_i es un valor del conjunto de mediciones Q y n es el total de mediciones. El problema con este modelo es que valores muy alejados del valor real harán que este estimado tenga un gran error. Para ello se necesita utilizar un método que permita estabilizar estas mediciones.
En el algoritmo 2 se explica el funcionamiento del método RANSAC para nuestro modelo. En él nos referimos a los *inliers* a los valores que se encuentran dentro del intervalo de confianza dado por θ. Una vez teniendo el mejor valor promedio $\bar{q}$ se puede hacer una evaluación de los valores en el conjunto Q, de tal forma que podemos ordenarlos por valor de desviación con respecto al promedio. Así, podemos seleccionar solo aquellos valores que no se alejen tanto del valor esperado. Esto permite aumentar la precisión en la estimación de la longitud y a estabilizarla a través del tiempo.

Algoritmo 2 Estabilizador RANSAC

Entrada: Conjunto de mediciones Q
Salida: $\bar{q}$

```
mejor_q ← Ninguno
mejor_error ← ∞
contador_inliers ← 0
contador_iter ← 0
Mientras contador_iter < max_iter Y contador_inliers < umbral_inliers Hacer
    Q_S ← subconjunto de Q
    q̄ ← calcular (4.11) sobre Q_S
    desviaciones = |Q_S − q̄|
    contador_inliers ← valores en desviaciones menores a θ
    Si contador_inliers ≥ min_inliers Entonces
        mejor_q ← q̄
        mejor_error ← max(desviaciones)
    Fin Si
Fin Mientras
```

4.2.6. Medición de múltiples peces

El módulo de detección automática permite obtener la detección de múltiples peces, por lo que se pueden obtener las mediciones de más de un pez al mismo tiempo. Para ello se debe resolver el importante problema de correspondencia estereoscópica, el cual no era necesario resolver para la reconstrucción de un solo pez, pues se suponía que los puntos encontrados debían corresponder entre sí.
Para este proceso se propone utilizar la información estereoscópica para decidir, con base en las estimaciones de la posición espacial, qué regiones de peces encontradas corresponden en ambas vistas. Los pasos son los siguientes:

1. Se obtiene el conjunto de detecciones de peces en cada vista $v \in 1, 2$ $C_v = \{c_1^v, c_2^v, c_3^v, c_m^v\}$, donde m es diferente para cada una, pues no se garantiza que se detecte el mismo número de peces en ambas.

2. De cada una de las detecciones se obtienen los puntos $\hat{m}_v = [m_{v,1}, m_{v,2}]^T$ correspondientes a los puntos medios de los lados verticales de la región.

3. Se hace una estimación del error al hacer la corrección de los puntos al hacer la reconstrucción. Recordemos que esta corrección devuelve un par de nuevos puntos $\dot{m}_i = [\dot{m}_{i,1}, \dot{m}_{i,2}]^T$ y $\dot{m}_j = [\dot{m}_{j,1}, \dot{m}_{j,2}]^T$, correspondientes a las regiones (i, j) que se están analizando. Estos nuevos puntos caen perfectamente en la línea epipolar más cercana a los puntos iniciales. Se calcula la distancia $d_{i,j}$ entre los iniciales y los corregidos de la siguiente manera
$$d_{i,j} = max(||m_{i,1} - \dot{m}_{i,2}||, ||m_{j,1} - \dot{m}_{j,2}||) \tag{4.12}$$

considerando que la distancia más pequeña nos permita identificar qué pares de regiones son correspondientes. Entonces, para cada región en C_1 se estima el error contra cada región en C_2, utilizando sus correspondientes puntos $\hat{m}_i$. Esto genera una tabla de errores que nos permite identificar qué par de regiones son correspondientes entre sí.

C_1 / C_2	1	2	...	j
1	$d_{1,1}$	$d_{1,2}$	...	$d_{1,j}$
2	$d_{2,1}$	$d_{2,2}$	...	$d_{2,j}$
...	...	...	...	...
i	$d_{i,1}$	$d_{i,2}$	...	$d_{i,j}$

Tabla 4.2. Resultado del cálculo de los errores de corrección. Mediante esta tabla se obtienen las correspondencias, considerando los valores más pequeños de distancias como un par de regiones válidas.

4. Se analiza secuencialmente cuáles regiones de la cámara 1 tienen un error menor de 30 píxeles (se llegó a este valor por experimentación, se considera un umbral suficiente para descartar un gran número de falsos positivos), calculado por la ecuación (4.12). Los pares con el menor error son considerados como correspondientes.

De esta manera se puede trabajar con un gran número de objetos detectados en ambas vistas, y aquellos cuyo error de reconstrucción sea el menor, serán considerados como el mismo objeto en el espacio.

4.2.7. Validación del sistema de monitoreo

Los resultados del sistema se validarán utilizando los datos conocidos de los tamaños de los peces utilizados, así como de las dimensiones en las que se operará el sistema. Se toman en cuenta dos mediciones: 1) la longitud del pez, y 2) la posición relativa de este respecto al sistema de cámaras. Estas estimaciones se harán con base en los puntos de interés ($m_{i,1}$ y $m_{i,2}$, $i = \{1, 2\}$) localizados en las imágenes y devueltos por el sistema de detección. Conociendo estas coordenadas se puede triangular la posición en el espacio ($M_{w,1}$ y $M_{w,2}$) de dichos puntos, gracias a técnicas de geometría epipolar.
Con la información obtenida anteriormente es de interés evaluar dos propiedades, considerando que ($M_{w,1}$ y $M_{w,2}$) se encuentran en extremos contrarios del pez. La primera corresponde a la longitud del pez (L_S), calculada de la siguiente manera:

$$L_S = ||M_{w,1} - M_{w,2}|| \tag{4.13}$$

La segunda corresponde con la distancia del punto espacial de la posición estimada del pez a la posición donde realmente se encuentra. Para esto se calcula el centro de masa del objeto $M_{w,c}$, considerando que ($M_{w,1}$ y $M_{w,2}$) se encuentran

en extremos contrarios del pez, con la siguiente ecuación:

$$M_{w,c} = \frac{1}{2}(M_{w,1} + M_{w,2}) \tag{4.14}$$

Para evaluar estas métricas se proponen dos funciones de error:

$$e_L = \left| \left(1 - \frac{L_S}{L} \right) \right| * 100 \tag{4.15}$$

$$e_D = ||M_{w,c} - M_w|| \tag{4.16}$$

donde L es la longitud real y M_w es la posición espacial relativa a las cámaras del *phantom* utilizado. e_L puede considerarse como un error porcentual que nos permite intuir rápidamente qué tan alejada está la estimación del valor real. Estas funciones tienen las siguientes características:

- El valor de error más pequeño posible es 0.
- Su evaluación siempre devolverá valores positivos.
- Son simétricas a las variaciones de las mediciones en cualquier dirección.

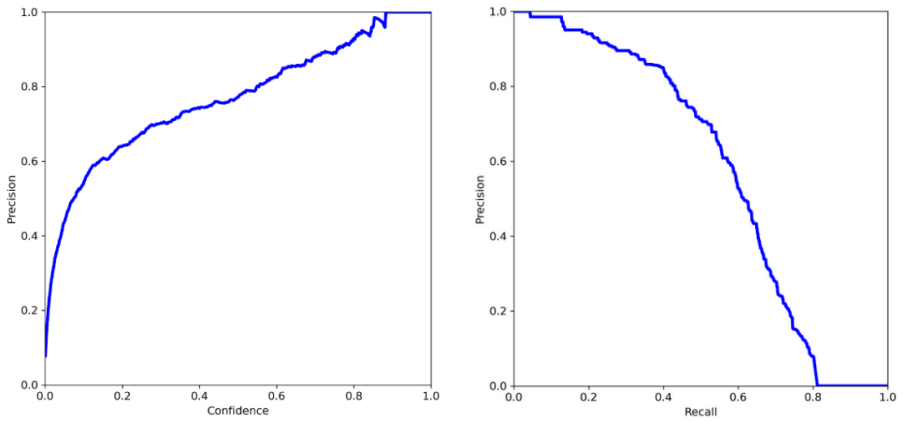
Capítulo 5

Resultados y discusión

Esta sección de resultados estará dividida en dos grandes secciones: las pruebas realizadas en el ambiente controlado y las realizadas en el ambiente semicontrolado. Para cada sección se detallará el diseño experimental y los materiales utilizados, así como los resultados en las detecciones, reconstrucción tridimensional y la validación de los mismos. A lo largo de la presentación de los resultados se hace una discusión de los mismos, así como se plantean posibles soluciones a las problemáticas encontradas.

