Automatic Analysis of Fish Farm Environment

by

Ahmed Waleed
Hadeer Medhat
Mariam Esmail
Kareem Osama

A dissertation submitted in partial fulfillment of the
requirements for the degree of
Bachelor of computer science

in

Department of Computer Science

in the

Faculty of Computer Science
of the
Misr International University, EGYPT

Thesis advisor:
Dr.Taraggy M Ghanim
Eng.Radwa Samy

(June 2020)

Abstract

Fish farm environment strongly affects fish behavior and health. An uncontrolled environment may cause fish mortality and severe losses in fish production that may affect countries' economies. The proposed system serves this field by analyzing fish farm environment. Many challenges are faced such as lack of data-sets and research. Additionally, fish behavior is hardly tracked due to fast fish movements, similar appearance, and high density of fish farms. The automatic disease recognition process is also challenging due to unclear water vision, noisy images, and the presence of objects other than fish in images. The proposed system is composed of Raspberry pi hardware circuit which is connected to sensors and camera. The sensors are responsible for examining water quality by measuring water temperature and rate of pH, while the camera acquires video frames for tracking fish movement by applying Kanade-Lucas-Tomasi (KLT) algorithm and detecting diseases by applying Convolutional Neural Networks and segmentation techniques. These diseases are Epizootic ulcerative syndrome (EUS), Ichthyophthirius (Ich), and Columnaris. Finally, the system sends a notification through an android or web application to inform users of any improper farm conditions and any detected infections.

Acknowledgments

First of all, We want to express our sincere thanks to Dr.Taraggy Mohiy, our thesis advisor for her great effort, expertise, guidance, and for providing us will all the necessary resources throughout the year along with his continuous belief in us and our project. Also, we would like to thank Eng. Radwa Samy for helping us at any time. We are also grateful to Dr. Ayman Ammar and Dr. Hussien khelfallah for providing us with the needed resources and for their continuous support and opening door for helping us at any time.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

Fish farms have a great role in food security and livelihood and are a source of income and social development in developing countries. Most of the eaten seafood is farmed. Production of farmed fish makes up forty-four percent of total fish production [1]. Farmed fish continues to grow faster than other major food production sectors. Annual growth declined to a moderate 5.8 percent during the period 2001–2016 [2], although double-digit growth still occurred in a small number of individual countries, particularly in Africa from 2006 to 2010 as shown in figure 1.1.

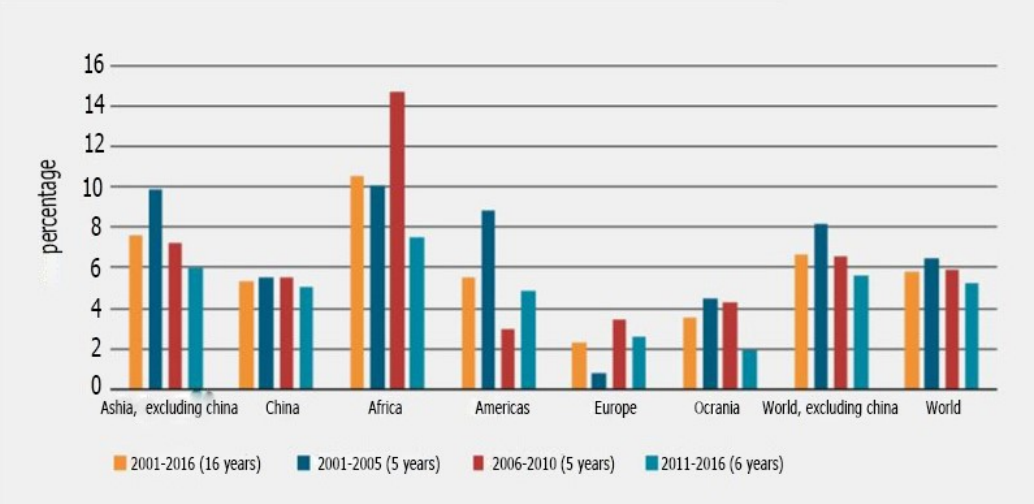


Figure 1.1: Average annual growth rate of fish farm production

Huge loss of production in fish farms is occurring because of many reasons. Among these causes, the disease is the most serious constraint that causes damage to the livelihood of farmers, reduced incomes, and food insecurity. Like any living organism, fish suffers from various diseases that cause their death. The annual loss of revenues because of the disease reaches up to billions of dollars. Different types of fish diseases are shown in figure. 1.2.

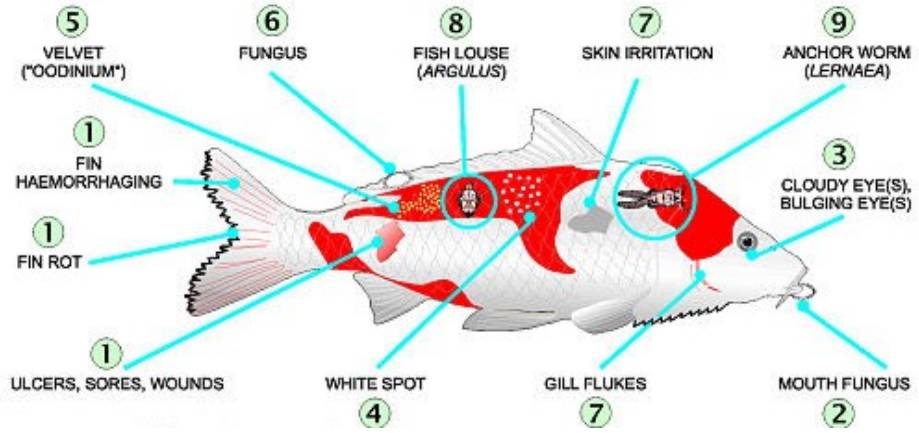


Figure 1.2: Different type of fish diseases

Fish diseases spread quickly in fish farms, therefore, preventing and controlling fish diseases is very important. Most of the available methods focus on classifying and detecting fish disease outside of water because underwater classification show challenges such as background noises, the presence of other water bodies in images, and image quality.

