Utilising Computer Vision Techniques for Automated Density and Growth Estimation in Precision Aquaculture Systems for Prawn Cultivation

By

Lean Jin Hao

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JUNE 2025

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Cheng Wai Khuen who has given me this bright opportunity to engage in an Utilising Computer Vision Techniques for Automated Density and Growth Estimation in Precision Aquaculture Systems for Prawn Cultivation project. It is my first step to establish a career in Distributed Systems field. A million thanks to you.

To my academic advisor, Dr. Lim Seng Poh, I sincerely thank you for your invaluable guidance, patience, and unwavering support throughout the course of this project. Your encouragement and mentorship have been instrumental to my growth, both academically and personally. Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

COPYRIGHT STATEMENT

This Final Year Project proposal is submitted in partial fulfillment of the requirements for the degree of Bachelor of Computer Science (Honours) at Universiti Tunku Abdul Rahman (UTAR). This Final Year Project proposal represents the work of the author, except where due acknowledgment has been made in the text. No part of this Final Year Project proposal may be reproduced, stored, or transmitted in any form or by any means, whether electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the author or UTAR, in accordance with UTAR's Intellectual Property Policy.

ABSTRACT

Prawn farming, a vital sector of the global aquaculture industry, faces challenges with traditional monitoring methods that are labor-intensive, error-prone, and lack real-time capabilities, leading to inefficiencies in feeding and harvest planning, particularly for small- and medium-scale farmers. This project aims to address these issues by developing a computer vision-based system for automated density and growth estimation of Cherax quadricarinatus prawns, enhancing operational efficiency and sustainability. Utilizing the lightweight YOLO11n neural network, a Raspberry Pi 5, and a PiCamera (Night Vision), the system automates prawn monitoring, improves accuracy through machine learning, and ensures affordability at $60-$80 per unit. A Cron Job feature enables continuous data collection, building a farm-specific dataset to overcome the lack of standardized prawn data. Deployed in a controlled pond environment, the system captured 2000 images under varying conditions, achieving real-time detection at 5 FPS, though initial tests revealed accuracy issues requiring further data and fine-tuning. By mitigating challenges like environmental variability, high costs, and technical complexity identified in prior studies, this solution offers a scalable, user-friendly tool that empowers smaller farms to optimize resource use and enhance productivity in precision aquaculture.

Area of Study: Internet of Things, Computer Vision

Keywords: Data Collection, Computer Vision, YOLOV11n, Raspberry Pi, Cherax quadricarinatus

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CAGR	Compound Annual Growth Rate
YOLO	You Only Look Once
CV	Computer Vision
CNNs	Convolutional Neural Networks
ML	Machine Learning
AI	Artificial Intelligence
FPS	Frames Per Second
RMS	Root Mean Square
R-CNN	Region-based Convolutional Neural Networks
SSD	Single Shot Detector
SFTP	Secure File Transfer Protocol
SCP	Secure Copy Protocol
FTP	File Transfer Protocol
WebDAV	Web-based Distributed Authoring and Versioning
Amazon S3	Amazon Simple Storage Service
VNC	Virtual Network Computing
mAP	Mean Average Precision
AR	Average Recall
YAML	yet another markup language

CHAPTER 1 Introduction

In this chapter, we present the background and motivation of our research, our contributions to the field, and the outline of the thesis. Prawn farming is a significant component of the global aquaculture industry, contributing to the supply of seafood worldwide [1]. The global shrimp industry has experienced substantial growth, with major production increases in countries like Ecuador, China, and Vietnam, driven by rising demand in key markets such as the US. Ecuador saw a remarkable compound annual growth rate (CAGR) of 25% between 2020 and mid-2023. However, some regions, including India, are now experiencing reduced exports due to market oversupply, highlighting the dynamic and fluctuating nature of the global shrimp market [1]. As the industry grows, there is an increasing need for precision aquaculture techniques to enhance productivity and sustainability [2]. Precision aquaculture employs advanced technologies to optimise farming practices, reducing waste and improving resource management [2].

1.1 Problem Statement and Motivation

1. Traditional methods for monitoring prawn density and growth are labour-intensive, time-consuming, and prone to human error, leading to inefficiencies in prawn farming operations:

Traditional methods for monitoring prawn density and growth are labour-intensive, time-consuming, and prone to human error, leading to inefficiencies in prawn farming operations [3]. This inefficiency can lead to overfeeding or underfeeding, resulting in wasted resources and potential harm to prawn health and farm productivity [2]. Automating these processes through advanced technologies like computer vision or machine learning algorithms can significantly enhance operational efficiency [2].

2. There is a lack of real-time monitoring tools in prawn cultivation, resulting in imprecise feeding and harvest planning, which can negatively impact the productivity and sustainability of aquaculture systems:

The absence of real-time monitoring tools in prawn farming makes it difficult to accurately track the growth of prawns and manage feeding schedules [3],[4],[5].

Feeding is often based on estimated population sizes rather than actual real-time data, leading to either overfeeding, which wastes resources and contributes to water pollution, or underfeeding, which hinders prawn growth [2]. Additionally, without continuous data on growth and density, farmers cannot optimally plan harvesting, which may lead to premature or delayed harvests, further affecting the economic sustainability of the farm [2], [5]. Real-time tools such as sensors, automated feeders, and computer vision systems are essential to address these inefficiencies by providing farmers with accurate, timely data [2].

3. Small- and medium-scale prawn farmers face challenges in adopting advanced technologies due to the high costs and complexity of implementation, limiting their ability to optimize cultivation practices:

For small- and medium-scale prawn farmers, the adoption of advanced technologies such as automated monitoring systems, machine learning algorithms, and IoT devices can be financially prohibitive [2]. High upfront costs for equipment and ongoing expenses for maintenance and upgrades create barriers to technology adoption. Additionally, the technical complexity of setting up and maintaining these systems can deter smaller operations that may not have access to skilled labor or technical expertise [2]. These challenges limit smaller farms' ability to optimize their cultivation practices, resulting in lower productivity and competitiveness compared to larger, tech-enabled operations [2].

The motivation for proposing the automated prawn cultivation system stems from the need to modernize traditional, labour-intensive practices that are inefficient and prone to human error. Current methods lack real-time monitoring, leading to inaccurate feeding and harvest planning, which reduces productivity. The system leverages advanced technologies like computer vision to provide real-time data, improving decision-making, resource management, and overall efficiency. Additionally, it aims to make precision aquaculture accessible to small- and medium-scale farmers by offering a cost-effective and scalable solution, ultimately promoting sustainability and supporting the growth of the prawn farming industry.

1.2 Project Objectives

1. Develop a computer vision-based system using lightweight neural networks like YOLO11n to automate the estimation of prawn density and growth, reducing reliance on manual methods:

This objective aims to replace traditional manual methods of monitoring prawn density and growth with an automated system powered by computer vision and deep learning. Lightweight neural networks, particularly MobileNetV2, are chosen due to their efficiency and ability to operate on resource-constrained devices, making them ideal for real-time monitoring in aquaculture settings. By training the system to recognize prawns and estimate their size and count from visual inputs such as camera feeds, the system can provide continuous, accurate data on prawn growth and population. This will significantly reduce the labor-intensive processes currently employed, cutting down human error, improving consistency, and allowing for more frequent data collection, leading to better decision-making in feeding schedules, health checks, and harvest planning.

2. Enhance the accuracy and efficiency of prawn population monitoring through the integration of machine learning algorithms:

This objective focuses on improving the precision of prawn population tracking by leveraging machine learning algorithms to handle complex visual data. By incorporating techniques such as transfer learning, deep learning, and pattern recognition, the system will be able to analyse various features such as prawn movement, size variation, and density in diverse conditions. This will lead to more reliable and accurate estimations of prawn population metrics compared to traditional observation methods. The integration of advanced algorithms will also optimize the system's performance, making it capable of processing data quickly and efficiently, even in low-resource environments. This improvement in accuracy and efficiency will enable farmers to make more informed decisions regarding feeding, pond maintenance, and prawn health management.

3. Design a cost-effective and scalable solution that can be readily adopted by small- and medium-scale prawn farmers for improved aquaculture management.

This objective targets the creation of an affordable and scalable solution tailored to the needs of small- and medium-scale prawn farmers. By keeping hardware requirements minimal and leveraging lightweight neural networks like MobileNetV2,

the system can be implemented using cost-effective equipment such as simple cameras and standard computing devices. The focus on scalability ensures that the solution can be easily adapted to different farm sizes and infrastructure without the need for expensive modifications. Additionally, the system will be designed with a user-friendly interface to ensure ease of use, even for farmers with limited technical expertise. By providing real-time monitoring and data-driven insights at a low cost, this solution will empower farmers to optimize their production processes, reduce waste, and increase profitability, making advanced aquaculture management accessible to a broader audience.

1.3 Project Scope and Direction

The project scope directly supports all three objectives specific to prawn cultivation, focusing on Cherax quadricarinatus. Firstly, it focuses on developing a computer vision-based system using lightweight neural networks such as YOLO11n, which will automate the detection and estimation of prawn density and growth. This reduces the need for labour-intensive, manual monitoring methods traditionally used in prawn farming. Secondly, by incorporating machine learning algorithms, the system will improve the accuracy and efficiency of monitoring prawn populations, allowing for precise tracking of prawn growth and population density. This ensures more effective management of feeding schedules and harvesting times, optimizing productivity. Lastly, the scope highlights designing a cost-effective and scalable solution tailored specifically for small- and medium-scale prawn farmers, ensuring that they can adopt these advanced technologies without prohibitive costs. The system will enable these farmers to improve aquaculture practices, boosting sustainability and operational efficiency in prawn farming.

1.4 Contributions

The proposed automated prawn cultivation system significantly contributes to the prawn farming industry by enhancing efficiency, accuracy, and sustainability. By automating monitoring processes through advanced technologies such as computer vision and machine learning, the system reduces the reliance on labour-intensive, error-prone methods, leading to improved operational efficiency and more precise data on prawn density and growth. This real-time data enables better management of feeding

and harvesting schedules, preventing issues like overfeeding, underfeeding, and suboptimal harvest timing, which in turn optimizes resource use and minimizes environmental impact. Furthermore, the system addresses the challenges faced by small- and medium-scale farmers by offering a cost-effective and user-friendly solution, thus making advanced technology more accessible and helping to level the playing field. Ultimately, this approach supports the sustainability of prawn farming and fosters industry growth by improving productivity and resource management across various farm sizes.

1.5 Report Organization

This report is structured into five chapters, each detailing a specific aspect of the project on utilizing computer vision techniques for automated density and growth estimation in precision aquaculture systems for prawn cultivation. The first chapter introduces the project, discussing the background of the global shrimp industry, the problem statement, motivation, objectives, project scope, and contributions. The second chapter reviews existing literature on automated monitoring technologies in aquaculture, focusing on systems like smart headsets, computer vision-based counting methods, and AI-driven image processing for prawn farming, while analyzing their strengths and weaknesses to contextualize this project's approach. The third chapter presents the proposed method, detailing the system requirements, architecture diagram, workflow, growth stage equations, and project timeline. The fourth chapter covers preliminary work, including the setup of software tools, data collection and annotation processes, YOLO11n model training and deployment, automated data collection via Cron Job, and a device comparison for cost-effectiveness and scalability. Finally, the fifth chapter concludes the report by summarizing the project's findings, contributions, and implications for prawn farming, emphasizing how the proposed solution addresses the identified challenges and its potential impact on the aquaculture industry.

CHAPTER 2 Literature Reviews

2.1 Smart headset, computer vision and machine learning for efficient prawn farm management [3]

2.1.1 Review

The field of aquaculture has increasingly adopted automated monitoring technologies to improve the efficiency of operations, particularly in prawn cultivation. Traditional methods for estimating prawn density and growth, such as the cast-net approach, are labour-intensive and error-prone. These methods often result in suboptimal data collection frequencies and inaccuracies, which can negatively impact farm productivity. Recent advancements in computer vision (CV) and machine learning (ML) have provided a more scalable and efficient solution for prawn growth monitoring. This study developed a smart headset integrated with a depth camera and CV techniques, which enabled real-time prawn size estimation during routine feed tray inspections. The study highlighted the need for high-frequency data collection to better understand prawn growth and optimize feeding and harvesting strategies. This solution demonstrated superior performance in terms of time efficiency compared to traditional methods, which require manual labour and invasive procedures that stress the animals. Other studies have explored the use of convolutional neural networks (CNNs) to detect and segment prawns for size estimation. This study [4] employed CNN models to estimate shrimp population density and size, providing real-time insights into shrimp growth trends. Their approach used a similar vision-based system, but with a focus on stationary camera setups rather than the wearable smart. The transition to wearable technology marks a significant advancement, offering greater flexibility and mobility for prawn farmers. Despite the promising outcomes, challenges remain in ensuring data quality in outdoor environments. External factors such as lighting and water reflections can introduce noise in the depth maps generated by the CV systems. This study addressed these challenges through advanced filtering and smoothing techniques to improve the accuracy of prawn length estimation. However, they also identified limitations, such as the need for further refinement of depth camera capabilities in uncontrolled field conditions.

In conclusion, the integration of computer vision and machine learning in prawn cultivation has the potential to revolutionize the industry. By automating the data collection process and providing real-time insights into prawn growth, these technologies can significantly reduce labour costs and improve farm productivity.

2.1.2 Strengths and Weakness

The study presents several strengths in its approach to prawn farm management using a smart headset integrated with computer vision and machine learning technologies. One of the key advantages is the automation of prawn size estimation, which significantly reduces the reliance on traditional, labour-intensive methods like cast-net sampling. By enabling real-time and non-invasive data collection during routine feeding activities, the solution enhances operational efficiency and minimizes stress on the animals. The use of wearable technology also introduces greater flexibility and mobility for farmers compared to previous systems that relied on stationary cameras. Additionally, the incorporation of advanced filtering and smoothing techniques helps to mitigate the impact of environmental challenges such as lighting variations and water reflections, thereby improving data accuracy.