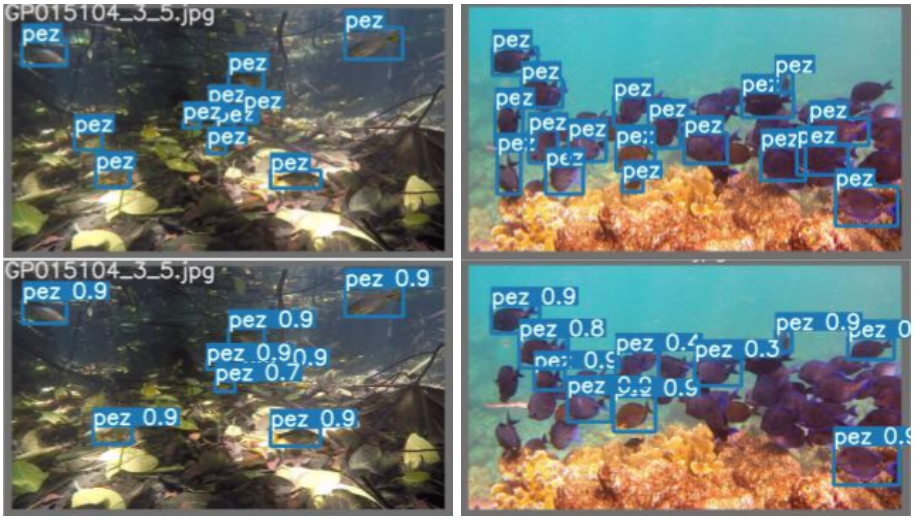
5.1. Entrenamiento de la red YOLOv5

Para evaluar el desempeño de la red se utilizaron dos métricas: la precisión y la sensitividad. La precisión nos permite saber, de todas las detecciones hechas, cuántas son válidas; mientras que la sensitividad nos da una mejor idea de cómo la red hace la detección de los peces en las imágenes. Dos gráficos son de interés para analizar estos resultados: la gráfica de precisión contra confianza, y la gráfica de precisión contra sensitividad. La primera nos muestra que considerando un umbral de confianza de 0.8 la precisión de las detecciones es de al menos del 0.8 igualmente (ver figura 5.1a). Subir este umbral de confianza genera menos error en las detecciones, aunque se pudo comprobar en los experimentos que disminuía drásticamente la cantidad de peces detectados, y en el caso de un solo pez muchas veces no era detectado. La segunda curva nos muestra qué tan buena es la red reconociendo peces. Lo deseable es que la caída de esta curva se encuentre más cercana al límite derecho, en este caso la precisión llega a un valor de 0 cuando la sensitividad es de 0.8, lo cual es considerado como un buen resultado.
Por otro lado, una visualización de los resultados del entrenamiento se muestran en la figura 5.2, utilizando las bases de datos $BD_{CNN,1}$ en la figura 5.2b, y $BD_{CNN,2}$ en la figura 5.2a). Rápidamente podemos ver que hay una gran cantidad de peces que no son detectados, a pesar de ser claramente visibles en la imagen de la derecha. En la imagen izquierda es un poco más complicado



(a) **(b)**

Fig. 5.1. Resultados del entrenamiento de la red YOLOv5. a) Relación del desempeño de la precisión respecto a la confianza de la detección. b) Relación del desempeño de la sensitividad contra la precisión



(a) **(b)**

Fig. 5.2. Visualización de los resultados del entrenamiento. En la parte superior se muestran las imágenes provenientes de los datos de prueba, y abajo las predicciones de la red.

hacer esta comparación, por el fondo de la imagen, sin embargo, en su mayoría las detecciones son correctas.
Estos resultados se consideran suficientes para trabajar en el ambiente contro-

lado, donde se tendrán las mejores condiciones de imágenes de los *phantoms*; como se mencionará más adelante, el desempeño disminuye en ambientes semi-controlados.

5.2. Ambiente controlado

Esta primera sección se enfocará en detallar el proceso de las pruebas en el ambiente controlado descrito en el capítulo 4. Antes de comenzar con los resultados finales se explicará cómo fue el proceso para el diseño del montaje, el cual tuvo diferentes etapas.
En un inicio se optó por trabajar en aire (no en medio acuático). Para esto se diseñó y fabricó una pequeña cámara de luz (ver figura 5.3), así como el uso de imágenes impresas de peces, como un punto inicial de las pruebas de detección y reconstrucción. La estructura tenía como función principal disipar la luz proveniente de los focos, así la iluminación de los objetos de prueba era uniforme y mejoraba la calidad de la detección.
Una vez teniendo una primera versión del sistema de detección y del ambiente controlado, se esperaba poder asistir al campus Sisal de la UNAM para poder hacer las pruebas subacuáticas, lo que no fue posible dada la situación sanitaria. Como solución se optó por adquirir una pecera con las dimensiones adecuadas para poder realizar las pruebas subacuáticas. De igual manera, los objetos de prueba (*phantoms*) fueron minuciosamente seleccionados dado que se buscó que tuvieran la mayor similitud visual con peces reales. Consideramos que con dos figuras de características diferentes sería suficiente para hacer las pruebas en el ambiente controlado.

Fig. 5.3. Primer montaje en aire que se utilizó. Permitió desarrollar parte de los algoritmos de detección, reconstrucción y la sincronización de los videos.

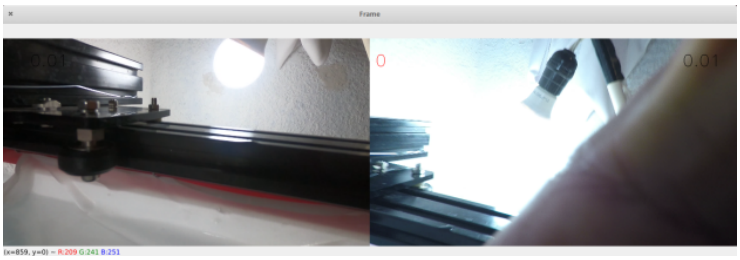
El principal problema con la pecera fue que las paredes, al ser de cristal, reflejaban los objetos que estaban dentro de ella; aún más cuando se utilizaba iluminación artificial, correspondiente al primer montaje. Para solucionarlo se colocó tela blanca por dentro de las paredes, lo que permitió utilizar iluminación indirecta (ya que se hacía pasar por la tela) en los experimentos, y eliminó completamente los reflejos.
Otro problema resultó posicionar los *phantoms* en lugares específicos de la pecera, ya que en un inicio se colocaban manualmente y no se contaba con un sistema de referencia robusto que garantizara la repetibilidad de los resultados. Como solución se utilizó un sistema de rieles graduado, así se conocería la posición exacta del objeto en cada momento del experimento. Así se logró llegar al montaje con el que se hicieron las pruebas y se presentan los resultados de esta sección.
Sobre los *phantoms* utilizados, es importante notar que las dimensiones de las figuras son más pequeñas que las de otros peces que podemos encontrar en la naturaleza. Sin embargo, el sistema propuesto no depende de las dimensiones del objeto a medir, sino de la capacidad de detectarlo en la imagen; la estimación de su longitud está adecuadamente resuelta por la reconstrucción tridimensional.

5.2.1. Calibración

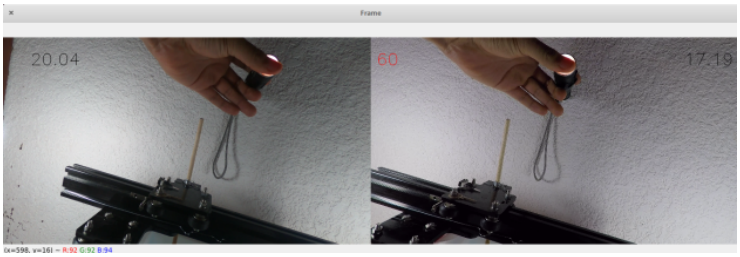
El primer paso, y uno que debe hacerse con el mayor cuidado posible, es la calibración del sistema estereoscópico de cámaras. Dado que la calibración es hecha de forma subacuática, se fabricaron patrones de ajedrez con un acabado plastificado mate que le permitió ser sumergido en agua. Además, se colocó sobre una placa de policarbonato, lo que permitió su manipulación sin distorsionarlo. Ya que trabajamos con datos de video, se hizo la grabación del patrón en diferentes posiciones dentro del ambiente controlado. Dadas las características espaciales del mismo, se tenían límites sobre dónde podía colocarse el patrón. Se tuvo que trabajar a distancias mayores a los $300mm$ desde las cámaras, ya que más cerca el patrón ocupaba mayor espacio que el de la imagen. Lo ideal es que sea completamente visible y ocupe al menos el 20 % del total de la imagen. Se realizó una grabación de 3 minutos, la cual fue procesada por un software propio (ver figura 5.4), diseñado con las capacidades de:

- Identificar el evento de sincronización y sincronizar los videos.
- Buscar el patrón de ajedrez en ambas vistas; en caso de que lo encuentre lo dibuja, en caso contrario marca un error y permite continuar con el procesamiento.
- Guardar pares de imágenes.
- Extraer secciones del video manualmente.