Analysis of fish behavior helps in the expectation of fish diseases. Tracking fish in the video is an effective way to investigate fish behavior. However, there are many difficulties in tracking which become greater in underwater environmental conditions. Fish tracking is a challenging task due to the nature of the videos in which fish is tracked. Traditional ways deal with videos in a controlled environment such as fish tanks with high lightening, fixed objects in the water, and purity of water. However, our videos were taken in uncontrolled earthen ponds in fish farms in which the vision is not clear, visual similarity between the fish in the same farm, fast fish movements, and water cleanness, as shown in figure 1.3. Fish farm videos also contain non-fish objects as algae. All these conditions affect the quality of the videos taken.

Figure 1.3: Earthen pond

The Internet of Things (IoT) is all of the things that are connected to the Internet. It is the ability to transfer data over a network. The IoT architecture is based on a 3-layer system that consists of a hardware layer, communication layer, and interface layer. IoT systems require some way of communication between different endpoints. These endpoints can be anything from IoT devices to applications, and eventually, the data needs to be stored somewhere for further processing. The hardware devices that are used in the proposed system are Raspberry pi 3 that is responsible to send data to the firebase database. This will communicate with android or web applications that will read the data and display it perfectly.

In this proposed system, deep learning is capable of analyzing fish farm environment for detecting and tracking fish. The proposed system consists of three subsystems: the hardware circuit, fish detection, and tracking, and finally the interface part which can be an android or web application. The hardware part is composed of a Raspberry pi hardware circuit which is connected to sensors and camera. The sensors are responsible for examining water quality by measuring water temperature and rate of pH, while the camera acquires video frames. Raspberry pi gets the sensors' readings and captured videos to pass them to the computer system. In Fish detection, convolutional Neural Networks and segmentation techniques are applied to fish images taken from videos to detect three different types of fish diseases. These diseases are Epizootic ulcerative syndrome (EUS), Ichthyophthirius (Ich), and Columnaris. The tracking was carried out by applying the Kanade-Lucas-Tomasi (KLT) algorithm and fish velocity is then calculated for tracking fish behavior. Finally, the system

sends a notification through an android application to inform users of any improper farm conditions and any detected infections.

1.2 Background

This section provides the user with the required background. The introduced subsections are fish diseases, analysis of different behaviors and ponds types.

1.2.1 Fish Diseases

Epizootic ulcerative syndrome (EUS) [3] is a disease caused by water mold which is known as Aphanomyces invadans. EUS is also known as red spot disease (RSD). EUS causes ugly red spots on infected fish skin, as shown in figure 1.4. These lesions can be expanded to form ulcers. EUS infection occurs when motile spores of the fungi Aphanomyces invadans in the water. More than 50 species of both farmed and wild fish, are susceptible to be infected by EUS disease.

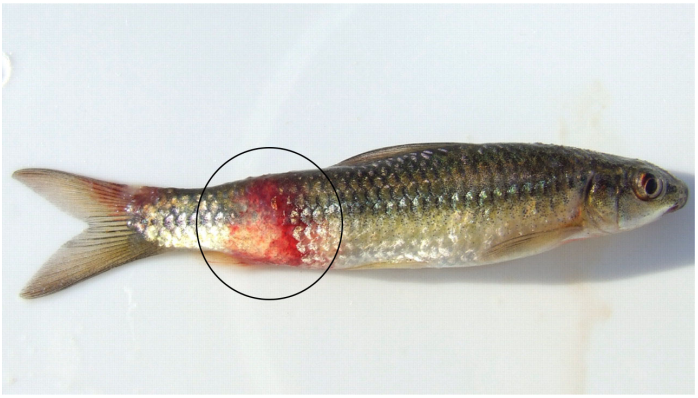


Figure 1.4: Fish infected with EUS disease

Ichthyophthirius (ICH) [4] is a large parasite that causes white spot disease that often appears on the skin and fins of infected fish as shown in figure 1.5. The disease occurs when any change happens in water temperature and the disease is particularly severe when fish are crowded. Once a fish is infected in a fish farm, it can cause large numbers of fish to die within a short period and in severe cases, control may be impossible and 100% of the fish can be expected to die.

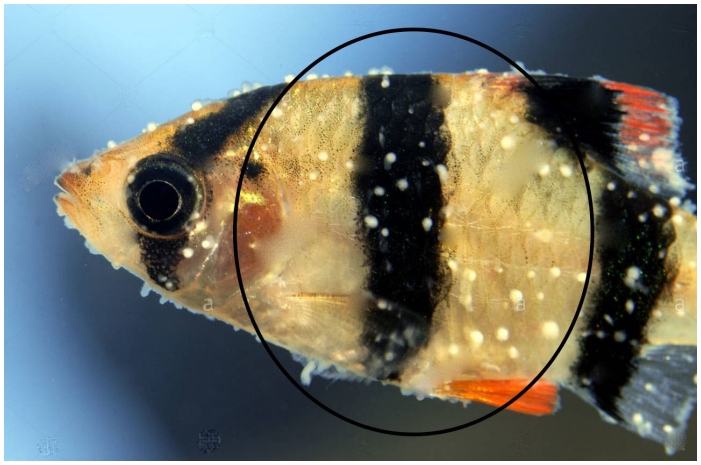


Figure 1.5: Fish infected with ICH disease

Columnaris [5] is a bacterial infection resulted from an infection caused by the Gram-negative. Columnaris is also known as cottonmouth and it affects the gills, the skin, and fins. Columnaris takes many days before causing fish death. In severe cases, the disease spread quickly. High water temperature accelerates the development of the disease. However, lowering the water temperature will not affect the outcome of the disease. Most columnaris infections are external and appear as white or grayish spots on the head and around the fins as shown in figure 1.6.