However, the study also highlights several weaknesses. Despite efforts to improve data quality, the system's performance can still be hindered by external factors in outdoor environments, such as inconsistent lighting and reflective water surfaces. The current capabilities of depth cameras remain limited under such uncontrolled conditions, pointing to the need for further hardware refinement. Moreover, the implementation of such high-tech solutions may present a barrier for small-scale farmers due to the initial cost, technical complexity, and ongoing maintenance requirements. While the study shows promising results, its findings may be limited if the system has not yet been extensively tested in diverse or large-scale field conditions.

The proposed solution with a Raspberry Pi module it can reducing the barrier for small-scale farmer, where Raspberry Pi has a very competitive price.

2.2 Automatic counting methods in aquaculture: A review [7]

2.2.1 Review

Accurate monitoring of prawn density and growth is critical for the effective management of aquaculture systems. Traditional manual methods are labour-intensive and prone to human error, making them unsuitable for large-scale operations. Consequently, automated solutions involving computer vision (CV) and machine learning (ML) techniques have gained traction in aquaculture, particularly in prawn farming.

Sensor-Based Counting Technologies: Early attempts at automation in aquaculture utilized sensor-based methods, such as infrared and resistivity counters, which detect fish or prawns as they pass through channels. These systems offer simplicity but are affected by environmental factors like water turbidity and overlapping fish or prawns, leading to inaccurate counts. Studies have shown that these methods underperform when applied in complex environments.

Computer Vision in Aquaculture: With the advent of computer vision technologies, aquaculture operations have shifted toward more accurate, non-invasive methods for monitoring fish and prawn populations. This study demonstrated the application of Fast R-CNN, a deep learning model, in underwater environments to detect fish with higher accuracy compared to traditional sensor methods. In prawn farming, computer vision systems have been adapted to estimate both density and individual growth rates by analysing visual data from camera feeds. Mask R-CNN and Cascade Mask R-CNN have been particularly successful in detecting and segmenting aquatic animals, enabling more accurate growth monitoring.

Machine Learning for Growth Estimation: Machine learning algorithms, lightweight models have been implemented to provide real-time monitoring solutions in aquaculture. These models analyse images to estimate prawn size and population density efficiently, making them suitable for small- and medium-scale farms. This study [4] emphasized the benefits of combining computer vision with machine learning to automate prawn size estimation and optimize feeding schedules.

Challenges in Automating Aquaculture: While computer vision and machine learning technologies have proven effective, there are challenges related to environmental noise, such as poor lighting and water clarity, which can affect the accuracy of these systems. Future research aims to improve the robustness of CV and ML models in these conditions, making them more reliable for use in aquaculture

settings. Additionally, the cost and complexity of these systems remain barriers to adoption for small- and medium-scale farmers.

Table 2.2.1: A comparison of different methods used for object counting in aquaculture

Counting methods		Advantages	Disadvantages	Application
Sensors based		Fast response, easy to implement	Required equipment to restrict the fish movement and damage to fish; prone to underestimated for overlapping fish	Integrated into the counting device for fry counting or ornamental fish count
Computer vision based	Image processing	Non-invasive, better accuracy with better algorithm architecture and optimization	Computing power of the hardware required, light attenuation underwater, unable to continuous counting	Population or abundance estimation of animal in underwater images by ROV
	Video analysis	Real-time, efficient		
Acoustic based	Hydroacoustic methods	Fast and efficient, not affected by water turbidity and light	Difficult to recognize small and overlapping fish.	Fish population estimation in waters such as lakes or rivers.
	Acoustic imaging	Able to obtain images close to the video images in dim murky water		

2.2.2 Strengths and Weakness

Automatic counting methods in aquaculture, such as sensor-based technologies, computer vision, and machine learning, offer substantial improvements over traditional manual approaches, particularly for prawn farming. Their strengths, higher accuracy, non-invasive monitoring, real-time capabilities, detailed insights, and adaptability—make them powerful tools for enhancing efficiency and sustainability. However, weaknesses like environmental sensitivity, high costs, technical complexity, data requirements, and computational demands highlight the need for ongoing research and development. Addressing these challenges could make these technologies more accessible and reliable, especially for small- and medium-scale farmers, revolutionizing prawn farming and aquaculture management.

With proposed solution the system simplifies operation with a pre-trained model and straightforward deployment on Raspberry Pi, reducing the need for specialized knowledge. YOLO11n's lightweight architecture runs efficiently on the Raspberry Pi (achieving 5 FPS with prawn images), minimizing computational demands.

2.3 Computer Vision Based Estimation of Shrimp Population Density and Size [4]

2.3.1 Review

Computer vision (CV) techniques have been applied extensively in the field of shrimp farming to automate the process of estimating shrimp population density and size. Shrimp is a significant aquaculture product, and continuous monitoring of shrimp population and growth is crucial for optimal farm management. Traditional methods of manual counting and measuring shrimp are labor-intensive and prone to errors. Therefore, automated methods using computer vision and deep learning techniques have been explored as efficient alternatives.

In this study, it implemented a CV-based system to monitor shrimp farms. The system utilizes **U-Net segmentation** combined with **marker-controlled watershed segmentation** and thresholding to count shrimps and estimate their lengths. The **U-Net model**, originally designed for biomedical image segmentation, is fully convolutional and able to assign class labels to each pixel in an image, making it highly effective in segmenting shrimp from their surroundings. The model consists of 23 convolutional layers organized into contracting and expansive paths, with the ability to function well even with small datasets.

The segmentation is followed by **marker-controlled watershed segmentation** to handle cases where shrimps are touching or overlapping each other. This technique allows for better separation of overlapping objects by computing watershed lines along object boundaries. The accuracy of this approach was tested on images of shrimp from a laboratory environment, yielding a **mean absolute error of 0.093** in shrimp counting and a **root mean square (RMS) error of 0.293 cm** in shrimp length estimation when the shrimps were separately located.

However, the study also noted limitations in the method's ability to accurately count shrimps when they are overlapping or touching, where the mean absolute error increased to 0.298. Despite these challenges, the system proved robust in its ability to segment even small parts of shrimp, such as tails.

In addition to the CV and segmentation techniques, the system includes a **web-based monitoring platform** using **Heroku**, which allows shrimp farmers to easily access shrimp population density and length data through a web interface. This cloud-based solution is particularly useful for non-experts, providing a user-friendly interface for real-time farm monitoring.

2.3.2 Strengths and Weakness

The computer vision-based system for shrimp population and size estimation revolutionizes aquaculture by automating labor-intensive tasks and delivering high accuracy in controlled environments, with counting errors as low as 0.093 and size estimation errors around 0.293 cm. It employs advanced segmentation techniques, such as U-Net and marker-controlled watershed, to tackle complex scenarios and provides an intuitive web interface for seamless data access. However, its performance drops significantly when shrimp overlap, with counting errors rising to 0.298, and it struggles in murky or poorly lit conditions due to its dependence on high-quality imaging. The system also demands substantial computational resources and lacks extensive real-world field validation, making its high implementation and scalability costs a potential barrier, particularly for smaller farms, limiting widespread adoption to well-resourced operations.

With proposed solution it provides Cost-Effective and Scalable. At $60-$80 per unit, the Raspberry Pi setup is affordable and easily replicable across multiple ponds, enhancing scalability for small farmers. Besides that, it provides flexible imaging. The PiCamera (Night Vision) captures usable images in low-light and turbid conditions, reducing dependency on pristine imaging environments.

2.4 AI-image processing and image recognition for intelligent prawn farming [5]

2.4.1 Review

The application of **Artificial Intelligence (AI)** and **image processing** in aquaculture has gained significant traction over the past few years, especially for tasks such as prawn growth stage detection and population estimation. The need for automation in prawn farming arises due to the labor-intensive and error-prone nature of manual methods. These technologies aim to enhance farming efficiency, ensure sustainability, and optimize resource utilization in prawn cultivation.

Convolutional Neural Networks (CNNs) have been widely adopted in aquaculture for image recognition and classification tasks. CNNs are particularly suitable for processing spatial data such as images, making them an ideal choice for prawn detection and growth monitoring. In prawn farming, CNNs can be used to classify prawns into different growth stages—Juvenile, Premature, and Mature—based

on their biometric traits. For example, **ShrimpNet**, an architecture developed for shrimp detection, has shown significant improvements in classification accuracy.

The **You Only Look Once (YOLO)** model has emerged as a popular choice for real-time object detection due to its high processing speed and accuracy. In prawn farming, YOLO can rapidly detect and classify prawns in different environments, such as underwater or in controlled ponds. YOLO's ability to handle high-speed processing makes it suitable for applications where large volumes of image data need to be processed in real-time.

Faster Region-based Convolutional Neural Network (R-CNN) is another deep learning model used for object detection in prawn farming. By leveraging the **ResNet** backbone, Faster R-CNN improves the accuracy of prawn detection by focusing on region proposals and bounding boxes for prawn images. The combination of ResNet with Faster R-CNN allows for deeper networks that reduce the risk of overfitting, a common challenge in deep learning applications.

Additionally, models like **SSD (Single Shot Detector)** and **CenterNet Hourglass 104** have been explored for prawn detection and classification. SSD is known for its balance between speed and accuracy, while CenterNet excels in keypoint detection and object localization tasks. These models provide alternatives depending on the specific requirements of the farming environment and computational resources available.

While these models have shown promising results, challenges remain in applying these techniques to prawn farming. For instance, the lack of available datasets for prawn growth stages necessitates the creation of custom datasets. In this project, the researcher manually collected and labeled prawn images to train the deep learning models. Moreover, the variation in environmental conditions, such as water clarity and lighting, can affect the accuracy of these models.

The success of AI-based systems in prawn farming hinges on selecting the right model and optimizing it for the given conditions. In this research, a combination of CNNs, YOLO, Faster R-CNN, and CenterNet is evaluated to determine the most suitable model for prawn growth stage detection and population estimation. The performance of these models is measured using metrics such as **mean Average Precision (mAP)**, **Average Recall (AR)**, and **loss**, ensuring that the model provides accurate and reliable results under various conditions.

2.4.2 Strengths and Weakness

AI-based image processing and recognition systems, leveraging models like CNNs, YOLO, and Faster R-CNN, offer transformative strengths for intelligent prawn farming. They automate tasks such as prawn counting and growth stage classification, replacing labor-intensive manual methods with high accuracy—often achieving over 90% precision in controlled settings. Real-time processing, especially with YOLO, enables continuous monitoring for timely interventions, while adaptability across models allows customization to diverse farming environments. These systems also optimize resource management by providing precise data for feeding and harvesting, reducing waste and costs. However, significant weaknesses persist. The lack of standardized datasets forces farms to invest heavily in custom data collection, delaying deployment. Environmental variability, such as water clarity and lighting, can degrade model performance, requiring costly preprocessing. Computational demands for training and running models are high, often necessitating expensive hardware or cloud services. Additionally, the risk of overfitting limits model generalizability, and the technical expertise needed for setup and maintenance creates barriers, particularly for small-scale farmers. Addressing these challenges is essential to unlock the full potential of AI in prawn farming.

With proposed solution, automated Dataset Creation, the Cron Job feature collects prawn images continuously, building a custom dataset tailored to my farm's conditions, bypassing the need for standardized datasets. Other than that, simplified Operation. The system's design prioritizes ease of use, with automated processes and a web-based interface (planned), minimizing technical expertise requirements.

2.5 Fish Species Detection and Recognition Using MobileNet v2 Architecture: A Transfer Learning Approach [8]

2.5.1 Review

Fish species detection and recognition is a crucial aspect of biodiversity conservation and fisheries management, where traditional methods of manual identification are often inefficient and error prone. Automated detection systems, powered by deep learning models like MobileNetV2, have emerged as solutions to these challenges. MobileNetV2, designed for resource-constrained devices, is

particularly suited for applications in marine environments due to its efficiency and ability to perform well on mobile or embedded systems.

MobileNetV2's architecture is built on inverted residuals and linear bottlenecks, which help reduce computational costs while retaining high accuracy. This makes it ideal for fish species classification, where real-time processing and minimal hardware requirements are essential. The model has been effectively used in various studies involving fish species detection, often combined with transfer learning techniques. By leveraging pre-trained weights from large datasets like ImageNet, MobileNetV2 adapts well to domain-specific tasks, achieving high accuracy in classifying fish species based on subtle visual differences such as color, shape, and texture.

Performance metrics like accuracy, precision, recall, and F1-score demonstrate MobileNetV2's robustness in detecting fish species with minimal errors. Studies have shown that the model can achieve near-perfect accuracy in fish species classification, making it a valuable tool for ecological monitoring. However, challenges such as variability in underwater environments and class imbalance still pose difficulties. Some studies have addressed these issues by employing class-aware loss functions to improve model performance in datasets with underrepresented species.

Overall, MobileNetV2 has proven to be a powerful tool for fish species detection, offering high accuracy and computational efficiency. Its application extends to ecological research, fisheries management, and conservation efforts, where accurate species identification is vital. Despite some challenges, MobileNetV2's flexibility and adaptability through transfer learning make it a promising solution for real-time monitoring and automated species recognition in aquatic environments.

2.5.2 Strengths and Weakness

MobileNetV2 is highly effective for fish species detection due to its computational efficiency, making it ideal for resource-constrained devices in marine environments. Its high accuracy, achieved through transfer learning, enables it to distinguish subtle visual differences in fish species, while real-time processing supports critical applications like ecological monitoring and fisheries management. The model's robust performance metrics minimize errors, and its flexibility allows adaptation to various underwater conditions. However, it is sensitive to environmental variability, such as changes in lighting or water clarity, which can reduce accuracy. Class imbalance in datasets can

lead to biased predictions, and its reliance on pre-trained data quality risks performance issues if underwater imagery is not well-aligned. In complex scenarios, like cluttered environments, misclassifications may occur, and hardware constraints in extreme conditions can limit deployment in remote or harsh marine settings. While MobileNetV2 offers significant advantages in efficiency, accuracy, and adaptability, these environmental and technical challenges require careful consideration for optimal use in aquatic environments.