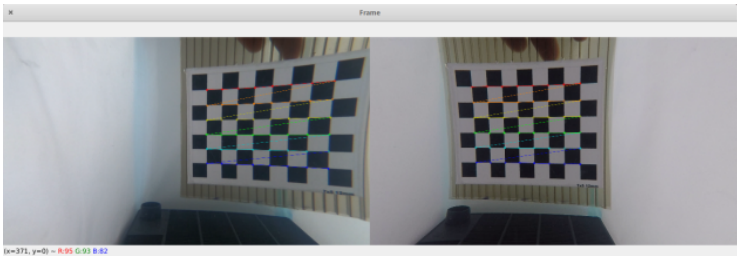
De esta manera se logró extraer 198 imágenes que se consideraron de la mejor calidad posible. Estas imágenes fueron después utilizadas para la calibración con el software "*Stereo Camera Calibrator App*" de Matlab [47]. Mediante este



(a) Captura inicial del video, sin sincronización



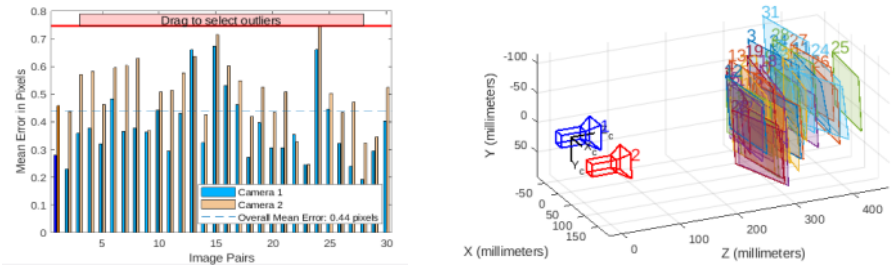
(b) Captura del evento de sincronización, a partir de aquí el video está sincronizado.



(c) Ejemplo de selección de cuadros donde se detecta el patrón de calibración.

Fig. 5.4. Capturas de pantalla del software diseñado e implementado para sincronizar y extraer video y pares de imágenes de los datos estereoscópicos.

programa se eliminaron imágenes con una alta evaluación del error de reproyección, alcanzando un valor promedio de 0.44 píxeles y 30 imágenes. Este error es el promedio de la distancia entre los puntos proyectados en las imágenes de los patrones y la posición real de los puntos en las imágenes. Tanto la distribución de los errores como la distribución espacial de los patrones de calibración respecto al par de cámaras pueden ser apreciadas en la figura 5.5.
Es importante recordar que la distribución de los patrones está limitada por el espacio físico en la pecera. Aunque pudiera parecer que esto es malo, lo que se pudo apreciar en los resultados, que se mostrarán más adelante, es que la mayor precisión de las mediciones fue alcanzada en esta región. Así confirmamos la importancia de generar una buena calibración para tener resultados confiables.



(a) **(b)**

Fig. 5.5. Resultados de la calibración. A la izquierda se muestran los errores de reproyección de cada una de las imágenes, el error promedio alcanzado fue de 0.44 píxeles. A la derecha se grafica la distribución espacial de los patrones utilizados con respecto al sistema de cámaras.

5.2.2. Adquisición de datos

Antes de adquirir los datos, se calibró el sistema de cámaras tal como se explicó anteriormente. Ya que la posición relativa entre las mismas estará fija, solo es necesario hacer una calibración, la cual sirve exclusivamente para dicha configuración. Una vez hecha esta, la obtención de los datos de video de cada uno de los *phantoms* (se trabajó con ellos por separado) se resume en los siguientes puntos:

1. Colocar el *phantom* en una de las posiciones marcadas de forma lateral, tal que se encuentre paralelo al plano de la imagen (ver figuras 5.6 y 5.7)
2. Obtener un registro de video de 5 segundos en dicha posición.
3. Cambiar a una nueva posición y repetir hasta agotar todas las posiciones.

Para el procesamiento de estos datos se diseñó un programa utilizando los algoritmos de detección automática y de reconstrucción tridimensional explicados en el capítulo 4 y programados en Python. Este programa toma como entrada el par de videos sincronizados y devuelve un historial de los peces detectados en cada cuadro, calculando su posición espacial y su longitud.
Para llevar a cabo el procesamiento de los videos, el sistema antes mencionado seguía los pasos descritos a continuación:

1. Detectar, cuadro a cuadro, el *phantom* que se encuentra en la escena.
2. Devolver los puntos de interés $(m_{1,1}, m_{1,2})$ y $(m_{2,1}, m_{2,2})$ de cada vista basándose en la región encontrada anteriormente. Cada uno de los pares corresponde: uno a la boca y otro a la aleta, aunque dada la forma de obtenerlos no es de interés saber cuál es cuál.
3. Hacer la reconstrucción tridimensional de cada par de puntos de interés $M_{w,1} = \begin{bmatrix} U_1 & V_1 & W_1 \end{bmatrix}^T$ y $M_{w,2} = \begin{bmatrix} U_2 & V_2 & W_2 \end{bmatrix}^T$, recordemos que estos dos puntos están posicionados en el espacio.

4. Calcular la posición del centro de masa del pez detectado mediante la ecuación (4.14).

5. Calcular la distancia que existe entre el par de puntos; esta es la longitud estimada L_S del objeto calculada por la ecuación (4.13).

Con este procedimiento se obtuvieron 3750 mediciones, 125 por posición, los cuales son presentados y discutidos en la siguiente sección.

5.2.3. Detección automática y puntos de interés

Para verificar que la detección funcionara adecuadamente, se revisó, cuadro a cuadro, los resultados de las detecciones en los videos obtenidos. Para este ambiente controlado, se tuvo un 100 % de efectividad en las detecciones. Es decir, que siempre que el *phantom* aparecía en la imagen, era detectado por la red YOLOv5. En el caso del *phantom* de pez cirujano, se obtuvieron niveles de confianza superiores al 85 % (ver figura 5.6). Los resultados son prácticamente iguales para el *phantom* de pez piraña, con niveles de confianza de más del 86 % (ver figura 5.7).

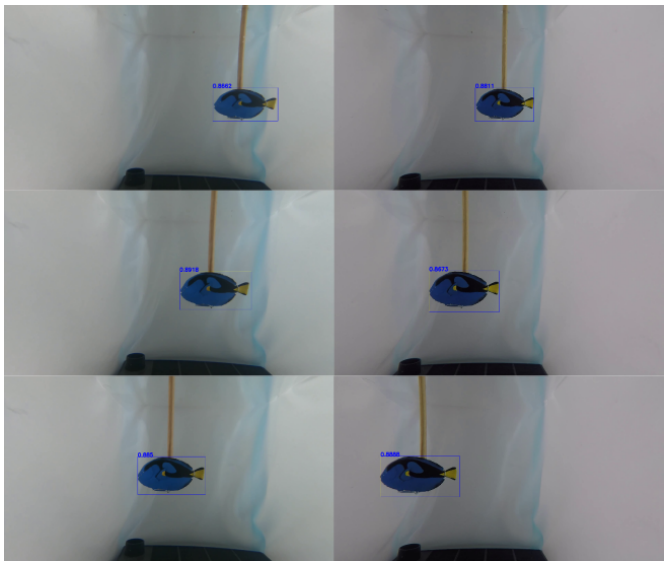


Fig. 5.6. Ejemplos de detecciones del *phantom* de pez cirujano en tres posiciones diferentes. El resto de detecciones en el video fueron tan satisfactorias como estas, logrando evaluaciones de confiabilidad de al menos 0,85.

La detección de los puntos de interés depende directamente de la calidad de la detección. Esta detección devuelve la región mejor evaluada de contener al pez. Es utilizando esta región que se seleccionan los puntos de interés correspondientes a la boca y la aleta caudal. Para esto, no es de relevancia conocer la

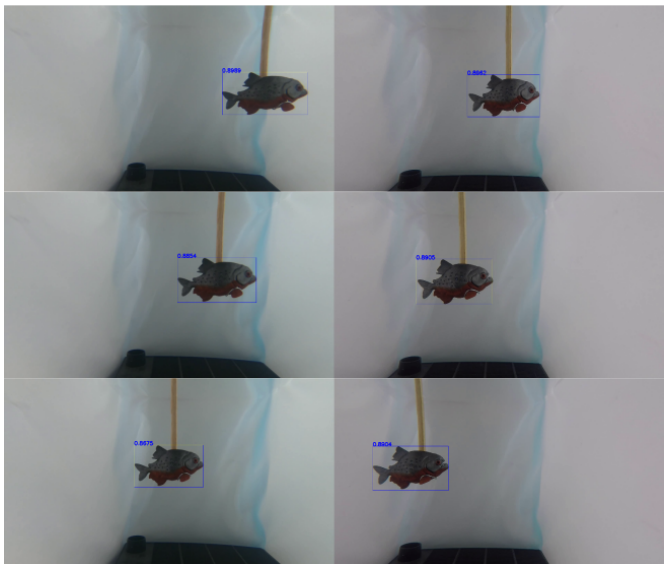


Fig. 5.7. Ejemplos de detecciones del *phantom* de pez piraña en tres posiciones diferentes. Estos resultados se repiten a lo largo del video, logrando niveles de confianza de al menos 0,82.

orientación del pez, pues solamente se tomaron en cuenta los puntos medios de los lados laterales de cada región (como se explica en el capítulo 4).
Al depender tanto de esta región de detección, llegamos a tener inconsistencias a la hora de obtener los puntos requeridos. Estas inconsistencias son generadas por una región deficiente (ver figura 5.8), ya sea por haber sobreestimado el tamaño del pez, o haberlo subestimado (generalmente la primera es la más común). Este es un problema que podría solucionarse al menos de dos maneras:

1. Mejorar la calidad de las detecciones, ya sea pre procesando las imágenes o mejorando el entrenamiento de la red.

2. Una vez teniendo la región, aproximar sus laterales hacia lo que correspondería la boca y la aleta. Esto podría hacerse buscando los bordes del pez y disminuir el tamaño de la región hasta que lo toque por ambos lados.