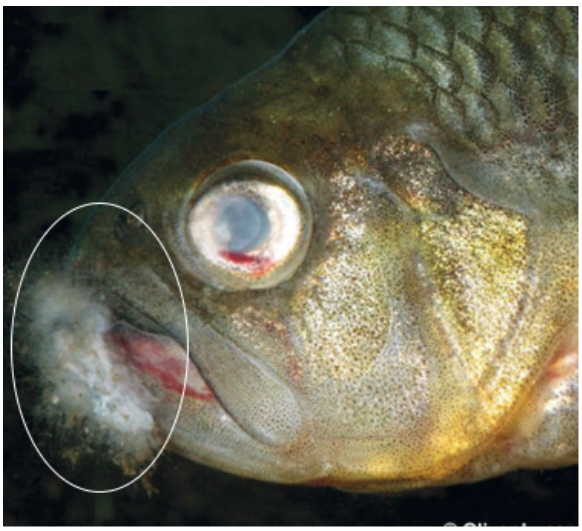


Figure 1.6: Fish infected with Columnaris disease

1.2.2 Analysis of different behaviors

Abnormal behaviors help in the expectation of fish diseases. Fish that have a disease can show a variety of behaviors. These are some common abnormal behavior in fish:

- Loss of appetite.
- Difficulty swimming.
- Fish swimming quickly.
- Gasping at surface of water
- Unusual isolation from the group of fish.

1.2.3 Ponds Types

There are different types of ponds of fish farms. These types are earthen ponds, concrete ponds, and floating cages. These ponds are categorized as either extensive or intensive. Extensive ponds [6] are much like natural ecosystems concerning nutrient inputs, nutrient cycling, species diversity, oxygen dynamics, and level of human intervention. Examples are shown in figure.1.7.

(a) Earthen ponds (b) Tank indoor systems

Figure 1.7: Extensive ponds

Intensive ponds [7] relies on technology to raise fish in artificial tanks at very high densities. Some intensive ponds are a completely closed system, this type of pond is called an integrated recycling system. Another type is cage ponds. Examples are shown in figure. 1.8

(b) Cage Ponds (a) Integrated Recycling Ponds

Figure 1.8: Intensive Ponds

1.3 Motivation

The main motivations of our project are to prevent and control the spreading of fish diseases in fish farms. According to some similar systems which are already implemented in the field of automatic analysis of the fish farm environment, it was found that each system was able to detect only one type of fish disease. Tracking fish behavior also were not taken into consideration and no one tried to implement a system that can expect that fish could have diseases based on their behavior. We consider these facts as a second motivation in our project. Many research papers tried different algorithms like in [8] [9] [10]. Their success rate in fish disease recognition ranges between 86% to 90%. Our third motivation is to achieve higher accuracy by applying different algorithms and trying different classifying techniques. Finally, our motivation is to building a useful application to inform users of any improper change in the fish farm environment. Also, we are creating our data-set because this field has lack of data-sets and research. This project should also take into consideration that manual diagnosis is not accurate somehow and have more chances of misdiagnosis. Therefore, an automatic approach is needed by using computer vision techniques to detect and diagnose fish diseases and enhance the accuracy of results.

1.4 Problem Definition

Automatic analysis of fish farm Environment has a lot of challenging points, whether in detecting or tracking approaches. In our project, we will detect the type of fish disease and track fish to expect any abnormal behavior. The two approaches have a lot of challenging points. Fish detection shows challenges such as background noises, the presence of other water bodies in images and image quality, while fish tracking is a challenging task due to

the nature of the videos in which fish is tracked. Our videos were taken in uncontrolled earthen ponds in fish farms in which the vision is not clear, visual similarity between the fish in the same farm, fast fish movements, and water cleanness. Fish farm videos also contain non-fish objects as algae. All these conditions affect the quality of the videos taken.

1.5 Project Description

1.5.1 Objectives

Our system is trying to solve problems concerning fish disease that can cause the whole fish in farms to die. Currently, manual inspection where experts in this domain can diagnose fish disease by themselves has many difficulties, due to fast fish movement may cause tracking the infected fish to be impossible by human vision. Poor quality of unclear water in Earthen ponds also causes limitations in diagnosis and tracking. The proposed system introduces an automatic approach by using computer vision and image processing techniques to detect and diagnose fish diseases.

1.5.2 Scope

Our purpose is to describe design a system that detects fish diseases and track fish movements in fish farms automatically and examine water quality to help in the expectation of any abnormal behavior. Finally, the system sends a notification through an android application to inform users of any improper farm conditions and any detected infections.

1.5.3 Project Overview

Our proposed system aims to analyze fish farm environment by detecting fish diseases and fish behavior. Fish behavior helps in the expectation of diseased fish. The proposed system is presented in three consequent stages. During the first stage, the system contains raspberry pi 3, along with a camera and other sensors. The sensors that are used are temperature and PH to measure underwater temperature and rate of pH. while fish captures are acquired by the camera. while fish captures are acquired by the camera. Raspberry pi 3 gets the sensor's measurements and acquired fish captures and passes the data to be stored in a computer system for processing. In the second stage, the processing part concerns tracking fish for detecting any abnormal behavior in farm environment and fish infections. Infection

detection starts by pre-processing, then segmentation of infected areas, and finally classification. In the pre-processing phase, different color spaces were applied on input images which are RGB, YCbCr, and XYZ. For segmentation, we built a gaussian distribution in the XYZ color space. The XYZ color space was used in the phase of classification by applying convolution neural networks (CNN). Tracking starts by detecting fish and displays bounding boxes around the detected fishes by applying vision.CascadeObjectDetector. Tracking fish movement is then applied by the Kanade-Lucas-Tomasi (KLT) algorithm. Finally, the system sends a notification through an android or web application to inform users of any improper farm conditions and any detected infections. The overview of our proposed approach is shown in figure. 1.9.