With proposed solution, it achieved optimized hardware. The Raspberry Pi 5, paired with YOLO11n and NCNN optimization, performs reliably in farm conditions, overcoming hardware constraints. Environmental Robustness. YOLO11n outperforms MobileNet v2 in diverse conditions, as shown in my training with varied prawn pond images, addressing sensitivity issues.

CHAPTER 3 Proposed Method/Approach

The processes of the project were categorized into different phases in the development, which were project pre-development, data pre-processing, model training architecture building and data training, and prediction on test dataset.

3.1 System Architecture Diagram

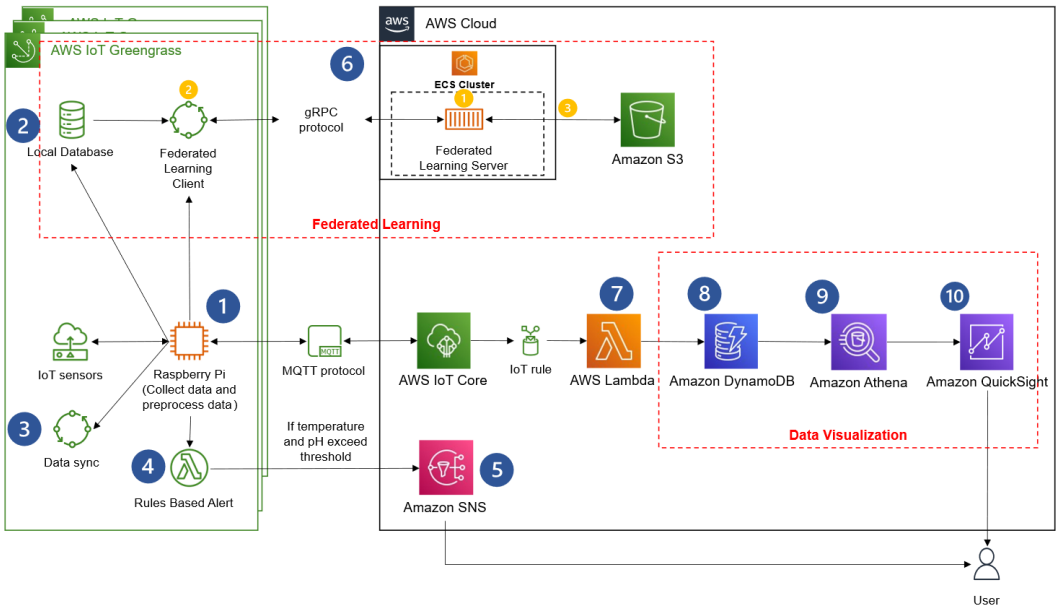


Figure 3.1.1 System Architecture Diagram of This Project

1. The raspberry pi kick start the sensors reading, capture and process data, pH value and temperature are collected every 10 seconds regardless the network connection
2. The sensors data are store in local storage, a relational database regardless the network connection
3. **If network present:** periodically check the temporary storage to perform data syncing with DynamoDB
4. While the sensors reading data, a rules-based condition will be evaluated
5. If temperature or pH reading exceed acceptable threshold, an alert will be sound via email to user
6. Federated Learning
   1. Server initialize a global model weight and upload to S3
   2. Client download global model weight and perform local training using local data
   3. Client upload trained local weight to S3
   4. Server perform model aggregation using weights from clients and upload aggregated weight as global model back to S3
7. Data process by Raspberry Pi will send into AWS Lambda for estimation of growth and density
8. **If network connection present:** the sensor data will be store into DynamoDB located in cloud as a backup source, else data will be duplicate into a temporary storage
9. Athena query data store in DynamoDB to serve as a data source for visualization
10. A near real time interactive dashboard visualize data for farm monitoring

Figure 3.1.2 Workflow of this project.

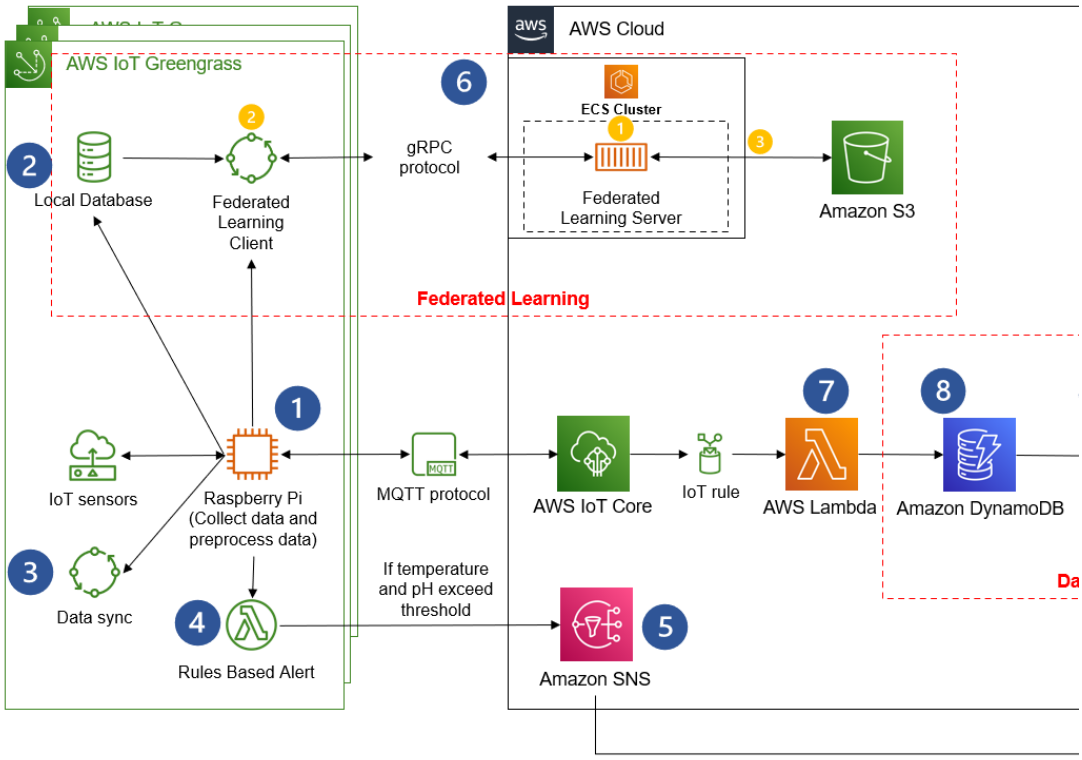


Figure 3.1.3: Main part of Computer Vision and Automated Density and Growth Estimation Module.

The overall system developed consists of three parts: hardware, software and Cloud. The hardware used in this system will be raspberry pi and the IoT sensor. Raspberry pi kick starts the PiCamera to collecting data and process data by pre-trained YOLO11n model, pH value and temperature are collected regardless the network connection. The sensors data are store in local storage, a relational database regardless the network connection. Data process by Raspberry Pi will be sent into AWS Lambda for estimation of growth and density. This report will only be focusing on Computer Vision, density and growth estimation.

3.2 Flowchart

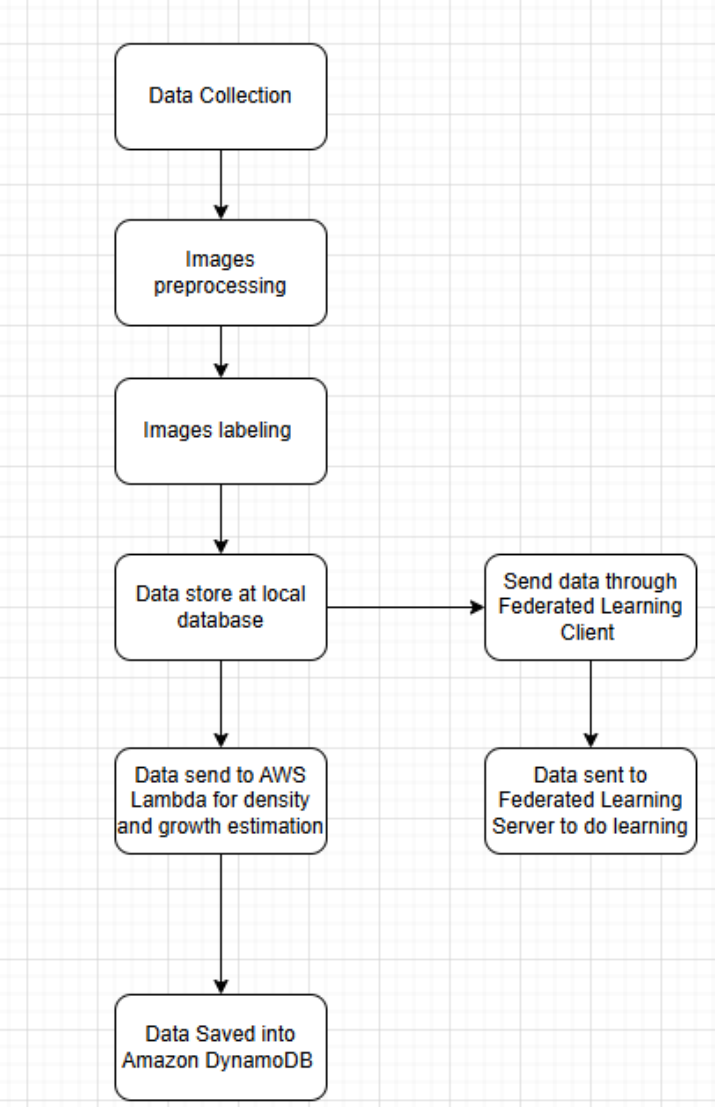


Figure 3.2.1: Flowchart of the Automated Density and Growth Estimation Module

The workflow for the Utilising Computer Vision Techniques for Automated Density and Growth Estimation in Precision Aquaculture Systems for Prawn Cultivation is structured into several key stages, each contributing to the efficiency and accuracy of the overall process. The system begins with data collection, where high-resolution cameras are installed in the prawn ponds to capture continuous images of the prawns in their natural environment. In some cases, sensors may also be deployed to monitor environmental factors such as water quality, temperature, and pH levels. Once the images are collected, the data annotation phase involves manually labelling each prawn within the images by drawing bounding boxes around them. The annotated dataset is crucial for training the machine learning model, where prawns are detected and localized accurately.

Following data annotation, the model development and training stage takes place. In this step, YOLO11n, is used for detection. The model is trained using the annotated data and fine-tuned to ensure accuracy and prevent issues like overfitting. After the model is properly trained, it is deployed on the Raspberry Pi for real-time monitoring. The system continuously processes live images from the cameras, allowing it to detect and count prawns in real-time. Where all the data will be stored at the local database. Once the Raspberry Pi is connected to the Internet, it will send the data through Federated Learning Client. Data will be sent to the Federated Learning Server to do learning.

The next stage, density and growth estimation, is where the system uses the pre-trained model to count the number of prawns and collect the data. The data will send to AWS Lambda to do with density and growth estimation within a given pond area. In AWS Lambda, there be a YOLO model that will only detect the shrimp body only, and it will further predict the growth stages of the prawns by analyzing their size. This data is used to optimize operations such as feeding schedules and harvest planning.

3.3 Growth Stage Classification

Growth stage classification assigns each prawn to a category (Hatchling, Juvenile, Adult) based on total length, estimated from YOLOv11n's bounding box width in top-down images. Total length includes the full body length, approximated by:

1. Total Length Estimation:

```python
def calculate_shrimp_length(bbox_width_px, bbox_height_px, frame_width, frame_height):
    """
    Calculate shrimp length in cm based on bounding box dimensions

    Args:
        bbox_width_px: Bounding box width in pixels
        bbox_height_px: Bounding box height in pixels
        frame_width: Frame width in pixels
        frame_height: Frame height in pixels

    Returns:
        Length in cm (using the longer dimension of the bounding box)
    """
    # Calculate pixels per cm for both dimensions
    pixels_per_cm_width = frame_width / BOX_WIDTH_CM
    pixels_per_cm_height = frame_height / BOX_HEIGHT_CM

    # Convert bounding box dimensions to cm
    width_cm = bbox_width_px / pixels_per_cm_width
    height_cm = bbox_height_px / pixels_per_cm_height

    # Use the longer dimension as the shrimp length
    length_cm = max(width_cm, height_cm)

    return length_cm
```

Figure 3.3.1: Code for Measure the Length of Shrimp

The length calculation system uses computer vision and calibration to measure real-world shrimp dimensions from camera images. When the YOLO model detects a shrimp, it draws a bounding box around the object in pixel coordinates. The system then converts these pixel measurements to real-world centimetres using a calibration process based on a predefined measurement box. First, it calculates how many pixels represent one centimetre by dividing the total frame dimensions by your box dimensions (e.g., 640 ÷ 40 cm = 16 pixels per cm horizontally). Then, it converts the bounding box width and height from pixels to centimetres using these conversion ratios. Finally, it takes the longer dimension of the bounding box as the shrimp's length, since shrimp are typically longer than they are wide. This approach ensures accurate measurements regardless of camera resolution or distance, as long as the camera is viewing the calibrated 40cm × 30cm area.

2. Stage Classification:

Table 3.3.1: Growth Phase [9], [10]

Growth Phase	Approx. Weight Range (g)
Hatchling	<1g
Juvenile	1-50g

Adult	>50g

The system can then use measured length to calculate weight using the Weight-length relationship W = 0.018 × L^3.06, providing comprehensive size analysis [11].