A pesar de esto, los resultados obtenidos son buenos y este error podría ser similar al obtenido por una persona en mediciones manuales de los peces.

5.2.4. Estimación de longitudes

Después de procesar todos los datos de video, por cada posición (H, P) se calculó el promedio de la longitud estimada dada; tanto para el pez cirujano, como para el pez piraña. Los resultados de este análisis son mostrados en la tabla 5.1. Se muestra el valor de la longitud estimada, así como la desviación estándar de

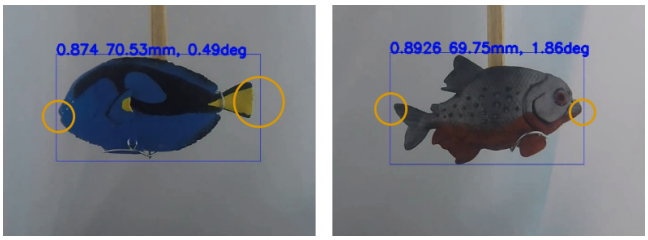


Fig. 5.8. Se muestran los errores en la localización de los peces en las imágenes. Para ambos casos existen cuadros donde la región de interés no se ajusta bien a la figura del pez. Estas diferencias, aunque parezcan pequeñas, pueden representar errores del orden de 2 o 3mm en las mediciones. En la parte superior de la detección se pueden apreciar tres valores, de izquierda a derecha tenemos: el valor de confianza de la detección, la longitud instantánea, el ángulo de inclinación del pez respecto al plano de la imagen.

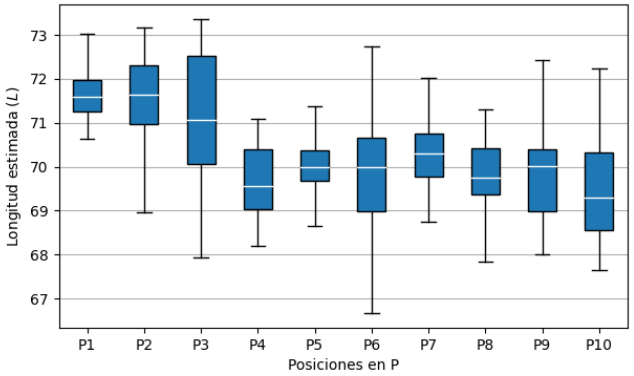
todas las mediciones. En negritas se muestran los valores máximos y mínimos por columna (es decir, por cada posición H).
A simple vista podemos notar que la variación en las mediciones no es muy grande cuando se trata de posiciones horizontales (en cada fila P), pero sí lo son en las columnas H. Esto puede intuirse al entender que a medida que la posición P aumenta, lo hace también la distancia a la que se encuentra el *phantom* de las cámaras. Gracias al proceso de calibración y los problemas encontrados en la detección de puntos de interés, se espera que esto sea así.

Tabla 5.1. Se muestra la longitud estimada promedio y la desviación estándar promedio de cada posición (H, P) (en mm). Las longitudes reales son: 70 mm para el *phantom* de pez cirujano, y 66 mm para el de pez piraña.

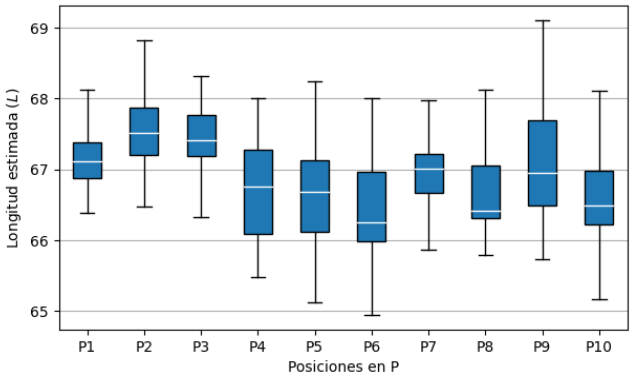
	Cirujano			Piraña		
	H3	H2	H1	H3	H2	H1
P1	71.66±0.47	71.38±0.43	**72.17**±0.93	67.90±0.63	67.05±0.31	66.99±0.25
P2	**72.28**±0.70	71.55±0.45	70.43±1.30	**67.91**±0.40	**66.99**±0.21	67.59±0.29
P3	71.48±0.70	**72.05**±1.25	69.55±0.55	66.96±0.34	**67.54**±0.24	**67.70**±0.34
P4	70.48±1.70	69.75±0.71	69.46±0.70	66.42±0.43	67.42±0.22	**66.06**±0.25
P5	70.49±0.61	70.12±0.29	**69.36**±0.59	**66.11**±0.51	67.46±0.32	66.60±0.26
P6	71.11±0.45	70.15±0.50	68.80±0.96	65.83±0.37	67.28±0.23	66.75±0.18
P7	70.46±0.87	70.40±0.53	69.52±1.33	66.62±0.71	67.01±0.32	67.09±0.26
P8	69.52±0.25	69.69±1.39	69.74±1.10	66.44±0.60	67.20±0.40	66.48±0.48
P9	69.32±0.50	70.21±0.57	69.90±1.21	67.65±0.76	67.15±0.44	66.66±0.55
P10	**68.76**±0.82	**69.66**±1.06	70.10±0.90	66.44±0.69	67.03±0.34	66.22±0.20

Ya que los resultados por fila P no parecen variar significativamente, se promediaron los resultados en este eje, obteniendo un comportamiento general según la distancia a la que se encuentren los *phantoms* de las cámaras; los resultados

se pueden revisar en la figura 5.9. Como se espera el comportamiento en ambos casos es similar, comenzando con un error relativamente alto en posiciones cercanas a las cámaras y disminuyendo a medida que se aleja. Lo más importante es que la media de las mediciones tiende a estabilizarse desde la posición $P3$, y estas son cercanas al valor esperado por cada *phantom*. El por qué de este comportamiento puede ser explicado por la misma calibración, ya que el patrón de ajedrez utilizado fue posicionado a distancias iguales o mayores a los 32cm de las cámaras (debido a las dimensiones de la pecera). Dicho lo anterior, era de esperar obtener menor error en las mediciones cercanas a la región espacial de la calibración.



(a) Longitudes estimadas del *phantom* de pez cirujano



(b) Longitudes estimadas del *phantom* de pez piraña

Fig. 5.9. Presentación de las mediciones de longitudes en gráficos de caja. Para este análisis se utilizaron las mediciones en las posiciones $H1$, $H2$ y $H3$ para cada posición en P. Recordemos que la longitud real del pez cirujano es de 70 mm y del pez piraña de 66 mm.

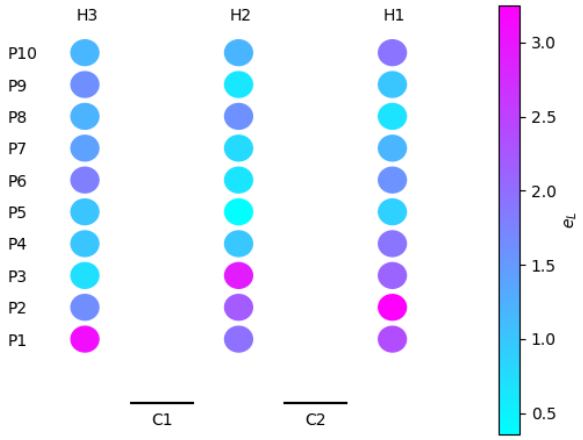
El haber utilizado gráficos de caja, nos permite darnos cuenta de que, aunque el promedio se acerca mucho a los valores esperados, existe una gran dispersión en las mediciones. Se identificó como principal razón la falta de precisión de la red neuronal al predecir la región que contiene al objeto. Sin embargo, a pesar de esto, la mayoría de las estimaciones caen dentro de una región cercana al valor de longitud real. Esto permitiría implementar algún método estadístico para estabilizar estas mediciones (como el descrito en el capítulo 4).

5.2.5. Validación de los resultados

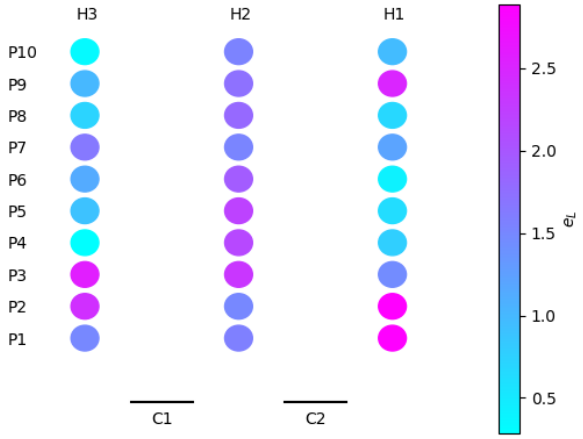
Utilizando las ecuaciones 4.15 y 4.16 se calcularon los errores e_L y e_D, respectivamente. Una representación visual de estos resultados se aprecia en la figura 5.10, en la que se muestra gráficamente la evaluación del error e_L mediante una escala de color donde el azul claro representa los valores más pequeños y el rosa los más grandes. Con esta representación podemos darnos una idea de la zona espacial donde la exactitud de nuestro método es mayor para cada uno de los casos.