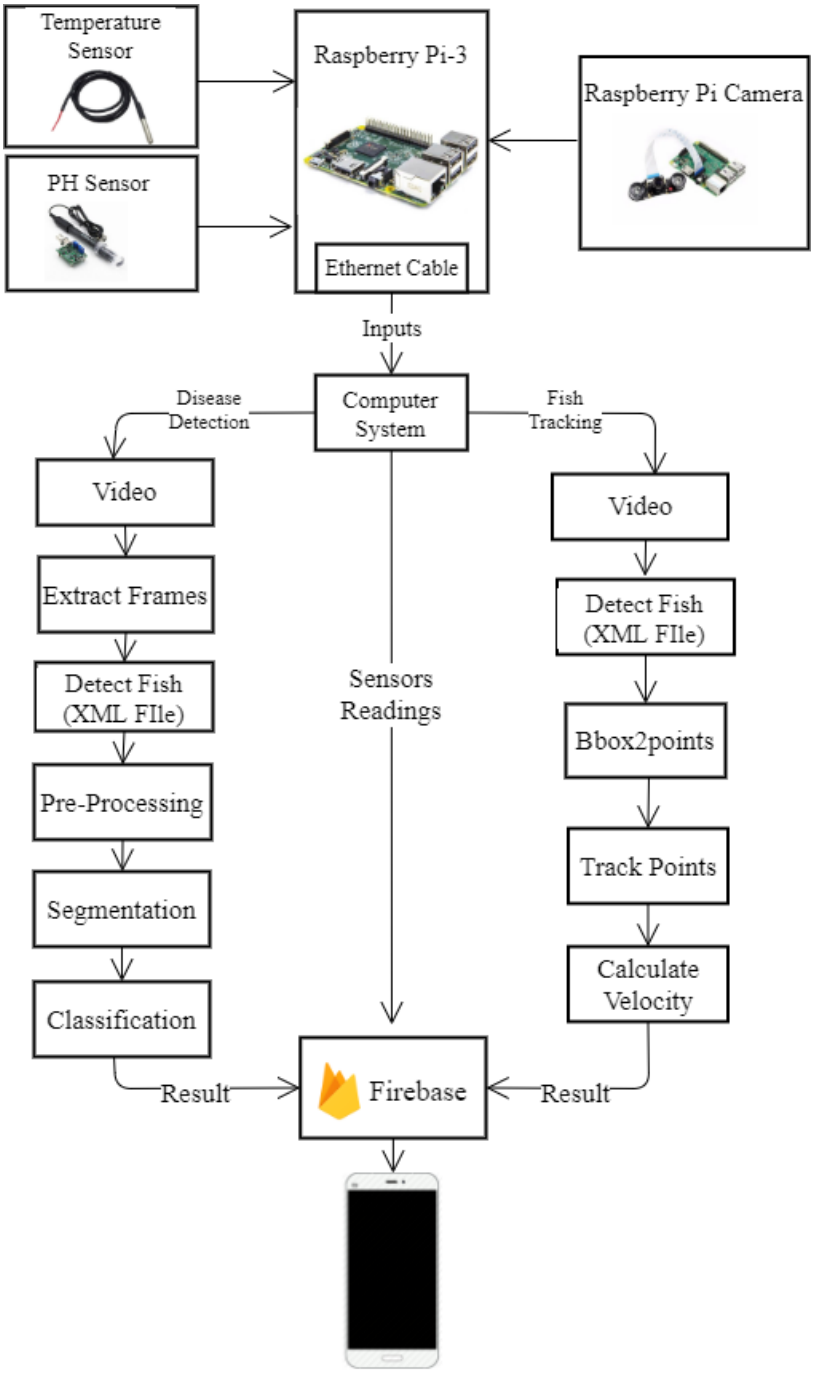


Figure 1.9: System Overview

Chapter 2

Automatic Analysis of Fish Farm Environment

2.1 Similar Systems Information

2.1.1 Similar System Description

2.1.1.1 Concepts And Analysis

In the field of automatic analysis of fish far environment, different systems are based on image processing and segmentation techniques [8], [9], [10], while others are based on deep learning techniques [11][12] [13][14]. Traditional feature were extracted in some systems, deep learning extracted features automatically in other system. The number of datasets is limited. Multiple systems cannot achieve disease identification, due to limitation of resources and lack of diseased images that is not large enough to train the system.

2.1.1.2 Different Approaches Objectives

In image processing technique, approaches [8], [9] recognize and identify the Epizootic Ulcerative syndrome (EUS) disease. Approach [9] was proposed to improve disease identification to be more accurately and also to calculate the diseased area. Another approach [10] was proposed to extract diseased regions and after final diagnosis notifications were sent to fish farmers. Different fish species were recognized by different approaches [15], [14]. Others [11], [12] accomplished fish identification in different water environments.

2.1.1.3 Categorization Based on Segmentation Approaches

Some approaches applied segmentation based on image processing techniques [8], [16], [10]. Histogram equalization was applied [8] for enhancing fish images then Canny's edge detection [16] was applied for segmentation. K-means clustering was applied [9] in segmenting diseased area in hue-saturation-value (HSV) color space, based on the hue values. Otsu's thresholding, morphological erosion and dilation operations [14] [10] were sometimes applied in similar systems for noise removal, while others applied Visual Object Tagging Tool (VOTT) [15] to remove noise. Manual cropping were performed [12] on various seabed images to separate the fish from seabed. Seabed images were labeled as "negative" to reduce misclassification.