3. **Average Total length and Weight:**

```
if shrimp_count > 0:
    avg_length = total_length / shrimp_count
    min_length = min(shrimp_lengths)
    max_length = max(shrimp_lengths)
    avg_weight = total_weight / shrimp_count
    min_weight = min(shrimp_weights)
    max_weight = max(shrimp_weights)
```

Figure 3.3.2 Equation for Average Total Length and Weight

4. **Density:**

```
def calculate_density_category(shrimp_count, area_sqm):
    """
    Calculate shrimp density category based on count per square meter

    Args:
        shrimp_count: Number of shrimp detected
        area_sqm: Area in square meters

    Returns:
        Density category string
    """
    if area_sqm <= 0:
        return "Unknown"

    density_per_sqm = shrimp_count / area_sqm

    if density_per_sqm < 5:
        return "Low Density (<5/m²)"
    elif density_per_sqm <= 15:
        return "Moderate Density (5-15/m²)"
    else:
        return "High Density (>15/m²)"
```

Figure 3.3.3 Code for Density

Table 3.3.2 Shrimp Density Category and Definition [12]

Density Category	**Definition**
Low density	1-4 shrimp/m^2

Moderate density	5-15 shrimp/m^2
High density	>15 shrimp/m^2

5. YOLO models detect shrimp body part:

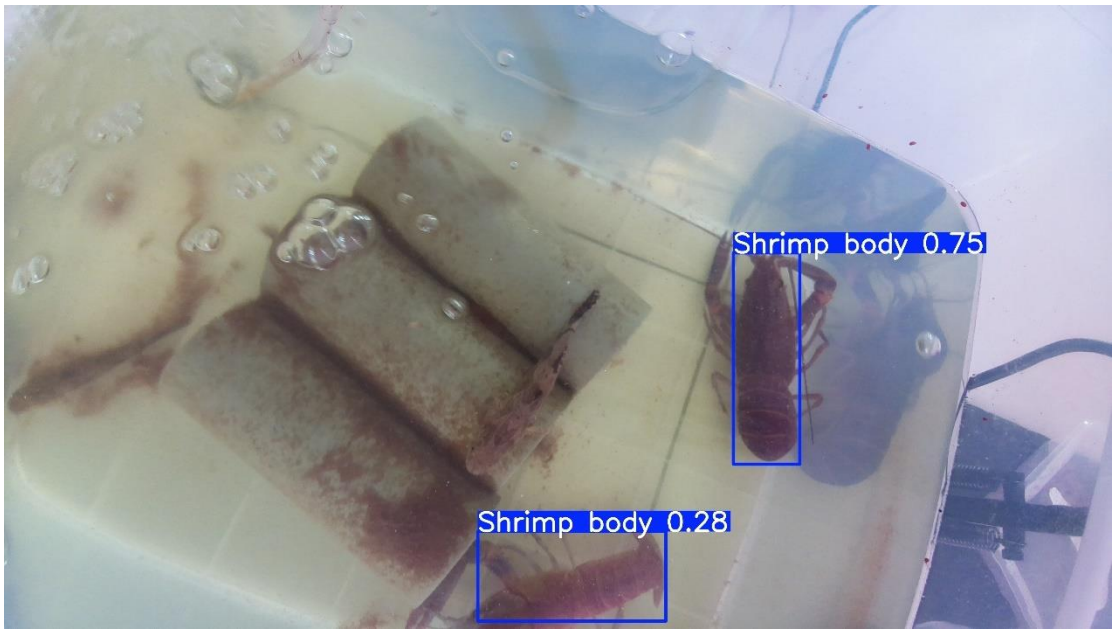


Figure 3.3.4: Detection Result of Shrimp body

CHAPTER 4 Preliminary Work

Preliminary Research is research on a topic that helps you get a better understanding on what types of sources are available and what is being said about a topic. This type of research helps solidify a topic by broadening or narrowing it down. This research can also help you when choosing Search Terms.

4.1 Data Sources

4.1.1 Data Collection

Prawn images were collected from a controlled prawn pond environment simulating the conditions of a small- to medium-scale farm. The PiCamera attached to the Raspberry Pi 5 was used to capture images of Cherax quadricarinatus prawns under various conditions, including different lighting (daylight and artificial light) and water clarity levels (clear and slightly turbid). Approximately 2000 images were collected initially, with plans to expand the dataset during later phases. Images were captured at different times of the day to account for lighting variations. Prawns of varying sizes (juvenile, premature, and mature) were included to support growth stage classification. The images were stored in JPG format.

Figure 4.1.1: Prawn Image Captured

Figure 4.1.2 Prawn Image Captured in Dark Condition

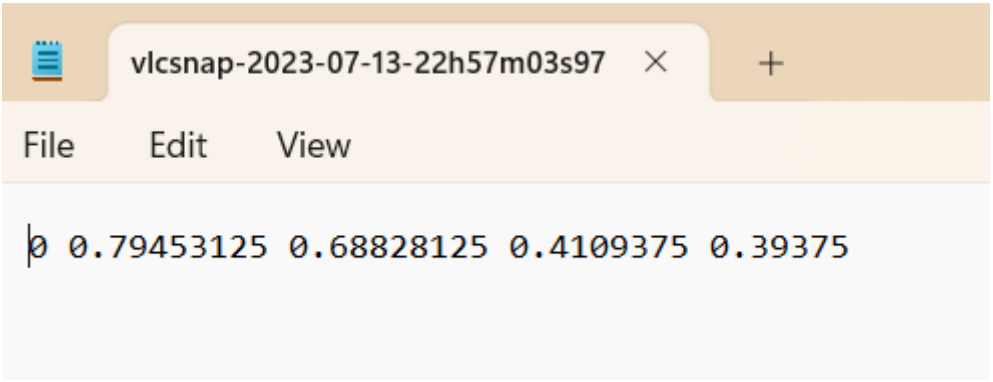


Figure 4.1.3 Class and Coordinates of Bounding Box

4.1.2 Images Annotation

Label Studio was used to annotate the collected images. The annotation process involved:

1. **Bounding Box Creation**: For each prawn in an image, a bounding box was drawn to enclose the entire prawn, including its body and claws. The bounding box coordinates were saved in YOLO format, which includes the class label (e.g., "prawn"), normalized center coordinates (x, y), and normalized width and height.

2. **Class Labeling**: Initially, a single class ("prawn") was used for detection. In future iterations, additional classes for growth stages (juvenile, premature, mature) may be introduced based on size estimation.

The dataset was split into training (90%), and test (10%) sets to support model development.

4.2 Device Comparison for Cost-Effectiveness and Scalability

This section compares the Raspberry Pi 5, used in the project for automated prawn density and growth estimation, with other devices to assess their suitability for small- and medium-scale prawn farmers. The comparison focuses on cost, performance, ease of use, and scalability, aligning with Objective 3: designing a cost-effective and scalable solution.

Devices Compared

We evaluated the following devices, selected for their potential in computer vision tasks for aquaculture:

- **Raspberry Pi 5**: The baseline device, with 4x Cortex-A76 CPU @ 2.4 GHz, 4/8 GB RAM, VideoCore VII GPU, and dual MIPI-CSI camera ports, priced at $60-$80.
- **NVIDIA Jetson Nano**: 4x Cortex-A57 CPU @ 1.43 GHz, 4 GB RAM, 128-core Maxwell GPU, priced at $99.
- **Rock Pi 4 B**: 2x Cortex-A72 @ 1.8 GHz + 4x Cortex-A53 @ 1.4 GHz, 1/2/4 GB RAM, 4x Mali-860 GPU, priced at €68-€100.
- **Odroid XU4**: 4x Cortex-A15 @ 2.0 GHz + 4x Cortex-A7 @ 1.4 GHz, 2 GB RAM, 6x Mali-628 GPU, priced at €75.

Comparison Criteria

- **Cost**: Essential for affordability, given the target audience of small- and medium-scale farmers.
- **Performance**: Ability to run YOLO11n in real-time for prawn detection.

- **Ease of Use**: Community support and availability of resources for setup and maintenance.
- **Suitability for Aquaculture**: Integration with cameras and deployment in farm environments.
- **Scalability**: Ease of deploying multiple units for larger farms.

Detailed Comparison

Table 4.2.1: Comparison of Devices [13], [14], [15], [16]

Device	**Cost (USD/EUR)**	**CPU/GPU**	**RAM (GB)**	**Camera Interface**	**Performance for YOLO11n**	**Ease of Use**	**Suitability for Farmers**
Raspberry Pi 5	$60-$80	4x Cortex-A76 @ 2.4 GHz, VideoCore VII	4/8	x2 MIPI-CSI	High, real-time capable	Excellent, large community	High, affordable, easy integration
NVIDIA Jetson Nano	$99	4x Cortex-A57 @ 1.43 GHz, 128-core Maxwell	4	Supported, details vary	High, GPU acceleration	Good, AI-focused community	Moderate, higher cost, better AI
Rock Pi 4 B	€68-€100	2x Cortex-A72 + 4x Cortex-A53, 4x Mali-860	1/2/4	Not specified	Moderate, may need optimization	Fair, smaller community	Moderate, cost varies, camera unclear
Odroid XU4	€75	4x Cortex-A15 + 4x Cortex-	2	None (USB/Ethernet)	Moderate, no camera slot	Fair, limited support	Low, lacks camera integration

Device	Cost (USD/EUR)	CPU/GPU	RAM (GB)	Camera Interface	Performance for YOLO11n	Ease of Use	Suitability for Farmers
		A7, 6x Mali-628					

Analysis

- **Cost**: Raspberry Pi 5 is the most affordable at $60 for 4 GB RAM, making it accessible for small-scale farmers. Jetson Nano at $99 is more expensive, while Rock Pi 4 B and Odroid XU4 are similarly priced but may require additional costs for camera setups.
- **Performance**: Raspberry Pi 5's CPU is sufficient for YOLO11n, as demonstrated in the project. Jetson Nano offers GPU acceleration, which could handle more complex models, but its CPU is less powerful. Rock Pi 4 B and Odroid XU4 may struggle with real-time performance without optimizations.
- **Ease of Use**: Raspberry Pi's large community ensures extensive resources, ideal for farmers with limited technical expertise. Jetson Nano has good AI support but a smaller community. Rock Pi 4 B and Odroid XU4 have less support, potentially complicating deployment.
- **Suitability for Aquaculture**: Raspberry Pi 5's dual camera ports integrate seamlessly with PiCamera (Night Vision), crucial for prawn monitoring. Jetson Nano supports cameras but at a higher cost. Rock Pi 4 B's camera interface is unclear, and Odroid XU4 lacks a built-in slot, adding complexity.
- **Scalability**: Raspberry Pi 5 is widely available and easy to deploy in multiple units, aligning with scalability needs. Jetson Nano is scalable but costlier per unit. Other devices may face availability or compatibility issues.

Conclusion

Research suggests Raspberry Pi 5 is the best fit for Objective 3, offering a cost-effective ($60-$80), scalable, and user-friendly solution for prawn farmers. While NVIDIA Jetson Nano provides better AI capabilities, its higher cost ($99) may not be justified for current needs. Other devices like Rock Pi 4 B and Odroid XU4 are less suitable due to camera integration challenges and limited community support. Thus, Raspberry Pi 5 aligns with the project's goal of providing an affordable, practical solution for small- and medium-scale farmer.

CHAPTER 5 System Implementation

5.1 Hardware Setup

The hardware involved in this project is computer, Raspberry Pi 5, PiCamera (Night Vision). A computer issued for the process of training for YOLO11n model. A Raspberry Pi 5 and PiCamera (Night Vision) is used for testing model and collecting data at the prawn farm.

Table 5.1.1 Specifications of laptop

Description	Specifications
Model	Asus TUF F15
Processor	Intel Core i5-10300H CPU @ 2.50GHz 2.50 GHz
Operating System	Windows 10
Graphic	NVIDIA GeForce GTX 1650
Memory	8GB RAM
Storage	512GB SSD

Table 5.1.2 Specifications of Raspberry Pi

Description	Specifications
Model	Raspberry Pi 5
Processor	Broadcom BCM2712 2.4GHz quad-core 64-bit Arm Cortex-A76 CPU
Graphic	VideoCore VII GPU
Memory	SDRAM 4GB
Storage	SanDisk Ultra 64GB

5.2 Software Setup

Before starting to train the YOLO11n model, there are five software needed to be installed and downloaded in the laptop:

1. Visual Studio Code (VS Code): Used as the primary code editor for writing and debugging scripts related to data preprocessing, model training, and deployment. VS Code's support for Python extensions and integrated terminal facilitated efficient development workflows.

2. Label Studio: Employed for annotating prawn images to create a labeled dataset for training the YOLO11n model. Label Studio allowed for the creation of bounding boxes around prawns in images, which were exported in YOLO format for compatibility with the training pipeline.

3. Google Colab: Utilized for training the YOLO11n model due to its access to free GPU resources, which significantly reduced training time compared to local hardware. Colab notebooks were used to execute Python scripts for model training, validation, and testing.

4. WinSCP: WinSCP (Windows Secure Copy) is a free, open-source file transfer client for Windows that supports protocols like SFTP, SCP, FTP, WebDAV, and Amazon S3. It's widely used for securely transferring files between a local computer (e.g., a Windows PC) and a remote device (e.g., a Raspberry Pi).

5. RealVNC Viewer: **RealVNC Viewer** is a remote desktop software application that allows you to **view and control another computer remotely** over a network using the VNC (Virtual Network Computing) protocol. In this project, it will be used to connect to the Raspberry Pi.

5.3 YOLO11n Model training

5.3.1 Model selection

The YOLO11n model was selected for its lightweight architecture, making it suitable for deployment on resource-constrained devices like the Raspberry Pi 5. The selection of YOLO11n was driven by its optimal balance of speed, accuracy, and efficiency, which are critical for real-time monitoring on resource-constrained devices like the Raspberry Pi. A comparative analysis of YOLO models (YOLO11n, YOLOv8n, and YOLOv5) was conducted using a benchmark dataset of coins (pennies, nickels, dimes,

and quarters), as shown in Figure 5.3.1. The results demonstrated YOLO11n's superior performance in detecting objects accurately while maintaining high speed.