Es mediante estas representaciones que podemos ver a simple vista el comportamiento del sistema propuesto. Anteriormente vimos que, en promedio, las mediciones son muy cercanas a los valores esperados. Ahora sabemos que mientras más cerca estén de las cámaras es mayor el error generado. Tal como se ha planteado anteriormente, se atribuye esto a la calibración. Recordemos que la mayor cantidad de imágenes utilizadas eran de patrones a más de 300mm de distancia, lo cual hace que a partir de $P3$ tengamos menor error en las estimaciones. De igual manera podemos ver que horizontalmente la variación de los resultados, en la mayoría de los resultados, no es significativa. Esto da pie a poder promediar los resultados en el eje H y observar claramente cómo la distancia hacia las cámaras influye en el error medido.

Entonces, podemos ver una síntesis de la evaluación de los errores en la figura 5.11, donde cada punto corresponde al promedio del error evaluado por cada medición de todas las posiciones H para cada posición P. Claramente vemos una relación directa entre los comportamientos de esta figura con los de la figura 5.9. De esta podemos notar dos comportamientos: 1) al igual que en análisis anterior, el error de estimación de longitud e_L disminuye a medida que la posición es más lejana a las cámaras; y 2) el error de estimación de la posición e_D aumenta casi de forma lineal. Lo primero es de esperarse dado que estos valores fueron obtenidos de las longitudes estimadas. Lo segundo parece ser más un problema de la resolución espacial del objeto en la imagen digital. A medida que el objeto se aleja de las cámaras, la exactitud de la reconstrucción tridimensional disminuye, ya que antes un píxel contenía la información de una región del objeto y al alejarse ese mismo píxel contiene una región más grande (más información condensada); lo que provoca errores en la recuperación de las coordenadas espaciales de los puntos de interés. Ya que el error de la región generada por la red neuronal es consistente, este error también lo es para ambos puntos, lo que hace que se calcule la posición del objeto considerando dicho error (que es relativo al sistema), pero que no afecte directamente a la estimación de



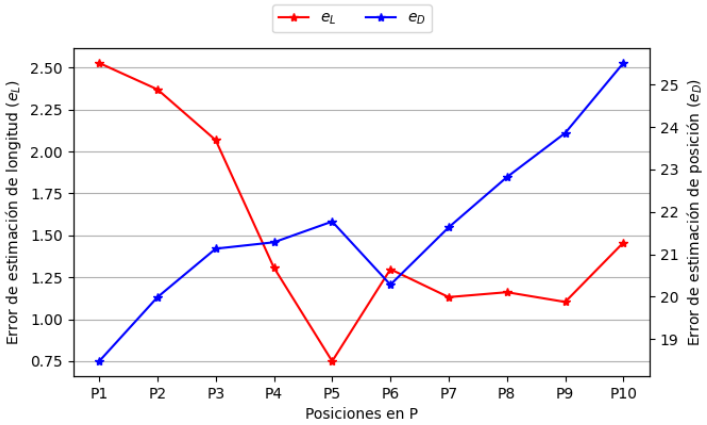
(a) Errores del *phantom* de pez cirujano



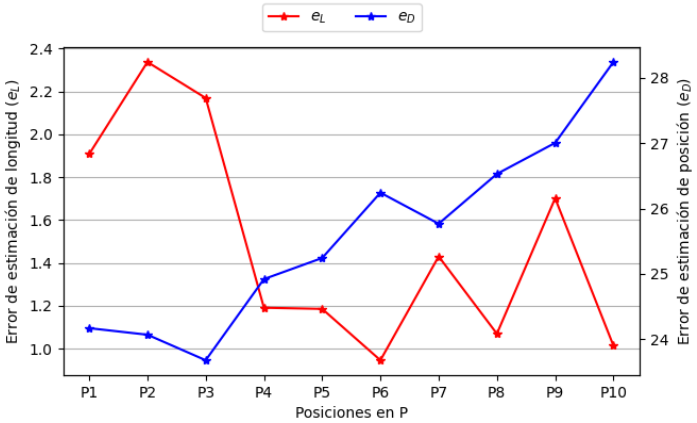
(b) Errores del *phantom* de pez piraña

Fig. 5.10. Representación gráfica de los errores por posición. Se muestra la posición relativa a las cámaras C_1 y C_2. Estos puntos corresponden con los esquematizados en la figura 4.10

la longitud (que es relativa al objeto).



(a) Errores para las mediciones del *phantom* de pez cirujano



(b) Errores para las mediciones del *phantom* de pez piraña

Fig. 5.11. Errores de las mediciones de longitud e_L y posicionamiento e_D.

5.2.6. Comparativa con trabajos similares

Como se mencionó en los antecedentes, se encontraron 2 trabajos que reportaron resultados de las mediciones individuales de peces por visión estereoscópica. Ellos calcularon el error cuadrático promedio relativo de los tamaños de los mismos, es decir, el porcentaje de la longitud real por el que su método erraba en la estimación.
En la tabla anterior se muestra la comparativa entre los trabajos antes mencionados y el aquí presentado. Podemos ver que el desempeño supera por mucho al de Rodriguez, y se acerca al de Shi. Es importante tomar en cuenta que la manera de procesar los datos fue diferente en las tres.

- Rodriguez *et al.* hicieron la detección del pez mediante la generación de un mapa de disparidad, luego lo segmentaron eliminando el fondo con una

Tabla 5.2. Comparativa de los resultados de estimación de longitud de otros trabajos con el aquí presentado

Método	Error	Procesamiento
Rodriguez *et al.* [37]	10 %	Mapa de disparidad y detección por segmentacion.
Shi *et al.* [38]	1.22 %	Mapa de disparidad y detección de puntos característicos
Método propuesto	0.75 - 2.5 %	Detección por aprendizaje automático y geometría epipolar

técnica basada en probabilidad Bayesiana y, basándose en los bordes del objeto detectado, decidían si se trataba de un pez o no. Con el mapa de disparidad hacían la medición del pez.

- Shi *et al.* obtuvieron el mapa de disparidad, con ello eliminaron el fondo y calcularon el envolvente convexo de la región segmentada. De este envolvente encontraron los puntos que coincidían con los bordes del pez, así como los puntos correspondientes al inicio de la aleta. Con estos puntos se reconstruye sus correspondientes puntos en el espacio, con lo que se puede obtener la distancia entre ellos, lo que equivale a la longitud del pez.

La principal diferencia entre estos métodos y el propuesto aquí, es que ellos trabajaron con peces vivos, mientras que aquí se trabajó con *phantoms*. Por lo tanto, no se garantiza la escalabilidad para el uso en peces vivos, donde tal vez se deban hacer algunas modificaciones. Por otro lado, en este trabajo la detección del pez está dada por la salida de la red YOLOv5, lo que hace que el tiempo de procesamiento disminuya drásticamente. En los trabajos mencionados no se especifican los tiempos de procesamiento, sin embargo, se sabe que la obtención de los mapas de disparidad es un proceso computacionalmente demandante, a comparación de utilizar una CNN. Sin embargo, la obtención de los puntos de interés es mucho más precisa en esos trabajos, pues hacen un proceso de segmentación para poder identificar con mayor exactitud la posición de la boca y la aleta caudal. Esto permite que, aunque el pez no se encuentre perfectamente alineado de manera horizontal, la estimación de la longitud sea mejor. Este es un paso que podría implementarse para mejorar los resultados del sistema propuesto.

5.3. Ambiente semi-controlado

Se realizaron pruebas en ambiente semi-controlado en el Campus Sisal de la UNAM con peces vivos. Esto fue posible gracias a David Espinosa, alumno de doctorado del posgrado en Ciencias del Mar y Limnología, quien actualmente trabaja con diferentes especies de peces en petenes del estado de Yucatán. La especie con la que se trabajó fue *Parachromis friedrichsthalii*, ya que estaban lo

suficientemente familiarizados como para no esconderse durante el tiempo en el que se hacían las pruebas, y nadar naturalmente frente a las cámaras.

5.3.1. Calibración

Antes de proceder a la grabación en la pecera con los peces, se realizó la calibración subacuática en una pecera vacía (sin peces), siguiendo los pasos mencionados en la metodología y replicados en la sección anterior (ver figura 5.12). Se grabó un video de aproximadamente 5 minutos donde se colocó el patrón en diferentes posiciones dentro del espacio disponible. De estos archivos de video se procedió a extraer 137 pares de imágenes, los cuales fueron procesados en Matlab R2021a, donde se disminuyó el error de reproyección hasta lograr tener un error promedio de 0.84 píxeles, utilizando 52 pares de imágenes al final.

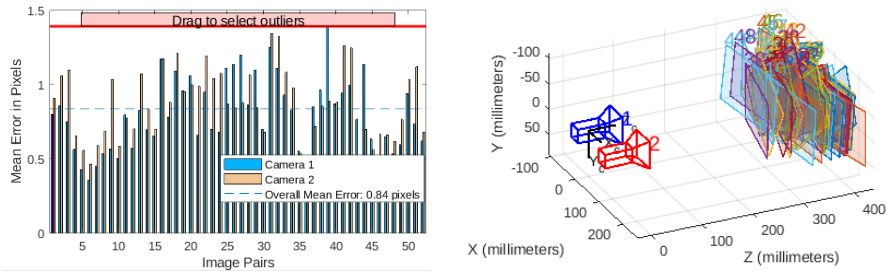
(a) **(b)**

Fig. 5.12. Capturas del momento de la calibración.