2.1.1.4 Features Extraction

Different types of features were extracted for automatic recognition of fish diseases. Features from accelerated segment test (FAST) [17] was applied [8] on EUS infected fish images. Principle Component Analysis (PCA) [8], [9] was applied to reduce feature vector dimensions. PCA achieved good results specially after FAST calculation [8]. Two different types of feature were extracted [10] based on the polar coordinates and the geometrical features.

2.1.1.5 Classification

Neural Network [18] was applied [8] on the real image of EUS infected fish images. PCA was also applied for classification [10]. Convolutional Neural Networks (CNN) were applied for fish species classification and achieved 96.29% accuracy in approach [14], while achieved 95% accuracy in approach [15]. CNN based techniques based on You Only Look Once algorithm (YOLO) were applied [12] for real-time fish detection and recorded 93% classification accuracy and 16.7 frames per second (FPS) in processing speed [12]. Faster RCNN technique is applied to calculate the accuracy of the localized image and it reached 99% [15]. CNN model were applied for fish detection in blurry ocean water [11]. Data augmentation were applied [11] to obtain more learning resources as the available images are not sufficient for training purpose. Dropout algorithm and Loss Function were applied to solve overfitting problem [11].

2.1.1.6 Conclusion and Results

FAST-PCA-NN combination was applied for feature extraction and classification and achieved 86% accuracy, it is better than others combination technique [8]. PCA was applied successfully in fish detection and achieved 90% accuracy when applied as feature extractor [9]. FAST was applied and outperformed Histogram of gradient (HOG) feature in results [8]. Neural Network was applied and outperformed other classifiers as K-NearestNeighbor (k-NN) [8]. Canny's edge detection outperformed Roberts, Sobel and Prewitt. Some approaches worked on different color spaces [9]. Another approaches have done segmentation without identification and diagnosing fish disease [8], [9]. YOLO was applied and outperformed HOG and Support Vector Machine (SVM) in classification accuracy [12]. Different activation functions were applied to improve the accuracy such as combination of ReLu and Sigmoid functions outperformed the combination between Relu and sofmax [15].

2.1.1.7 Databases

Approaches [8], [9] have done their experiments on EUS diseased fish images. Approach [8] collected their database from National Bureau of Fish Genetic Resources (NBFGR,Lucknow) and ICAR-Central Inland Fisheries Research Institute (CIFRI), while approach [9] collected their database from the different part of the Barak Valley and Assam. Approach [10] collected their database from 8 different aquaculture farms in the Wando, Jindo, and Yosu areas of Korea and their approach were applied on 60 different parasites microscopic images. Approach [14] tested their proposed method on Fish4Knowledge [19]. Approach [12] used the labeled fishes in the wild image dataset provided by National Oceanic and Atmospheric Administration (NOAA) Fisheries [20], it also gathered additional underwater images from field experiment on Ullungdo Island, Republic of Korea. Approach [11] collected their training data set from the Gulf of Mexico by a digital camera.

2.1.1.8 Effect of IOT Technology

The proposed system [21] presents an underwater vehicle based on Raspberry PI for monitoring and protecting the aquatic ecosystem. The Raspberry Pi is used to be connected with various sensors. The sensors which are used are pressure and temperature sensors to measure the underwater pressure and temperature. These sensors parameters are displayed with the help of a display which is connected to raspberry pi as it cannot be connected

to LCD directly. Accelerometer and magnetometer are also used for speed control and for finding the direction.

Another approach [17] designed AQUABOT for detecting aquatic debris. Aquatic debris affects the ecosystems badly. It cause damage to marine life, water transport and human health. AQUABOT is a vision-based monitoring robot system which include raspberry pi connected to camera and other sensors for debris detection. This paper developed several lightweight for debris detection which include an image registration algorithm and an adaptive background subtraction algorithm. Image registration algorithm is applied to extract the horizon line over the water and use it to record images to reduce the effect of camera shake, while adaptive background subtraction algorithm is used reliable detection of debris objects.

Different approaches [22] implemented Nusantara 3-Autonomous Underwater Vehicle (N3-AUV) for exploring underwater. N3-AUV is a robot that can move underwater based on a program that is installed in the microprocessor of the Vehicle body. The N3-AUV include two cameras, a Raspberry Pi, a depth sensor, and a compass. It uses two cameras, where the first camera is placed under the frame to record pictures of cases under the car and the second in front of the frame to record the conditions in front of the vehicle. The N3-AUV performance was tested in a pool. N3-AUV achieved an average speed of about 0.5 m/s and a maximum speed of about 1 m/s.

2.1.2 Comparison with Proposed Project

Table 2.1: Comparative results of similar systems

Systems	Recognition problem	Features extraction	Classifier	Recognition rate
[8]	EUS disease	FAST	Neural Network	86%
[9]	EUS disease	PCA	Mophological open	90%
[10]	white spot, trichodina and scuticociliate	polar coordinates, geometrical features	PCA	90%
[23]	Fish species	Artificial neural networks (ANN)	ANN	99%
[15]	Fish species	CNN	CNN	95%
[14]	Fish species	CNN	CNN	96.29%
[11]	Fish detection	CNN	YOLO	93%

2.1.3 Screen Shots from previous systems

The following figures shows screen shots from systems mentioned in the previously.