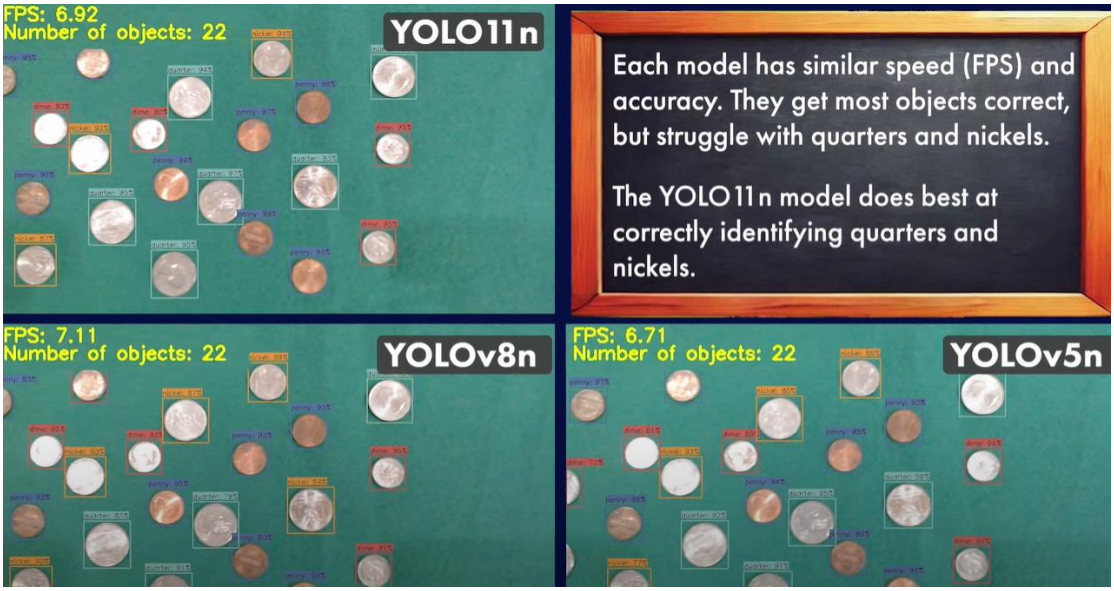


Figure 5.3.1: Benchmark Results of YOLO Models [17]

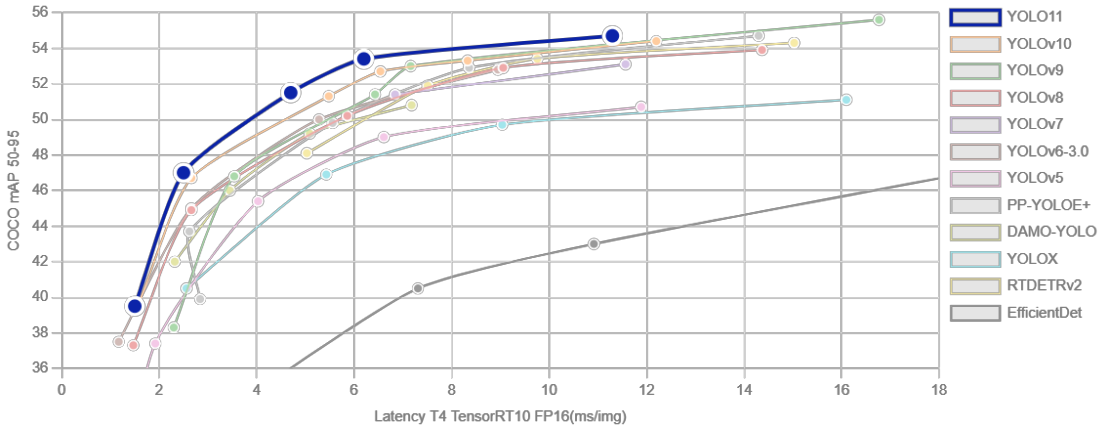


Figure 5.3.2: Benchmark Results from Ultralytics [18]

The benchmark results align with the project's requirements for prawn detection, where accurate identification of prawns (analogous to quarters and nickels in the coin dataset) is crucial for density and growth estimation. YOLO11n's ability to detect objects with higher accuracy, particularly in distinguishing similar-sized objects, made it the optimal choice for this application. Additionally, its FPS of 6.89 on the benchmark dataset supports its feasibility for real-time inference on the Raspberry Pi, where an FPS of 5 was achieved with prawn images.

5.3.2 Model training process and deployment

Model training in Google Colab. All the training and deployment will be done in Google Colab.

1. Upload data in zip file format

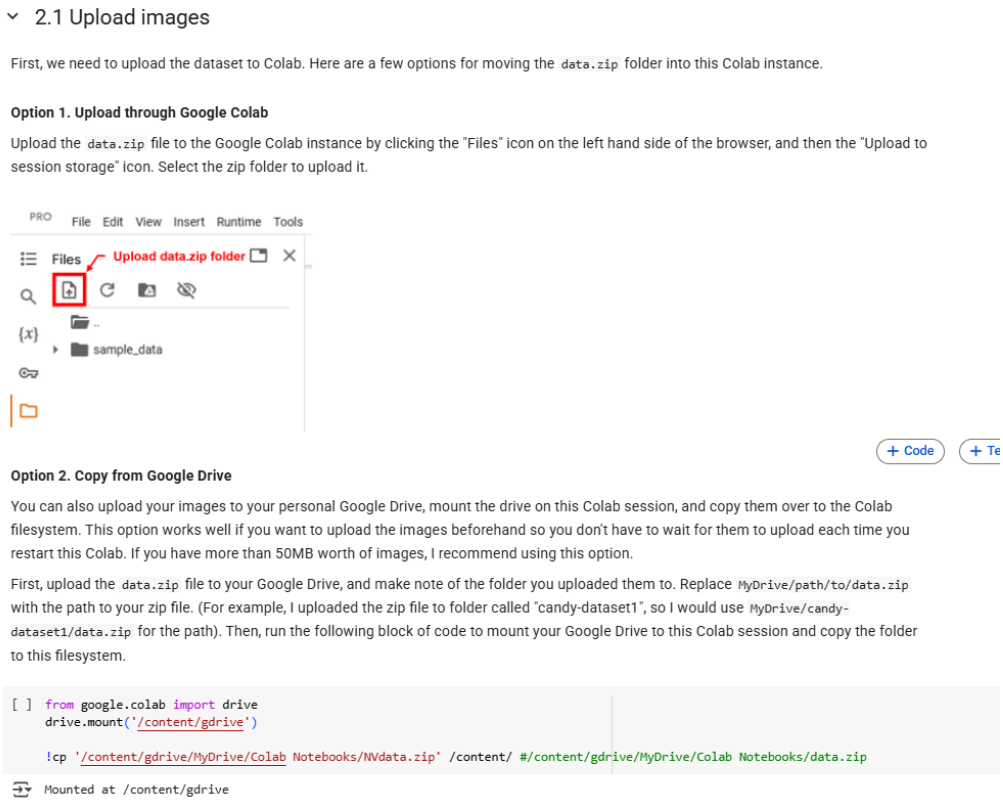


Figure 5.3.3 Ways to Upload Datasets to Google Colab

2. Unzip and split the images into train and validation folders

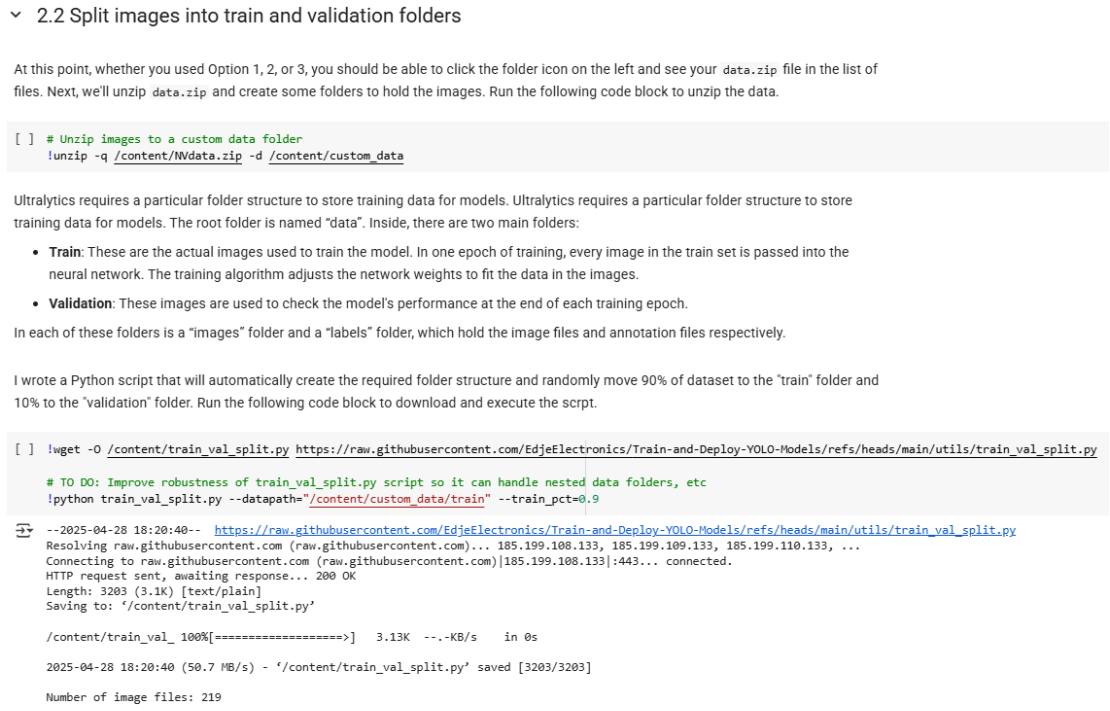


Figure 5.3.4 Code Snippet to Unzip and Split the Images into Train and Validated Folders

3. Install Ultralytics

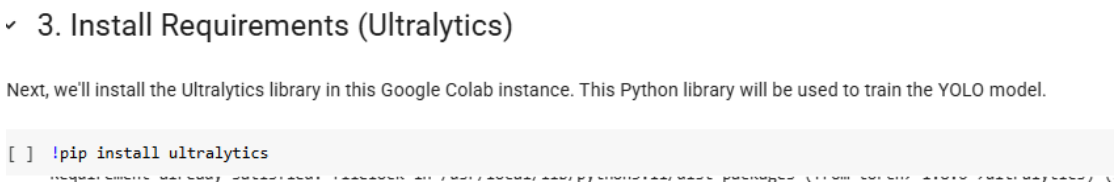


Figure 5.3.5 Code Snippet to Install Ultralytics Library in the Google Colab Instance

4. Configure Training

There's one last step before we can run training: we need to create the Ultralytics training configuration YAML file. This file specifies the location of your train and validation data, and it also defines the model's classes. An example configuration file model is available here.

Run the code block below to automatically generate a `data.yaml` configuration file. Make sure you have a labelmap file located at `custom_data/classes.txt`. If you used Label Studio or one of my pre-made datasets, it should already be present. If you assembled the dataset another way, you may have to manually create the `classes.txt` file (see here for an example of how it's formatted).

```
[ ]  # Python function to automatically create data.yaml config file
     # 1. Reads "classes.txt" file to get list of class names
     # 2. Creates data dictionary with correct paths to folders, number of classes, and names of classes
     # 3. Writes data in YAML format to data.yaml

     import yaml
     import os

     def create_data_yaml(path_to_classes_txt, path_to_data_yaml):

       # Read class.txt to get class names
       if not os.path.exists(path_to_classes_txt):
         print(f'classes.txt file not found! Please create a classes.txt labelmap and move it to {path_to_classes_txt}')
         return
       with open(path_to_classes_txt, 'r') as f:
         classes = []
         for line in f.readlines():
           if len(line.strip()) == 0: continue
           classes.append(line.strip())
       number_of_classes = len(classes)

       # Create data dictionary
       data = {
           'path': '/content/data',
           'train': 'train/images',
           'val': 'validation/images',
           'nc': number_of_classes,
           'names': classes
       }

       # Write data to YAML file
       with open(path_to_data_yaml, 'w') as f:
         yaml.dump(data, f, sort_keys=False)
       print(f'Created config file at {path_to_data_yaml}')

       return

     # Define path to classes.txt and run function
     path_to_classes_txt = '/content/custom_data/classes.txt'
     path_to_data_yaml = '/content/data.yaml'

     create_data_yaml(path_to_classes_txt, path_to_data_yaml)
```

Figure 5.3.6: Creation of YAML File

Figure 4.3.5 shows the creation of a configuration file in YAML format. This file should include the dataset paths for training and validation, as well as the number of classes and their corresponding names.

5. Train Model

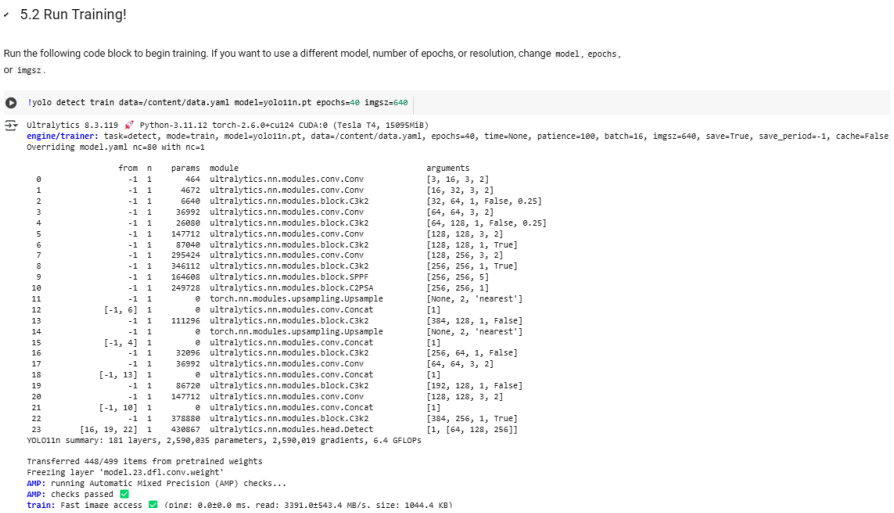


Figure 5.3.7 Log Information of Model Training

The dataset contains over 200 images. Training the model for 40 epochs with an input resolution of 640×640 is suitable to ensure efficient performance and faster processing.