En este caso, la distribución espacial de los patrones también estuvo limitada por las dimensiones de la pecera, aunque fue posible ocupar un mayor volumen. En la figura 5.13 se muestran los resultados de la calibración, tanto la distribución de los errores de reproyección, como la distribución espacial de los patrones con respecto a las cámaras.

5.3.2. Adquisición de datos

Una vez el sistema calibrado se pasaron las cámaras a la pecera con peces, cuidando que la posición relativa entre ellas no cambiara (si no la calibración sería inservible). Antes de meterlas a la pecera se aseguró de generar el evento



(a) **(b)**

Fig. 5.13. Resultados de la calibración. A la izquierda se muestran los errores de reproyección de cada una de las imágenes, el error promedio alcanzado fue de 0.84 píxeles. A la derecha se grafica la distribución espacial de los patrones utilizados con respecto al sistema de cámaras.

visual de sincronización, luego se dejaron en un extremo, asegurándose de poder capturar la mayor cantidad de información posible.
En un inicio los peces reaccionaron tímidamente a las cámaras, aunque pronto comenzaron a posicionarse cerca de las mismas. Esto trajo la ventaja de poder captar el mayor número de peces en algunos momentos del rodaje, aunque como desventaja se tuvo: 1) la oclusión continua entre peces, y 2) algunos peces captados no se encontraban dentro del área de visión estereoscópica, por lo que solamente eran captados por una cámara. En total se adquirieron 8 minutos de grabación, que equivale a 12000 imágenes que tuvieron que ser preprocesadas por dos razones:

1. La diferencia en color e intensidades entre ambas vistas era evidente. Esto producía detecciones inexactas para ambas vistas, dificultando el proceso de encontrar la correspondencia entre peces.

2. Vistos desde la perspectiva subacuática, el color de los peces es muy similar al del fondo, por lo que la red YOLOv5 tuvo dificultades a la hora de encontrar los peces.

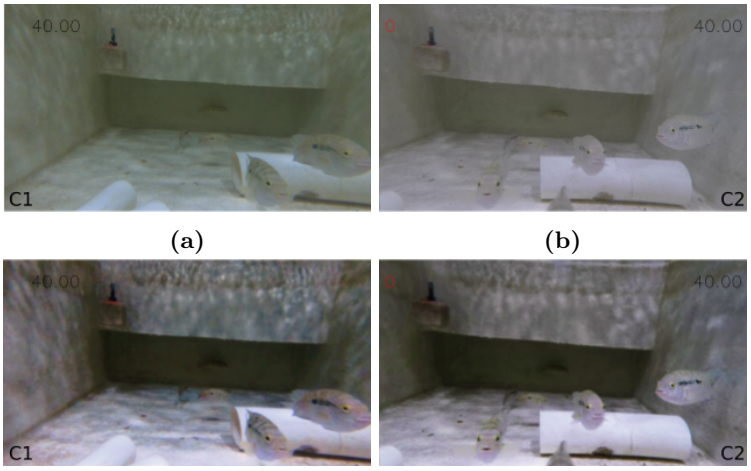
Tomando lo anterior en cuenta, se decidió procesar las imágenes de la siguiente manera:

1. Se ajustó los niveles de intensidad por medio de un reescalamiento sigmoidal, descrito por la ecuación 4.3, descrita en el capítulo 4. Esto permitió aumentar el contraste de las imágenes y equilibrar los niveles de intensidad.

2. Se utilizó un filtro gaussiano de 3×3 con $\sigma = 2$ para eliminar parte del ruido original y el generado por el reescalamiento.

3. Finalmente, para que ambas vistas tuvieran un color similar, se hizo el emparejamiento de histogramas tomando como referencia a la vista derecha.

Con este procesamiento se tienen imágenes que, a simple vista, son mucho mejores (ver figura 5.14), pero que lamentablemente (como se verá más adelante) no generaron una mejora en las detecciones como se esperaba. Cada uno de los cuadros de ambos videos fueron procesados individualmente por el sistema propuesto y descrito en la sección anterior, con el que se obtuvo el historial de los peces detectados, su posición espacial y su longitud estimada.



(a) **(b)**

(c) **(d)**

Fig. 5.14. Resultado del procesamiento de las imágenes originales. (a) y (b) son las imágenes originales de la cámara izquierda y derecha, respectivamente. (c) y (d) son las imágenes después de ser procesadas. Es evidente la mejora en contraste y color.

5.3.3. Detección automática y puntos de interés

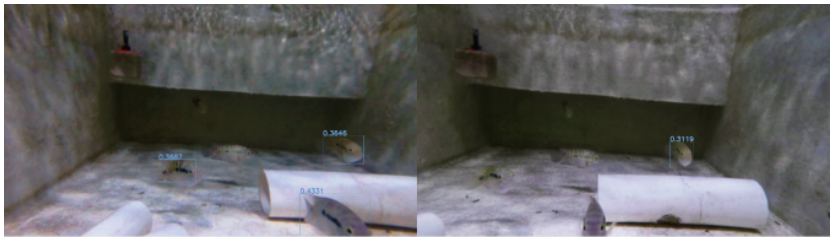
A diferencia de los resultados con el ambiente controlado, en el ambiente semi-controlado se tuvo más dificultad en la detección de los peces. A pesar de pre procesar los datos de video, y observar una mejora en la calidad de las imágenes, la red YOLOv5 no tuvo una mejora significante. Durante el proceso de detección se encontraron diversos problemas, los cuales se enlistan a continuación:

1. A pesar de poder detectar visualmente a los peces, la red no lo hacía (figura 5.15a). Analizando el por qué pasaba esto, se llegó a la conclusión de que la figura de los peces no estaba bien definida, a pesar de haber mejorado la imagen, era todavía difícil distinguir a los peces del fondo. La principal razón de esto parece ser el ruido debido a las sombras de la superficie del agua. En algunas ocasiones esos patrones de sombras eran muy similares a los de los peces.

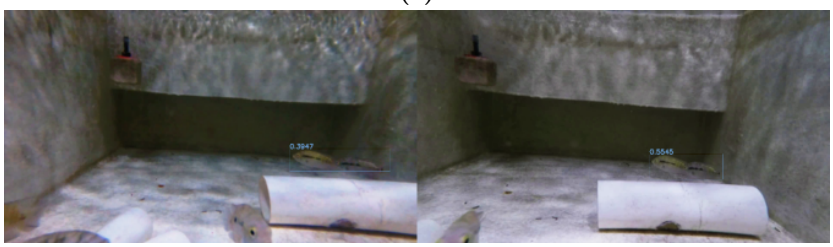
2. Dado que se trabajó con más de un pez, cuando dos o más se encontraban

muy juntos, la red llegó a hacer la detección como si se tratara de uno solo (figura 5.15b). En estos casos, cuando hacía la correspondencia con otro pez en la otra vista, se descartó la medición.

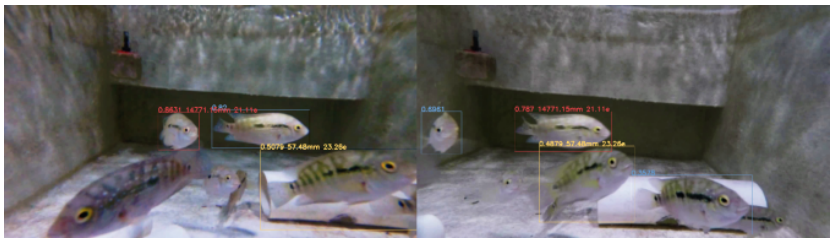
3. Aun cuando se lograba detectar al mismo pez en ambas vistas, algunas veces la correspondencia era deficiente (figura 5.15c). Esto hacía que dos peces distintos fueran identificados como el mismo, dando evidentemente resultados de longitud equivocados. Recordemos que la correspondencia está basada en geometría epipolar, por lo que no toma en cuenta la información radiométrica (niveles de intensidad) del pez localizado.



(a)



(b)



(c)

Fig. 5.15. Ejemplos de los problemas encontrados en la detección de los peces. a) La red es capaz de detectar algunos de los peces en escena, y a pesar de esto no puede encontrar correspondencias dado que no se cumplen las condiciones especificadas. b) En ocasiones, cuando dos o más peces se encuentran cerca, la red los considera uno solo. c) Al hacer la correspondencia entre los peces encontrados, muchas veces no se hace correctamente.

Una manera de poder solucionar los problemas anteriores es mejorar el entre-

namiento de la red. Por ejemplo, generando nuevos *datasets* con imágenes de los peces en las peceras. No se planteó la idea de eliminar el fondo, pues los efectos de sombras que se generan hacen que el fondo cambie drásticamente de un instante a otro. Para mejorar la correspondencia se podrían utilizar métodos de extracción de características para utilizarse como un factor al momento de decidir los pares de peces en las imágenes que corresponden al mismo en la escena. También se podría considerar el color y la orientación, como parámetros adicionales, que harían el proceso más robusto.