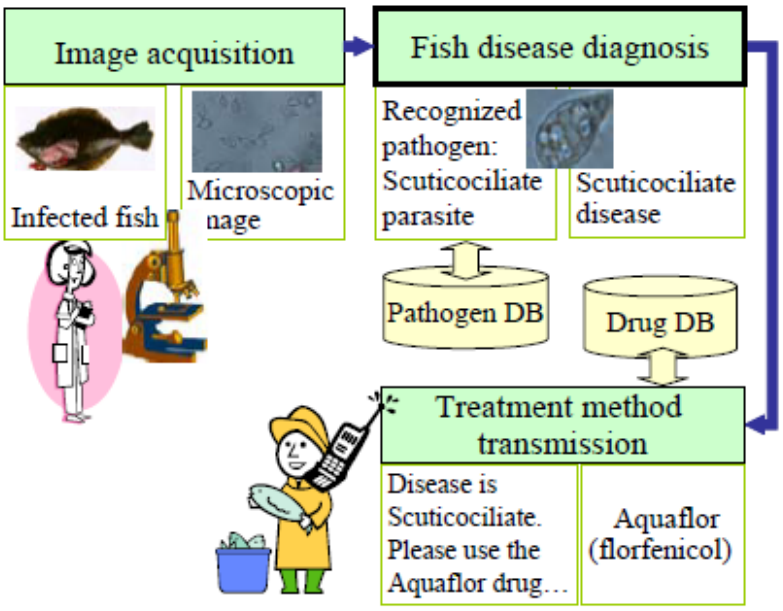


Figure 2.1: Flow of the developed fish disease diagnosis system based on image processing

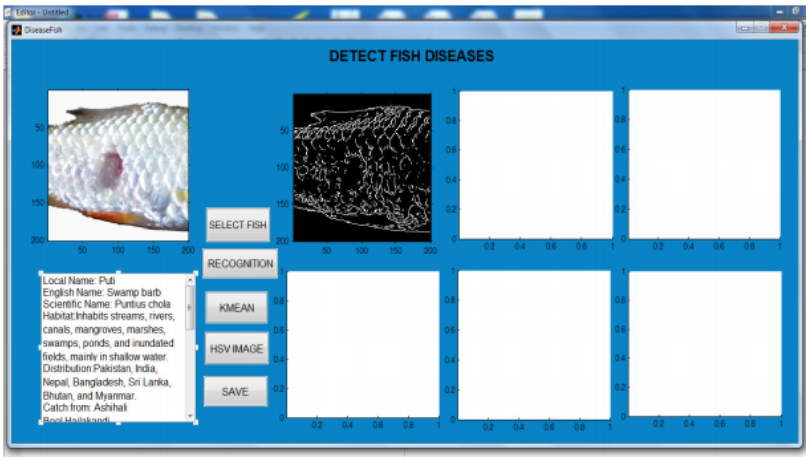


Figure 2.2: Fish Recognition using PCA

2.2 Project Management and Deliverables

2.2.1 Tasks and Time Plan

Table 2.2: Time plan table

Task	Start Date	End Date
Proposal evaluation	12/9/2019	26/9/2019
Writing SRS document	30/10/2019	28/11/2019
SRS evaluation	28/11/2019	28/11/2019
Writing SDD document	1/1/2020	12/2/2020
SDD evaluation	12/2/2020	15/2/2020
System implementation	25/4/2020	21/5/2020
Implementation evaluation	21/5/2020	24/5/2020
Final presentation	4/7/2020	4/7/2020

2.2.2 Budget

Table 2.3: Budget table

Item	Quantity	Cost
Raspberry Pi 3	1	1200 LE
Temperature Sensor	1	55 LE
PH Sensor	1	700 LE
Raspberry Pi Camera	1	300 LE
Cable camera	1	100 LE
Waterproof box	1	1000 LE
Memory	1	200 LE
Cables	1	200 LE
Matlab original version	1	8800 LE

Chapter 3

Software Requirements Specifications

3.1 Introduction

3.1.1 The Purpose

The purpose of this software requirements specification (SRS) document is to outline our system requirements for automatic analysis of the fish farm environment. The main requirements for this system are to identify and diagnose some fish diseases before spreading, and analyzing fish behavior as it helps in Prediction and detection of fish disease. The system is proposed to automatically recognize and identify three different types of fish diseases. These diseases are Epizootic ulcerative syndrome (EUS), Ichthyophthirius (Ich), and Columnaris. This document will provide a fulfilled illustration of every single stage and algorithms used in this stage. Along with a full description of what the system will do. This software requirements specification document defines how our audience sees the product and its functionality.

3.1.2 Scope

This SRS document targets owners of fish farms and experts in the fish farm domain. Our application will help them in saving much more time rather than manual detection. It will also be beneficial for the government in increasing fish production. This document will provide details about user characteristics and problems that user face that our project will

solve it. Fish disease is a substantial source of loss to the owner of fish farms. Production costs increase due to outbreaks of fish diseases due to the cost of treatment. Therefore, our automatic identification system for diseased fish is necessary to prevent fish diseases before huge losses occur.

3.2 Overview

Our proposed system aims to analyze fish farm environment by detecting fish diseases and fish behavior. Raspberry Pi kit is used and connected to sensors, camera and a personal computer (PC). The proposed system is presented in three consequent stages. During the first stage, water quality is examined by measuring water temperature and rate of pH, while fish captures are acquired by the camera. The kit gets the sensor's measurements and acquired fish captures. In the second stage, all inputs are passed to the PC for processing. This processing concerns detecting any abnormal behavior in the farm environment and fish infections. Infection detection starts by pre-processing, then segmentation of infected areas, and finally classification. Finally, in the third stage, the system sends a notification through an android or web application to inform users of any improper farm conditions and any detected infections. The overview of our proposed approach is shown in Fig. 3.1.