6. Test Model

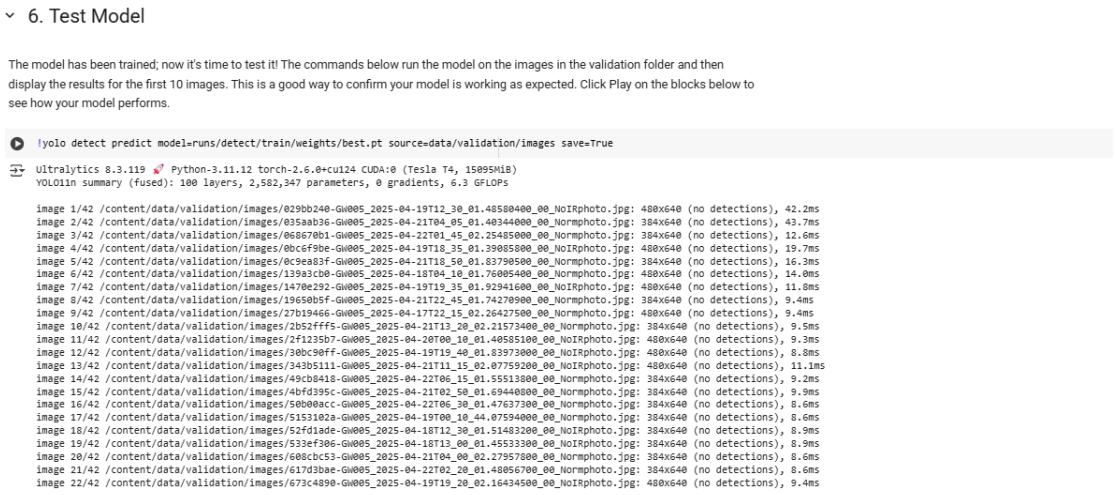


Figure 5.3.8: Logs Information of Testing the Model

```
[ ] import glob
    from IPython.display import Image, display
    for image_path in glob.glob(f'/content/runs/detect/predict/*.jpg')[:10]:
      display(Image(filename=image_path, height=400))
      print('\n')
```

Figure 5.3.9: Code Snippet to Load and Display Predicted Images

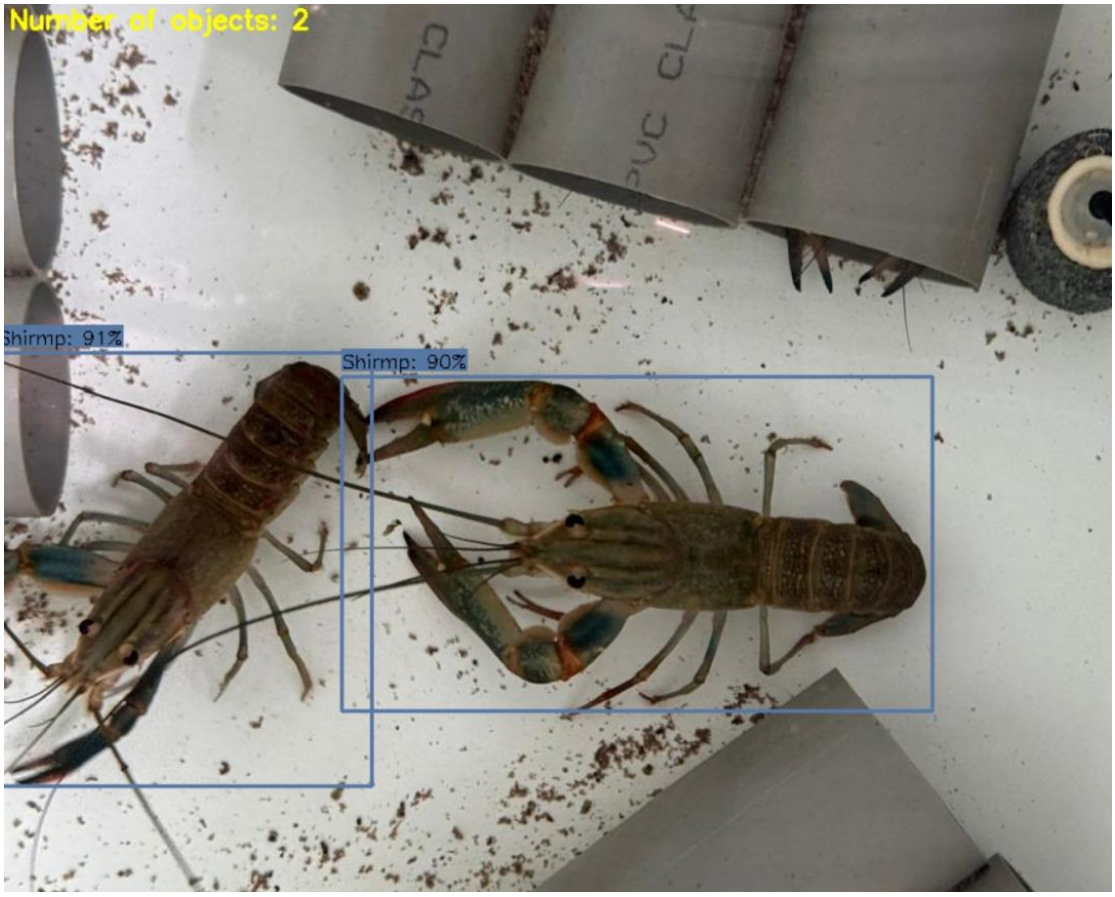


Figure 5.3.10 Object Detection Results of Model with Bounding Boxes Drawn

5.3.3 Model deployment

1. Download the Trained Model

7.1 Download YOLO Model

First, zip and download the trained model by running the code blocks below.

The code creates a folder named `my_model`, moves the model weights into it, and renames them from `best.pt` to `my_model.pt`. It also adds the training results in case you want to reference them later. It then zips the folder as `my_model.zip`.

```
[ ]  # Create "my_model" folder to store model weights and train results
     !mkdir /content/my_model
     !cp /content/runs/detect/train/weights/best.pt /content/my_model/my_model.pt
     !cp -r /content/runs/detect/train /content/my_model

     # zip into "my_model.zip"
     %cd my_model
     !zip /content/my_model.zip my_model.pt
     !zip -r /content/my_model.zip train
     %cd /content
```

Figure 5.3.11: Code Snippet to Zip and Download the Trained Model

2. Deploy on Raspberry Pi

```
evan@raspberrypi:~/yolo $ python3 -m venv --system-site-packages venv
evan@raspberrypi:~/yolo $
evan@raspberrypi:~/yolo $ source venv/bin/activate
(venv) evan@raspberrypi:~/yolo $
```

Figure 5.3.12: Command to Create a New Virtual Environment

This command creates a new virtual environment named venv in the ~/yolo directory, allowing access to system-wide packages while providing an isolated space for project-specific dependencies. And the second command use to confirms that the virtual environment has been successfully activated, and the terminal session is now using the isolated Python environment for subsequent commands.

3. **Install Ultralytics and ncnn**

```
(venv) evan@raspberrypi:~/yolo $ pip install ultralytics ncnn
```

Figure 5.3.13: Command to Install Ultralytics and ncnn Packages

The command pip install ultralytics ncnn installs two critical packages in the virtual environment on the Raspberry Pi:

- **Ultralytics**: Provides the YOLO11n model and associated tools for object detection, which is central to your project's goal of automated prawn density and growth estimation.
- **NCNN**: Enhances the deployment of the YOLO11n model by optimizing neural network inference for the Raspberry Pi, ensuring efficient real-time performance.

This step is a key part of setting up the Raspberry Pi for running the YOLO11n model, aligning with the project's objective of creating a cost-effective, real-time monitoring solution for prawn farming.

4. **WinSCP to transfer model and python script**

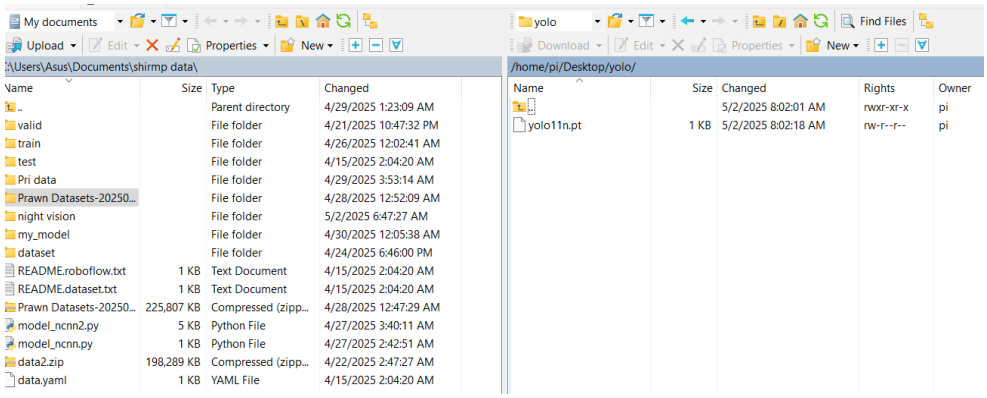


Figure 5.3.14: WinSCP to Transfer the Collected Data to Raspberry PI

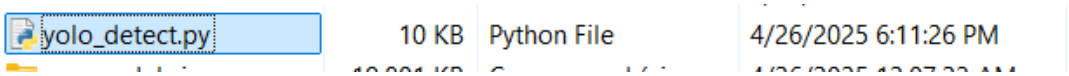


Figure 5.3.15: Python File to be Transferred to Raspberry PI

5. **Export model into ncnn format**

```
(venv) evan@raspberrypi:~/yolo $ yolo export model=yolo11n.pt format=ncnn
```

Figure 5.3.16: Command to Export the Trained Model to ncnn format

Figure 5.3.17: Converted YOLO11n Model in ncnn format

The file will convert into yolo11n_ncnn_model

6. **Run Inference**

```
pi@prawn1:~/Desktop/yolo $ python yolo_detect.py --model=yolo11n_ncnn_model --so
urce=picamera0 --resolution=640x640
```

Figure 5.3.18: Command to Run Inference in Raspberry PI

This command in Figure 4.3.17 runs the model in Raspberry Pi, by using the PiCamera with an input resolution of 640x640 for optimal speed and compatibility.

7. **Result**

Figure 5.3.19 Detection Results in Raspberry PI

Figure 4.3.18 illustrates the FPS and the number of objects detected. However, the presence of two prawns in the frame indicates potential duplication or misdetection, which may compromise the accuracy of the data.

5.3.4 Cron Job to Collect Data Through Pi-Camera

Due to the limited amount of available data for training an accurate model, a Cron Job was configured on the Raspberry Pi to automate data collection via the PiCamera. This method provides a convenient and consistent way to gather additional images over time.

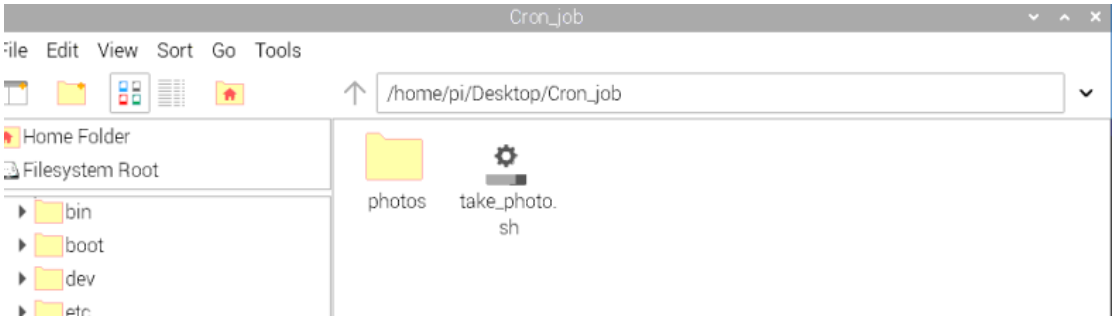


Figure 5.3.20: Content of the Cron_job Folder

Figure 5.3.21: Shell Script to Take Photo

Figure 5.3.22: Command to Open Crontab Configuration File

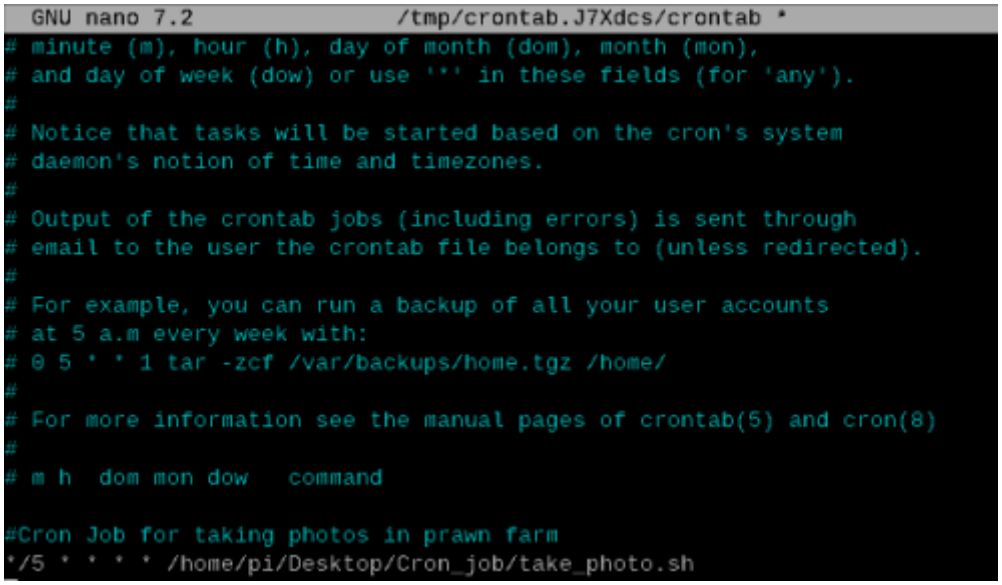


Figure 5.3.23: Crontab Configuration

A new folder named Cron_job was created on the Raspberry Pi to manage scheduled data collection tasks. Within this folder, a shell script (take_photo.sh) was written, as shown in Figure 4.3.20. This script captures an image using the PiCamera and saves it to the photos directory located inside the Cron_job folder. Figure 4.3.21 illustrates the command used to open the crontab configuration file. As shown in Figure 4.3.22, the crontab entry */5 * * * * /home/pi/Desktop/Cron_job/take_photo.sh schedules the take_photo.sh script to execute every 5 minutes, enabling automated and periodic image capture.