5.3.4. Estimación de longitudes y validación

Para poder determinar la longitud exacta de cada pez que se captura en los videos, es necesario poder identificarlos de manera precisa. Ya que los peces son muy similares entre ellos y no se encuentran marcados, esta tarea es imposible de hacer. En casos donde es imposible rastrear cada pez individualmente, Muñoz *et al.* [23, 58] hicieron un análisis de distribuciones, comparando mediciones manuales de peces contra las estimaciones hechas por sus sistemas. En este caso contamos también con dos distribuciones: una generada por las mediciones manuales de los peces; y la segunda por las estimaciones del sistema de detección automática. La forma de estos datos se muestra en la figura 5.16, donde en el eje horizontal se grafican las medidas en mm, y en el vertical la proporción en frecuencia de las medidas.

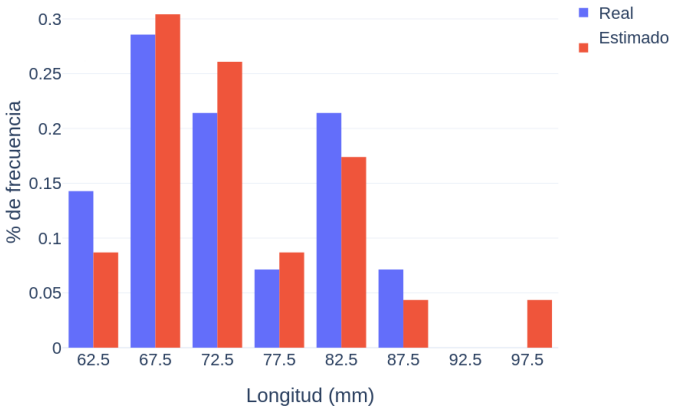


Fig. 5.16. Distribución de los datos de longitudes de peces. Se muestran los datos según la frecuencia relativa (el porcentaje respecto del número de mediciones por muestra).

Para determinar si ambas muestras provienen del mismo conjunto de datos, Muñoz *et al.* utilizaron el test Welch's ANOVA [59] para generar una métrica que de respuesta a esta pregunta. Este test es útil cuando los tamaños de las

muestras no son iguales, lo que se cumple en nuestro caso, pues no hay garantía de tomar una sola medición por pez; por ejemplo, se tienen 14 datos reales de las longitudes de peces, y 56 longitudes estimadas por el sistema. Con esta prueba se pretende comprobar si existe o no una diferencia estadística significativa entre ambas distribuciones, lo que nos permitiría confirmar que las estimaciones hechas por nuestro sistema son similares a las medidas reales de los peces.

Tabla 5.3. Comparación de los resultados del test Welch's ANOVA

Método	p-*value*
Muñoz *et al.* [23]	0.3884
Método propuesto	0.1458

En la tabla anterior se muestra una comparación entre el trabajo de Muñoz y el que se propone en este trabajo. En ambos se obtiene un valor del p-*value* mayor al 5 %, valor que es utilizado como referencia para decidir sobre la diferencia estadística significativa entre dos pruebas. La interpretación de este resultado nos permite ver que tal diferencia no es significativa entre ambas distribuciones, por lo que podemos afirmar que el sistema ha funcionado adecuadamente en la estimación de longitudes de múltiples peces en video.
Es cierto que el valor obtenido por nuestro método es mucho menor al de Muñoz, pero tengamos en cuenta la diferencia entre el número de mediciones que pudieron hacer ellos, respecto al que se logró tener en este trabajo. Se espera que con una muestra mucho mayor y después de realizar las correcciones pertinentes en el sistema de detección propuesto, este valor pueda igualmente aumentar.
Finalmente se menciona que los resultados de este trabajo fueron presentados en el XIII Congreso Mexicano de Inteligencia Artificial (COMIA 2021) en la sección "Aplicaciones del Aprendizaje Profundo". Está pendiente su publicación en la revista *Research in Computer Science*.

Capítulo 6

Conclusión

El prototipo propuesto ha demostrado generar buenos resultados en la detección y estimación de longitud de *phantoms* de peces en un ambiente controlado, obteniendo errores de entre el 0.75 y el 2.5 % de la longitud real, lo cual se encuentra cercano a lo obtenido por [38] y mejorando lo obtenido por [37]. Para el caso del ambiente semi-controlado, el desempeño disminuyó debido a diversas problemáticas que fueron satisfactoriamente identificadas. Sin embargo, con los datos que se lograron obtener se pudo confirmar el funcionamiento adecuado del sistema. Se espera que mejorando el proceso de detección el desempeño mejore sustancialmente, ya que la reconstrucción tridimensional depende directamente de la calidad de la detección, de acuerdo a la metodología propuesta.
Como trabajo a futuro se propone mejorar el entrenamiento de la red neuronal, capturando más datos en los medios semi-controlados como petenes y tanques de cultivo. De igual forma, se podrían conseguir cámaras especiales para convertirlo en un sistema de procesamiento en tiempo real, pues la detección hecha por la red YOLOv5 tiene la capacidad de hacerlo. Finalmente, se deberá continuar las pruebas para trabajar con múltiples peces en vivo en ambientes semi-controlados, lo cual implicará tomar otras consideraciones dada la forma y movimiento de los mismos para mejorar la tarea de correspondencia estereoscópica.

Referencias

[1] N. Goyal, M. Dave y A. Verma. «Data aggregation in underwater wireless sensor network: Recent approaches and issues». en. En: *Journal of King Saud University - Computer and Information Sciences* 31.3 (jul. de 2019), págs. 275-286. DOI: 10.1016/j.jksuci.2017.04.007.

[2] R. Gonzalez y R. Woods. *Digital image processing*. 3rd ed. Upper Saddle River, N.J: Prentice Hall, 2008.

[3] T. Huang. «Computer Vision: Evolution and Promise». En: *Report* (1997).

[4] S. Papert. «The Summer Vision Project». en_US. En: (jul. de 1966). URL: https://dspace.mit.edu/handle/1721.1/6125.

[5] D. Marr. «Vision: A Computational Investigation into the Human Representation and Processing of Visual Information». En: (jul. de 2010). DOI: 10.7551/mitpress/9780262514620.001.0001.

[6] Leo Breiman. «Random Forests». En: *Machine Learning* 45.1 (1 de oct. de 2001), págs. 5-32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324. URL: https://doi.org/10.1023/A:1010933404324 (visitado 06-10-2021).

[7] Corinna Cortes y Vladimir Vapnik. «Support-vector networks». En: *Machine Learning* 20.3 (1 de sep. de 1995), págs. 273-297. ISSN: 1573-0565. DOI: 10.1007/BF00994018. URL: https://doi.org/10.1007/BF00994018 (visitado 06-10-2021).

[8] Steven Walczak. *Artificial Neural Networks*. Encyclopedia of Information Science and Technology, Fourth Edition. 2018. DOI: 10.4018/978-1-5225-2255-3.ch011. URL: https://www.igi-global.com/chapter/artificial-neural-networks/www.igi-global.com/chapter/artificial-neural-networks/183727 (visitado 06-10-2021).

[9] F. Rosenblatt. «The perceptron: A probabilistic model for information storage and organization in the brain.» en. En: *Psychological Review* 65.6 (1958), págs. 386-408. DOI: 10.1037/h0042519. URL: http://doi.apa.org/getdoi.cfm?doi=10.1037/h0042519.

[10] D. Rumelhart, G. Hinton y R. Williams. «Learning representations by back-propagating errors». En: *Nature* 323.6088 (oct. de 1986), págs. 533-536. DOI: 10.1038/323533a0.

[11] K. Fukushima. «Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position». En: *Biological Cybernetics* 36.4 (abr. de 1980), págs. 193-202. DOI: 10.1007/BF00344251.

[12] P. Kim. «Convolutional Neural Network». en. En: *MATLAB Deep Learning*. Berkeley, CA: Apress, 2017, págs. 121-147. ISBN: 9781484228449 9781484228456. DOI: 10.1007/978-1-4842-2845-6_6.

[13] *National Library of Ireland*. URL: http://catalogue.nli.ie.

[14] E. Trucco y A. Verri. *Introductory techniques for 3-D computer vision*. Upper Saddle River, NJ: Prentice Hall, 1998.

[15] M. Gosta y M. Grgic. «Accomplishments and Challenges of Computer Stereo Vision». En: *52nd International Symposium ELMAR*. Zadar, Croatia, 2010, págs. 57-64.

[16] D. Li, Y. Hao e Y. Duan. «Nonintrusive methods for biomass estimation in aquaculture with emphasis on fish: a review». En: *Reviews in Aquaculture* (sep. de 2019), raq.12388. DOI: 10.1111/raq.12388.

[17] B. Ruff, J. Marchant y A. Frost. «Fish sizing and monitoring using a stereo image analysis system applied to fish farming». En: *Aquacultural Engineering* 14.2 (ene. de 1995), págs. 155-173. DOI: 10.1016/0144-8609(94)P4433-C.

[18] D. Espinosa-Mendoza. «Variaciones temporales de la comunidad de peces en un humedal costero de Yucatán, mediante imágenes subacuáticas y técnicas tradicionales». Tesis de mtría. Instituto de Ciencias del Mar y Limnología: UNAM, 2019.

[19] L. Mazzei y col. «Low cost stereo system for imaging and 3D reconstruction of underwater organisms». En: *OCEANS 2015 - Genova*. Genova, Italy: IEEE, mayo de 2015, págs. 1-4. DOI: 10.1109/OCEANS-Genova.2015.7271554.