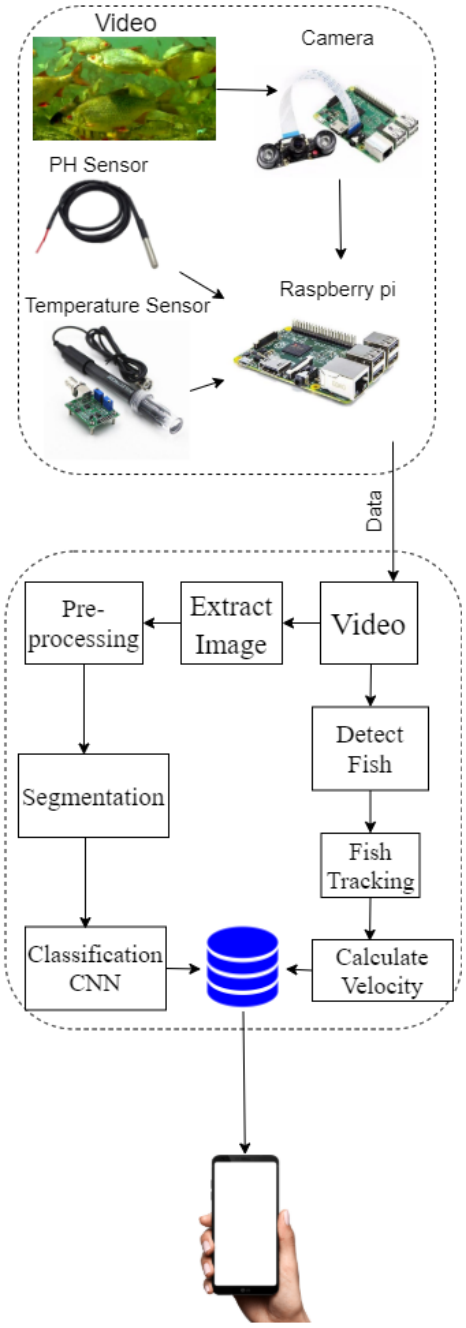


Figure 3.1: Proposed System Overview

3.2.1 Business Context

Fish diseases have a serious impact on the country's economy. Most experts in the fish industry currently have limited time and have to make a quick decision without a good diagnosis of the problem. The process of manually detecting fish diseases and fish abnormal behavior using human vision consumes a lot of time and limits the accuracy of detection. Experts also may face difficulties in vision due to fast fish movement and unclear water in earthen ponds that also cause limitations in diagnosis and tracking. Therefore, there is an urgent need for an automated system using computer vision to detect and reduce the impact of fish diseases. This automated system will save time for experts and provides more accurate results. Moreover, reduce system running costs by eliminating the need for continuous human monitoring and increasing behavior analysis accuracy by excluding the human subjectivity factor. Our proposed automated system also reduces the cost of fish production by controlling fish diseases as infectious diseases can cause multibillion-dollar loss annually.

3.3 General Description

3.3.1 Product Functions

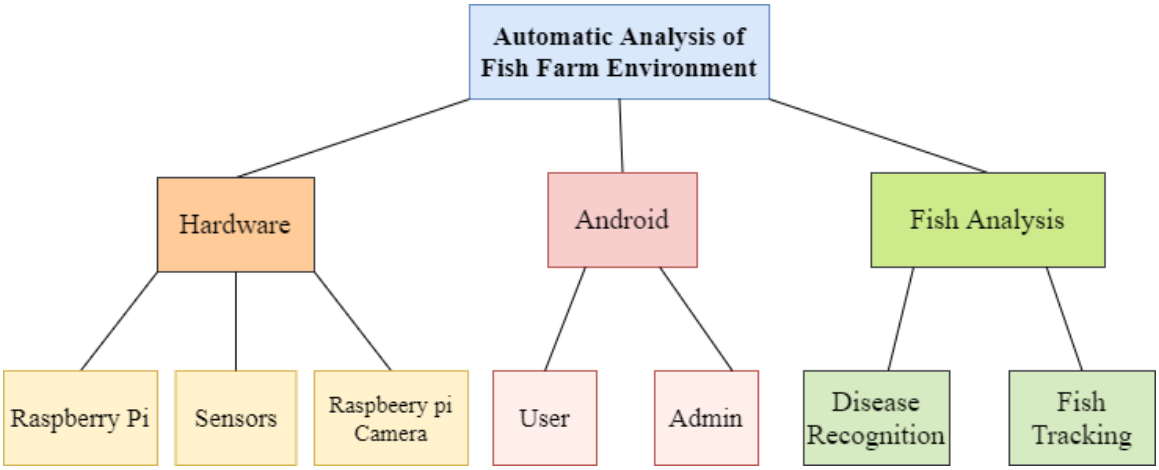


Figure 3.2: Proposed System Block Diagram

This part outlines all sections separately in details:

3.3.1.1 The Hardware Section

This section is composed of Raspberry pi hardware circuit which is connected to sensors and camera.

- Sensors: The sensors are responsible for examining water quality by measuring water temperature and rate of pH
- Raspberry pi: The Raspberry pi gets the sensor's measurements and acquired fish captures to send them to the PC for processing.
- Camera: The camera is responsible for capturing fish movements.

3.3.1.2 Android Section

This section is to inform users of any improper farm conditions and any detected infections. This section include user that interact with the system and admin which control all system's partitions.

Admin is responsible for:

- Sign in and Sign out.
- Manage Farm.
- Manage Hardware.
- Delete user account.
- Listing all users.
- Update his profile information.

User is responsible for:

- Sign in/up and Sign out.
- View Profile.
- Edit profile.
- View Readings.
- View his farm data.
- View Notifications.

3.3.1.3 Fish analysis

This part concerns tracking fish for detecting any abnormal behavior in farm environment and fish infections.

The infection detection part passes through the following stages:

1. The Pre-Processing Module
    - Convert to YCBCR color space.
    - Convert to XYZ color space.
    - Resizing.
2. Segmentation Module
    - Apply Gaussian distribution.
    - Calculate mean of Cb/Cr.
    - Calculate co-variance.
    - Apply texture features.
3. Classification Module

    It responsible for the classification of the input image as Fish disease/not Fish disease and diagnose which type of diseases.