CHAPTER 6 System Evaluation and Discussion

6.1 Performance Metrics

6.1.1 mAP50-95

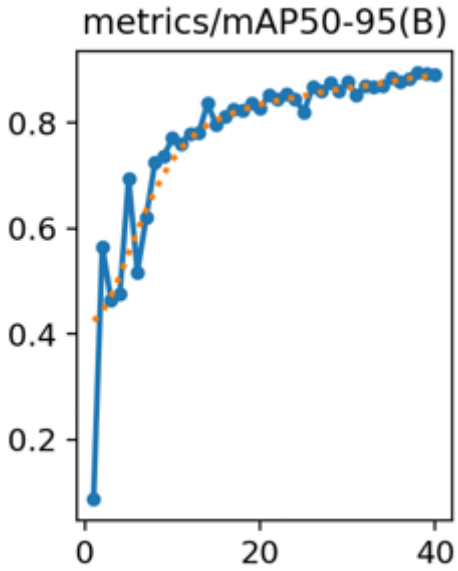


Figure 6.1.1: mAP50-95

One common statistic used to assess the accuracy of object detectors such as YOLO is the mean average precision, or mAP. IoU 0.5-0.95 provides the mAP values. One may see that from the figure.

After about five epochs, the mAP@0.5-0.95 rapidly improves from its low starting point during the first epochs (<0.2). In the last epochs, the curve stabilises between 0.85 and 0.88, displaying a consistent rising tendency. This suggests that as the model is trained, its detection accuracy increases dramatically and eventually converges to a high degree of performance.

All things considered, the high final mAP@0.5-0.95 values show that the model has mastered the ability to accurately forecast bounding boxes over a variety of IoU thresholds. At later epochs, the curve gradually flattens, indicating that the model has achieved a high degree of convergence.

6.1.2 F1 Curve

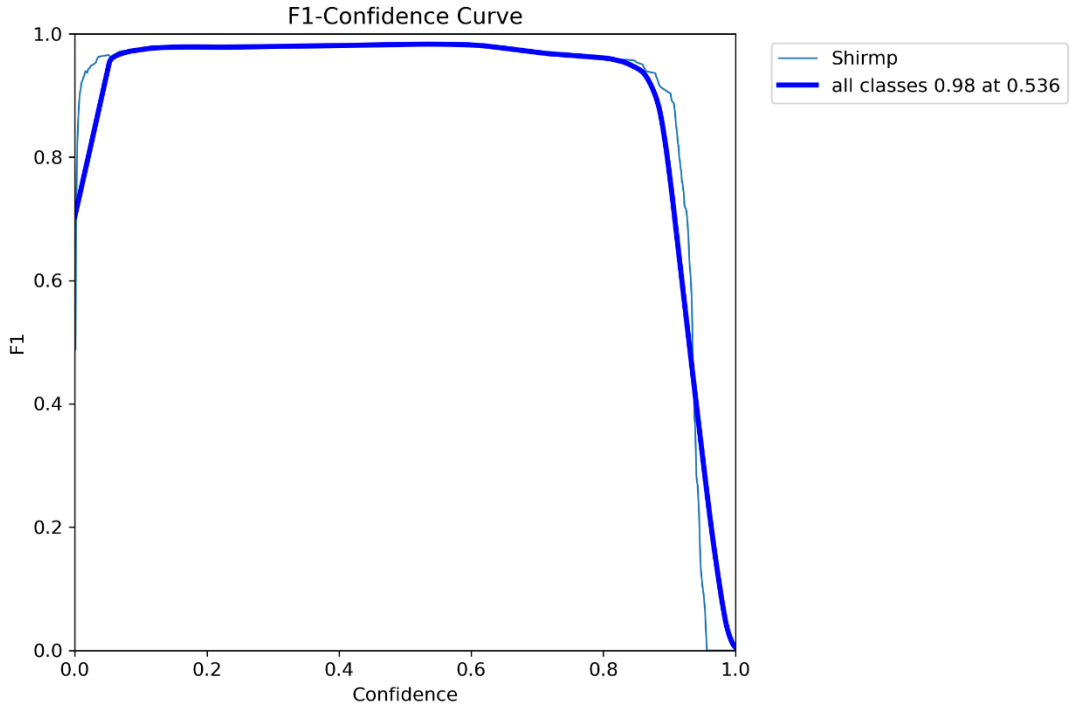


Figure 6.1.2: F1-Confidence Curve

The F1–Confidence curve shows how the F1 score changes as the confidence level rises or falls. The F1 score offers a fair way to assess detection performance since it is the harmonic means of precision and recall. The highest F1 score, at a confidence level of 0.536, is 0.98 for all classes. The F1 curve for the "shrimp" class shows consistent detection accuracy, closely following the overall performance. Over a broad range of confidence levels (about 0.2 to 0.85), the F1 score stays high (>0.95), indicating that the model is accurate and resilient to varying threshold choices. The high peak F1 score indicates a superb memory and precision balance. The curve's flat plateau further suggests stability, indicating that the model is not unduly affected by the confidence threshold selection. In reality, this is preferable since it permits flexibility in determining deployment thresholds without significantly compromising accuracy.

6.1.3 Recall-Confidence Curve

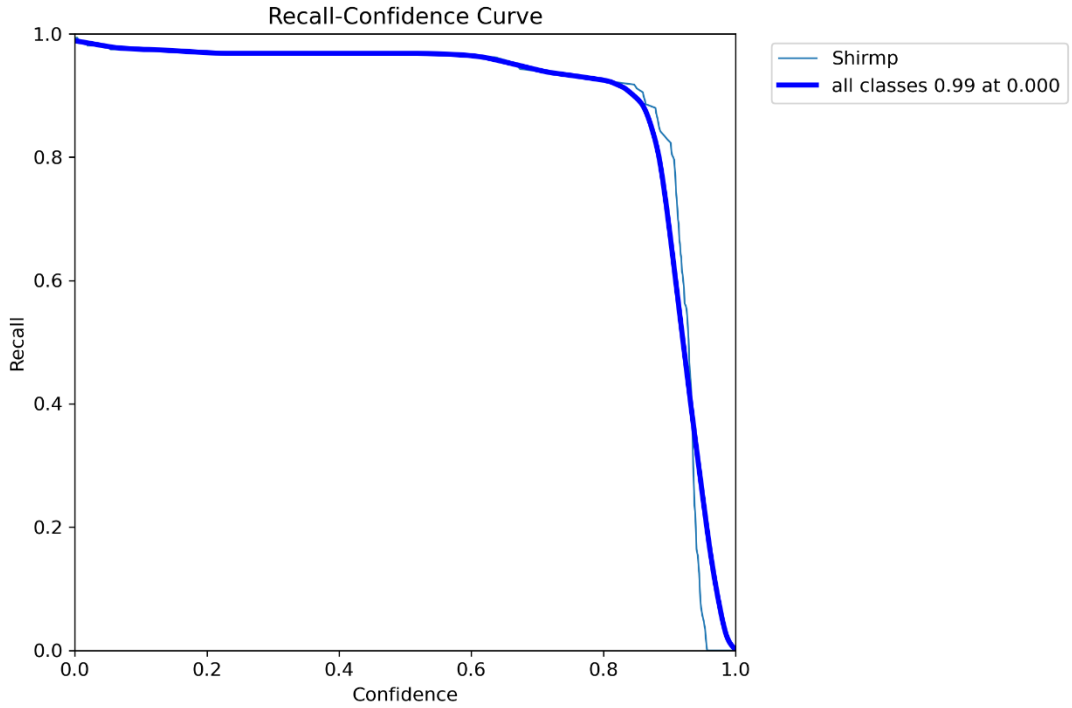


Figure 6.1.3: Recall-Confidence Curve

According to the graph, at specific confidence levels, the model seems to have a high recall for every class. This indicates that a significant percentage of the actual prawns in the dataset can be accurately identified by the model. The model performs well overall, as evidenced by the recall for all classes combined reaching 0.99.

6.1.4 Precision Curve

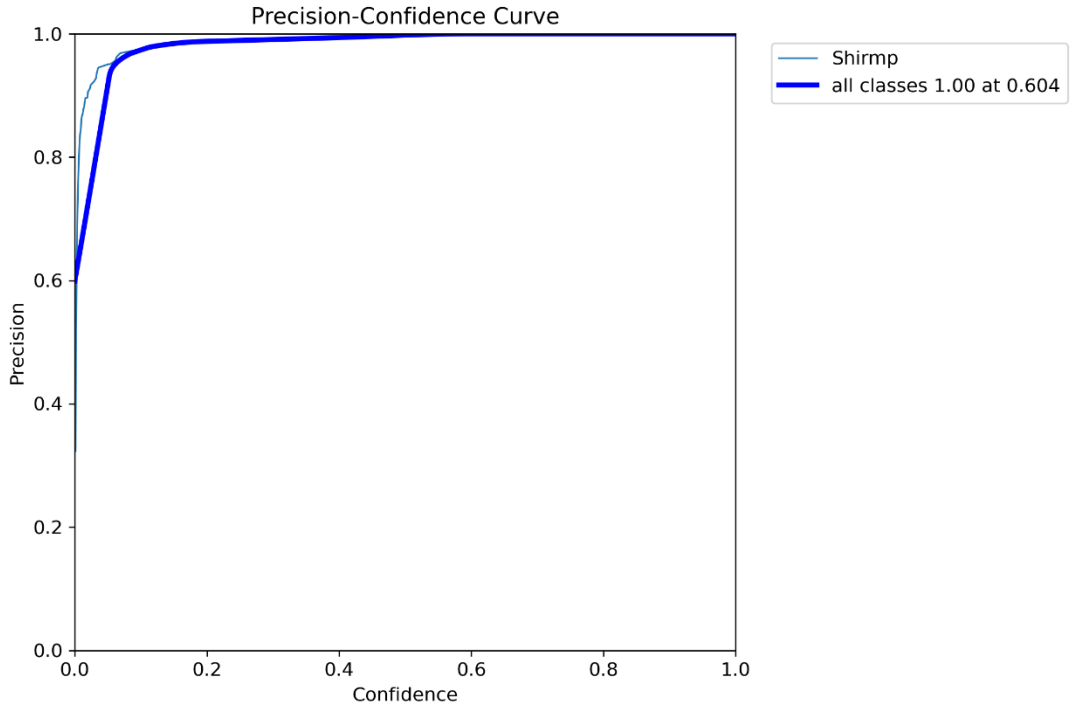


Figure 6.1.4: Precision-Confidence Curve

The model has great precision across confidence levels, according to the graph. Predictions for the Shrimp class are extremely accurate since precision rapidly stabilizes around 1.0. Overall, at a confidence level of about 0.5, the model achieves perfect precision (1.00), demonstrating strong reliability.

6.2 Testing Setup and Results

The Raspberry Pi must be configured for the project in order to perform computer vision training by taking pictures for local data. Refer to chapter 4.3.4 for the location where the Cron-job will be run to gather the data. We'll train and use the local model. To test the outcome, a local density and growth estimation will be conducted. The Raspberry Pi's height will be 42 cm, and the region that the pi camera can detect will be 30 cm by 40 cm. Local data should also be used to define the weight-length relationship. Using the YOLO11n model, the training procedure will be the same as in chapter 4.3.2. However, the shrimp's body and carapace will be the exclusive focus of this training.

```python
# Camera calibration constants for 40cm x 30cm box
BOX_WIDTH_CM = 40.0  # Width of the measurement box in cm
BOX_HEIGHT_CM = 30.0  # Height of the measurement box in cm

# Box area in square meters
BOX_AREA_SQM = (BOX_WIDTH_CM * BOX_HEIGHT_CM) / 10000.0  # Convert cm² to m²

def calculate_shrimp_length(bbox_width_px, bbox_height_px, frame_width, frame_height):
    """
    Calculate shrimp length in cm based on bounding box dimensions

    Args:
        bbox_width_px: Bounding box width in pixels
        bbox_height_px: Bounding box height in pixels
        frame_width: Frame width in pixels
        frame_height: Frame height in pixels

    Returns:
        Length in cm (using the longer dimension of the bounding box)
    """
    # Calculate pixels per cm for both dimensions
    pixels_per_cm_width = frame_width / BOX_WIDTH_CM
```

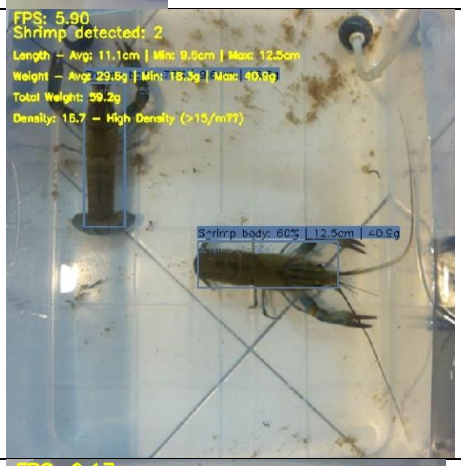
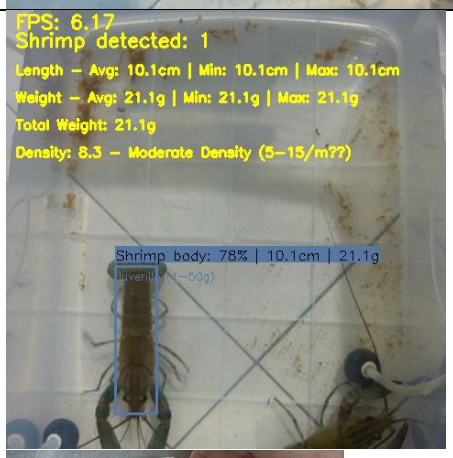
Figure 6.2.1 Area detected by Pi-Camera

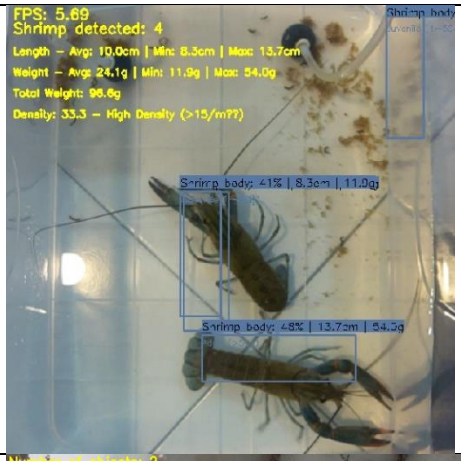
```python
def calculate_shrimp_weight(length_cm):
    """
    Calculate shrimp weight based on length using the formula: W = 0.018 * L^3.06

    Args:
        length_cm: Shrimp length in cm

    Returns:
        Weight in grams
    """
    weight_grams = 0.018 * (length_cm ** 3.06)
    return weight_grams
```

Figure 6.2.2 Weight-length Relationship Calculation

Figure 6.2.3 Image captured by Pi-Camera

Table 6.2.1: Test Cases and Results

No	Test Case	Remark
1		A bounding box is used to detect the prawn body.
2		The shrimps are identified, their density is measured, and they are appropriately categorized.
3	 	When compared to their real length, the prawns are discovered and measured with an inaccuracy of -0.9 cm. To calculate the weight, however, the length must be known.