[20] Alvaro Rodriguez y col. «Fish Monitoring and Sizing Using Computer Vision». En: *Bioinspired Computation in Artificial Systems*. Ed. por J. Ferrández-Vicente y col. Vol. 9108. Cham: Springer International Publishing, 2015, págs. 419-428. DOI: 10.1007/978-3-319-18833-1_44.

[21] C. Costa y col. «Extracting fish size using dual underwater cameras». En: *Aquacultural Engineering* 35.3 (oct. de 2006), págs. 218-227. DOI: 10.1016/j.aquaeng.2006.02.003.

[22] B. Zion. «The use of computer vision technologies in aquaculture – A review». En: *Computers and Electronics in Agriculture* 88 (oct. de 2012), págs. 125-132. DOI: 10.1016/j.compag.2012.07.010.

[23] P. Muñoz-Benavent y col. «Enhanced fish bending model for automatic tuna sizing using computer vision». en. En: *Computers and Electronics in Agriculture* 150 (jul. de 2018), págs. 52-61. DOI: 10.1016/j.compag.2018.04.005.

[24] D. Perez y col. «Automatic measurement of fish size using stereo vision». En: *2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*. Houston, TX, USA: IEEE, mayo de 2018, págs. 1-6. DOI: 10.1109/I2MTC.2018.8409687.

[25] M. Hao, H. Yu y D. Li. «The Measurement of Fish Size by Machine Vision - A Review». En: *Computer and Computing Technologies in Agriculture IX*. Vol. 479. Cham: Springer International Publishing, 2016, págs. 15-32. DOI: 10.1007/978-3-319-48354-2_2.

[26] G. Sanchez-Torres, A. Ceballos-Arroyo y S. Robles-Serrano. «Automatic Measurement of Fish Weight and Size by Processing Underwater Hatchery Images». En: *Engineering Letters* 24.4 (2018).

[27] C. Hsieh y col. «A simple and effective digital imaging approach for tuna fish length measurement compatible with fishing operations». En: *Computers and Electronics in Agriculture* 75.1 (ene. de 2011), págs. 44-51. DOI: 10.1016/j.compag.2010.09.009.

[28] H. Lu y col. «Underwater Optical Image Processing: a Comprehensive Review». En: *Mobile Networks and Applications* 22.6 (dic. de 2017), págs. 1204-1211. DOI: 10.1007/s11036-017-0863-4.

[29] U. von Lukas. «Underwater Visual Computing: The Grand Challenge Just around the Corner». En: *IEEE Computer Graphics and Applications* 36.2 (mar. de 2016), págs. 10-15. DOI: 10.1109/MCG.2016.24.

[30] M. Shortis y col. «A review of techniques for the identification and measurement of fish in underwater stereo-video image sequences». En: Munich, Germany, mayo de 2013, 87910G. DOI: 10.1117/12.2020941.

[31] F. Westling, C. Sun y D. Wang. «A Modular Learning Approach for Fish Counting and Measurement Using Stereo Baited Remote Underwater Video». En: *2014 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. Wollongong, NSW: IEEE, nov. de 2014, págs. 1-7. DOI: 10.1109/DICTA.2014.7008086.

[32] F. Shafait y col. «Fish identification from videos captured in uncontrolled underwater environments». en. En: *ICES Journal of Marine Science: Journal du Conseil* 73.10 (nov. de 2016), págs. 2737-2746. DOI: 10.1093/icesjms/fsw106.

[33] E. Castillo-Varguez. «Redes neuronales profundas para la detección de peces en ambientes subacuáticos». Facultad de Ingeniería: UNAM, 2019. URL: http://132.248.52.100:8080/xmlui/handle/132.248.52.100/17464.

[34] K. Komeyama y col. «Body Measurement of Reared Red Sea Bream Using Stereo Vision». en. En: *Journal of Robotics and Mechatronics* 30.2 (abr. de 2018), págs. 231-237. DOI: 10.20965/jrm.2018.p0231.

[35] T. Tanaka y col. «Annual monitoring of growth of red sea bream by multi-stereo-image measurement». en. En: *Fisheries Science* 85.6 (nov. de 2019), págs. 1037-1043. DOI: 10.1007/s12562-019-01347-7.

[36] F. Shafait y col. «Towards automating underwater measurement of fish length: a comparison of semi-automatic and manual stereo–video measurements». En: *ICES Journal of Marine Science* 74.6 (jul. de 2017). Ed. por Howard Browman, págs. 1690-1701. DOI: 10.1093/icesjms/fsx007.

[37] Alvaro Rodriguez y col. «Fish monitoring and sizing using computer vision». En: *Bioinspired Computation in Artificial Systems*. Ed. por J. Ferrández-Vicente y col. Vol. 9108. Cham: Springer International Publishing, 2015, págs. 419-428.

[38] C. Shi y col. «An automatic method of fish length estimation using underwater stereo system based on LabVIEW». en. En: *Computers and Electronics in Agriculture* 173 (jun. de 2020), pág. 105419.

[39] Grace W. Lindsay. «Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future». En: *Journal of Cognitive Neuroscience* 33.10 (), págs. 2017-2031. ISSN: 0898-929X. DOI: 10.1162/jocn_a_01544. URL: https://doi.org/10.1162/jocn_a_01544.

[40] A. Dhillon y G. Verma. «Convolutional neural network: a review of models, methodologies and applications to object detection». En: *Progress in Artificial Intelligence* 9.2 (jun. de 2020), págs. 85-112. DOI: 10.1007/s13748-019-00203-0.

[41] N. Aloysius y M. Geetha. «A review on deep convolutional neural networks». En: *2017 International Conference on Communication and Signal Processing (ICCSP)*. Abr. de 2017, págs. 0588-0592. DOI: 10.1109/ICCSP.2017.8286426.

[42] R. Hartley y A. Zisserman. *Multiple view geometry in computer vision*. 2nd ed. Cambridge, UK ; New York: Cambridge University Press, 2003.

[43] Z. Zhang. «A flexible new technique for camera calibration». En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (nov. de 2000), págs. 1330-1334. DOI: 10.1109/34.888718.

[44] Mark Shortis. «Camera Calibration Techniques for Accurate Measurement Underwater». En: *3D Recording and Interpretation for Maritime Archaeology*. Ed. por J. McCarthy y col. Cham: Springer International Publishing, 2019, págs. 11-27.

[45] R. Takahashi, T. Matsubara y K. Uehara. «Data Augmentation Using Random Image Cropping and Patching for Deep CNNs». En: *IEEE Transactions on Circuits and Systems for Video Technology* 30.9 (sep. de 2020), págs. 2917-2931. DOI: 10.1109/tcsvt.2019.2935128.

[46] F. Rovira-Más, Q. Wang y Q. Zhang. «Design parameters for adjusting the visual field of binocular stereo cameras». En: *Biosystems Engineering* 105.1 (ene. de 2010), págs. 59-70. DOI: 10.1016/j.biosystemseng.2009.09.013.

[47] Inc. The MathWorks. *Computer Vision Toolbox*. Natick, Massachusetts, United States, 2020. URL: https://www.mathworks.com/products/computer-vision.html.

[48] G. Bradski. «The OpenCV Library». En: *Dr. Dobb's Journal of Software Tools* (2000).

[49] G. Braun y M. Fairchild. «Image lightness rescaling using sigmoidal contrast enhancement functions». En: *Journal of Electronic Imaging* 8 (oct. de 1999). DOI: 10.1117/1.482706.

[50] J. Redmon y A. Farhadi. «YOLOv3: An Incremental Improvement». En: *arXiv* (2018).

[51] Glenn J. y col. *ultralytics/yolov5: v4.0 - nn.SiLU() activations, Weights & Biases logging, PyTorch Hub integration*. Ver. v4.0. Ene. de 2021. DOI: 10.5281/zenodo.4418161.

[52] T. Mingxing, P. Ruoming y V. Quoc. *EfficientDet: Scalable and Efficient Object Detection*. 2020. arXiv: 1911.09070 [cs.CV].

[53] R. Tal y col. *TResNet: High Performance GPU-Dedicated Architecture*. 2020. arXiv: 2003.13630 [cs.CV].

[54] I. Sergey y S. Christian. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: 1502.03167 [cs.LG].

[55] R. Prajit, Z. Barret y V. Quoc. *Searching for Activation Functions*. 2017. arXiv: 1710.05941 [cs.NE].

[56] W. Chien-Yao y col. *CSPNet: A New Backbone that can Enhance Learning Capability of CNN*. 2019. arXiv: 1911.11929 [cs.CV].

[57] K. He y col. «Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition». En: *Lecture Notes in Computer Science* (2014), págs. 346-361. DOI: 10.1007/978-3-319-10578-9_23.

[58] P. Muñoz-Benavent y col. «Automatic Bluefin Tuna sizing using a stereoscopic vision system». en. En: *ICES Journal of Marine Science* 75.1 (ene. de 2018). Ed. por Howard Browman, págs. 390-401. DOI: 10.1093/icesjms/fsx151.

[59] B. Welch. «On the comparison of several mean values: an alternative aproach». En: *Biometrika* 38.3-4 (1951), págs. 330-336. DOI: 10.1093/biomet/38.3-4.330.