4		Although there are just two prawns in this instance, they are detected. The background was identified by the model as FP, or a prawn.
5		A bounding box is used to accurately detect every prawn.
6		In this instance, prawns have been found. The detected prawn is not properly bounded, though.

6.3 Project Challenges

Lack of data is one of the project's biggest problems. In computer vision, training requires a large amount of data and might take many months. In addition to the typical environment, night vision must be incorporated to increase reliability. The time and personnel expenses are high. Data labelling necessitates manual labelling, which is expensive. The choice of model is the next obstacle. It is needed to ensure that the data can be continuously incorporated into the model in order to overcome these obstacles and improve the model's accuracy. By using cloud-based training and pushing the most recent model to edge devices, federated learning can overcome these

problems. To ensure smooth operation on edge devices, the model to be used should be lightweight. YOLO11n will be utilised to solve this problem because it is straightforward to train and small enough to run the detection.

6.4 Objectives Evaluation

The project effectively put into practice a computer vision system that uses bounding boxes to identify prawn bodies. The model's capacity to detect and assess shrimp presence in aquaculture environments was demonstrated by the accurate measurement and classification of shrimp density. In order to provide more reliable population statistics and lessen dependency on manual monitoring, this automation is a significant step.

Additionally, the method demonstrated dependable measuring accuracy, with a mean error of -0.9 cm between real measurements and prawn length estimation. Although this margin is suitable for first testing, it identifies locations where boundary precision needs to be increased to improve growth monitoring even more. Density estimation performed as predicted in test situations, correctly identifying and classifying two shrimps. Limitations still exist, though, as the model occasionally misidentified background objects as shrimp and occasionally generated bounding boxes that were not in proper alignment with shrimp bodies, which affected weight estimation and measurement accuracy.

Cost-effectiveness and accessibility are two other accomplishments of the project. All of the software tools utilised were open-source, and the system can be set up with reasonably priced hardware, including a Raspberry Pi and Pi-Camera, which should cost between $60 and $80. Due to its affordability, the solution is especially appealing to small and medium-sized farms as a less expensive substitute for pricey commercial equipment. Additionally, by using federated learning, the model may ensure long-term adaptability and scalability by continuously improving over time as additional data is gathered.

In conclusion, the project has shown that inexpensive, lightweight computer vision models may be used for prawn detection, density estimation, and growth monitoring. The accomplishment of these goals establishes groundwork for a scalable, reasonably

priced, and constantly evolving system that can greatly assist prawn producers, even though enhancements are required to solve bounding inaccuracies and false positives.

CHAPTER 7 Conclusion and Recommendation

7.1 Conclusion

This project, titled "Utilising Computer Vision Techniques for Automated Density and Growth Estimation in Precision Aquaculture Systems for Prawn Cultivation," successfully tackles significant challenges in the prawn farming industry by developing an innovative, automated monitoring system. The global shrimp industry has experienced substantial growth, with countries like Ecuador achieving a compound annual growth rate (CAGR) of 25% between 2020 and mid-2023, driven by rising demand in markets such as the US. However, traditional methods for monitoring prawn density and growth, such as cast-net sampling, are labour-intensive, time-consuming, and prone to human error, leading to inefficiencies like overfeeding, underfeeding, and suboptimal harvest timing. These inefficiencies contribute to wasted resources, environmental degradation, and reduced farm productivity. Additionally, the lack of real-time monitoring tools hinders precise feeding and harvest planning, while small- and medium-scale farmers face barriers in adopting advanced technologies due to high costs and technical complexity. Motivated by these issues, the project aimed to enhance productivity and sustainability in prawn farming by automating monitoring processes, improving accuracy, and designing a cost-effective solution accessible to smaller operations.

The proposed solution leverages computer vision and machine learning to address these challenges, focusing on three key objectives: automating prawn density and growth estimation, enhancing monitoring accuracy and efficiency, and ensuring affordability and scalability. A lightweight YOLO11n neural network was implemented to automate the estimation process, achieving a real-time inference speed of 5 FPS on a Raspberry Pi 5. Machine learning algorithms were integrated to improve the precision of prawn population tracking by analysing visual data from camera feeds, enabling accurate monitoring of growth and density across diverse conditions. To ensure accessibility, the system uses affordable hardware, including the Raspberry Pi 5 ($60-$80) and PiCamera (Night Vision), making it viable for small- and medium-scale farmers. The system was deployed in a controlled prawn pond environment, capturing 2000 images of Cherax quadricarinatus prawns under varying lighting and water clarity conditions to ensure robustness.

The literature review revealed several weaknesses in existing systems, such as environmental sensitivity, high costs, technical complexity, data scarcity, and limited field validation, which this solution mitigates. For instance, a prior smart headset system struggled with inconsistent lighting and depth camera limitations, while this system uses YOLO11n, trained on diverse images, to handle such variability. Similarly, automatic counting methods faced high computational demands, which the lightweight YOLO11n model on Raspberry Pi addresses efficiently. The system also simplifies operation through automated data collection via a Cron Job, capturing images every 5 minutes to build a farm-specific dataset, overcoming the data scarcity issue noted in AI-based prawn farming studies. Additionally, the use of affordable hardware reduces financial barriers highlighted across all reviewed studies, ensuring scalability for smaller farms.

A novel idea derived from this project is the integration of automated data collection via a Cron Job on the Raspberry Pi to continuously build a tailored dataset for prawn monitoring. Unlike prior studies that relied on manual data collection or pre-existing datasets, this approach ensures a steady stream of farm-specific images, capturing prawns across growth stages and environmental conditions without additional labor. This automation not only addresses the lack of standardized datasets but also supports ongoing model improvement, enhancing long-term accuracy and adaptability. Furthermore, the use of YOLO11n on a Raspberry Pi, optimized with NCNN for real-time performance, provides a practical balance of efficiency and affordability, a combination less emphasized in previous works that often prioritized high-end hardware.

Preliminary results demonstrate the system’s feasibility, though challenges remain. The YOLO11n model successfully detected prawns in real-time, but initial tests showed inaccuracies, such as detecting only one of two prawns in an image, underscoring the need for more data and fine-tuning. The Cron Job feature mitigates this by expanding the dataset over time, while the device comparison confirmed the Raspberry Pi 5 as the optimal choice for small-scale farmers due to its affordability, ease of use, and camera integration. Overall, this project contributes to the aquaculture industry by providing a practical, data-driven tool that enhances operational efficiency, reduces resource waste, and promotes sustainability, particularly for small- and medium-scale prawn farmers, aligning with the project's original goal.

7.2 Recommendation

A number of enhancements are suggested to improve the prawn detecting method even more. First, by using data augmentation and fine-tuning annotations, the training dataset can be expanded and diversified to include images of prawns orientated in various lighting and water conditions, hence increasing the precision of bounding boxes. Equally crucial is lowering false positives, which can be accomplished by using background filtering, modifying the confidence threshold, and adding negative samples during training. Furthermore, broadening the dataset will improve the model's generalisation across various settings and phases of prawn development. Since length is currently used to estimate weight, a local weight-length connection needs to be established in the appropriate setting.Adopting federated learning would also guarantee that, without sacrificing scalability or privacy, the system keeps getting better as more farms contribute data. Practically speaking, the system should have an intuitive user interface for farmers that offers visual dashboards, real-time information, and automated warnings to aid in decision-making. Last but not least, extensive field testing in various aquaculture settings is necessary to evaluate the system's resilience, validate its cost-effectiveness, and show how valuable it is for raising farm output.

REFERENCES

[1] “World Aquaculture Society: Shrimp farming advances, challenges, and opportunities - world aquaculture society,” Shrimp farming advances, challenges, and opportunities, https://www.was.org/article/Shrimp-farming-advances-challenges-and-opportunities.aspx (accessed Sep. 11, 2024).

[2] “INTENSIFYING AND EXPANDING SUSTAINABLE AQUACULTURE PRODUCTION,” Intensifying and expanding sustainable aquaculture production, https://openknowledge.fao.org/server/api/core/bitstreams/9df19f53-b931-4d04-acd3-58a71c6b1a5b/content/sofia/2022/expanding-sustainable-aquaculture-production.html (accessed Sep. 11, 2024).

[3] X. Mingze, A. Rahman, C. Nguyen, S. Arnold, and M. John, “Smart Headset, Computer Vision and Machine Learning for Efficient Prawn Farm Management,” Aquacultural Engineering, vol. 102, 2023. doi:https://doi.org/10.1016/j.aquaeng.2023.102339

[4] H. Zhou et al., “Instance segmentation of shrimp based on contrastive learning,” Applied Sciences, vol. 13, no. 12, p. 6979, Jun. 2023. doi:10.3390/app13126979

[5] J. Kite-Powell, “Here’s how this company uses AI to grow shrimp sustainably,” Forbes, https://www.forbes.com/sites/jenniferhicks/2022/10/28/heres-how-this-company-uses-ai-to-grow-shrimp-sustainably/# (accessed Sep. 11, 2024).

[6] X. Saining, R. Girshick, P. Dollár, T. Zhuowen, and K. He, “Aggregated Residual Transformations for Deep Neural Networks,” Computer Vision and Pattern Recognition, 2017.

[7] D. Li et al., “Automatic counting methods in aquaculture: A Review,” Journal of the World Aquaculture Society, vol. 52, no. 2, pp. 269–283, Oct. 2020. doi:10.1111/jwas.12745

[8] B. V. C. Sekhar, K. Rangaswamy, P. Anjaiah, K. Naveen, and K. Sumalatha, “Fish Species Detection and Recognition UsingRivista Italiana Filosofia Analiticaer Learning Approach,” Rivista Italiana Filosofia Analitica, vol. 14, no. 1, 2023.

[9] Jones, C, “Fisheries and Aquaculture,” Fao Fisheries & Aquaculture, https://www.fao.org/fishery/en/culturedspecies/cherax_quadricarinatus (accessed Sep. 22, 2025).

[10] “FWS,” Redclaw (Cherax quadricarinatus) Ecological Risk Screening Summary, https://www.fws.gov/sites/default/files/documents/Ecological-Risk-Screening-Summary-Redclaw.pdf (accessed Apr. 4, 2020).

[11] Naguib S., Ahmad Safuan Sallehuddin, Ahmad Syazni, and Mohamad Zulkarnain, “(pdf) length-weight relationship and condition factor of Australian red claw crayfish (Cherax quadricarinatus) from three locations in Peninsular Malaysia,” Length-weight relationship and condition factor of Australian red claw crayfish (Cherax quadricarinatus) from three locations in Peninsular Malaysia, https://www.researchgate.net/publication/354721150_Length-weight_relationship_and_condition_factor_of_Australian_red_claw_crayfish_Cherax_quadricarinatus_from_three_locations_in_Peninsular_Malaysia (accessed Sep. 2021).

[12] C. Jones, “(PDF) Redclaw Crayfish,” Redclaw Crayfish, https://www.researchgate.net/publication/257939107_Redclaw_Crayfish (accessed Jan. 1998).

[13] Buy A raspberry pi 5 – raspberry pi, https://www.raspberrypi.com/products/raspberry-pi-5/ (accessed Sep. 22, 2025).

[14] “Nvidia Jetson Orin Nano Super 8GB dev kit,” Cytron Technologies Malaysia, https://my.cytron.io/p-nvidia-jetson-orin-nano-8gb-dev-kit?gclid=Cj0KCQjwxL7GBhDXARIsAGOcmIOncrfiHJo_7FWqvSh-EnVLmMqlI0wf7TKsD0fsx_hbLJycGVtauRQaAlP_EALw_wcB (accessed Sep. 22, 2025).

[15] “Radxa Rock 4B,” Radxa News RSS, https://radxa.com/products/rock4/4b/ (accessed Sep. 22, 2025).

[16] Odroid XU4 - octa core Odroid Computer, https://www.odroid.co.uk/odroid-xu4 (accessed Sep. 22, 2025).

[17] Edje Electronics, “YOLO11, YOLOv8, and YOLO v5 Speed and Accuracy Comparison on Raspberry Pi 5 and NVIDIA RTX 4050 GPU,” YouTube. Jan. 06, 2025. [Online]. Available: https://www.youtube.com/watch?v=_WKS4E9SmkA

[18] Ultralytics, “Benchmark,” Ultralytics YOLO Docs, https://docs.ultralytics.com/modes/benchmark/#benchmark-visualization (accessed Sep. 22, 2025).

APPENDIX

A.1 Poster