    - CNN

    1. ResNet50
    2. ResNet101
    3. Alex-Net
    4. ResNet18
    5. VGG16
    6. VGG19
    7. mobilenetv2
    8. Xception
    9. Inceptionresnetv2
    10. Shufflenet
    11. Nasnetmobile
    12. Nasnetlarge
    13. Squeezenet
    14. Inceptionv3
    15. Densenet201
    16. Googlenet

The Tracking part passes through:

- Detect Fish in video.

- Convert bounding box to set of points.
- Track points.
- Calculate Velocity.

3.3.2 Similar Systems

Different approaches [8], [24], [9] [10] applied computer vision techniques to detect and identify fish diseases. Some approaches [8][9] were proposed to classify epizootic ulcerative syndrome (EUS) diseased fish. Color segmentation methodology is often applied on fish images to extract damaged skin [24]. Another approach [10] extracted infected regions and send notifications to fish farmers. Notification include diagnosed disease and the suggested treatment.

Another approaches[25] [26] [27] are introduced to analyze fish trajectories in videos. Trajectories are classified to normal and abnormal behaviors. The study of fish behavior is important to analyze the environmental conditions that may cause diseases in the future. Normal fish trajectories are first identified then abnormal ones are detected by applying filtering mechanisms [25][27].

An Approach [8] applied histogram equalization followed by edge detection for segmentation. Canny's edge detector achieved the best results, compared to other edge detectors. Features from Accelerated Segment Test (FAST)[17] outperformed other features like Histogram of gradient (HOG)[28] features when been applied on EUS infected images. Principal Component Analysis (PCA) [29], [9] was applied after FAST features to reduce feature vector dimensions. Neural Network[18] achieved better recognition results compared to K-Nearest Neighbor (k-NN) [30] when applied as a classifier.

K-Means clustering [31] was applied for segmenting diseased area in hue-saturation-value (HSV) color space of fish diseased images, based on the hue values. Morphological operations[32] were applied to compute the diseased area, as shown in equation 3.1.

$$A \bigcirc B = (A \ominus B) \oplus B \tag{3.1}$$

Where $\ominus$ and $\oplus$ represent erosion and dilation, respectively and B is the structuring element.

Another usage of morphological operations was noise removal and edge detection during preprocessing stage [10]. Morphological erosion and dilation operations were applied to delete noise as shown in equations 3.2 and 3.3. Diseased area were detected by discriminat-

ing small and large connected components.

$$Erosion : I \otimes S_E = X : S_E + X < I \tag{3.2}$$

$$Dilation : I \oplus S_D = I^c \ominus (-S_D)^C \tag{3.3}$$

where I is a source image, S_E and S_D are the structuring elements for erosion and dilation operations,respectively. Two types of features were extracted [10], the first is based on the polar coordinates and the second set is based on the geometrical features as defined in equations 3.4 and 3.5, respectively.

$$X = r\cos\theta, Y = r\sin\theta \tag{3.4}$$

Where $r$is the radial distance from the origin and θ is the counterclockwise angle with the x-xis, In terms of X, Y, r and θ the geometric features were calculated by

$$r = \sqrt{x^2 + y^2}, \theta = \tan^- 1(\frac{y}{x}) \tag{3.5}$$

Convolutional Neural Networks (CNN)[33], were applied for fish disease classification in another approach [14].Images were enhanced by applying Gaussian blurring[34] and morphological operations. Otsu's thresholding[35] and Pyramid Mean Shifting (PMS) [36], defined in equation 3.6, were then applied to enhance classification by CNN.

$$m(x) = \frac{\sum_{x_i \in N_n} K(x_i - x)x_i}{\sum_{x_i \in N_n} K(x_i - x)} \tag{3.6}$$

Approach [25] proposed a filtering mechanism like a cascade classifier [37] that is based on filtering mechanism. It define rules for normal trajectories. Any trajectories not satisfying the rules are considered abnormal behaviors.

Hierarchical approaches [26][27] were proposed to extract different types of features, like curvature scale space (CSS) [38], moment descriptors[39], velocity, acceleration, centered Distance Function (CDF) [38], Vicinity Features, Loop Features, Features Based on Normalized Size of Bounding Box. Affinity propagation (AP) [40] is applied for clustering method. AP perform better than K-means, mixture models and mean-shift clustering. Outlier detection [41] method is applied. It is presented in two types of outlier trajectories:

those located in small clusters and those that exist in dense clusters.

$$\theta_i = \frac{Y_{i+1} - Yi}{X_{i+1} - X_i} \tag{3.7}$$

3.3.3 User Characteristics

There are three types of users that interact with the system:

1. Owners of fish farms:
   - Must have basic knowledge in using Android mobile devices, how to use the device and apps.
2. Experts of fish domain:
   - Must have basic knowledge in using Android mobile devices, how to use the device and apps.
3. Admin:
   - Must be able to work with firebase and to manage the firebase through UI.
   - Must have knowledge in using Android mobile devices, how to use the device and apps.

3.3.4 User Problem Statement

Fish disease diagnosis suffers from some limitations that need a high level of expertise to be solved. Fish disease domain is affected by varying expertise of experts. Diagnosis differs based on expert skills. Experts may face some problems due to fast fish movement which causes tracking infected fish to be impossible by human vision. Poor quality of water in earthen ponds also causes limitations in diagnosis and tracking. Therefore, we proposed a system to detect fish diseases and analyze fish behavior automatically.

3.3.5 User Objectives

Applying image processing and computer vision techniques, vision can be improved, and tracking fish becomes easier. The user will only receive notification in their mobile phones, to know if there are any improper changes in fish farm environment or any detected infections.

3.3.6 General Constraints

There is no system that has no constrains, our system will have some constrains:

- Mobile application applicable for android mobile devices only.
- Device must be connected to the internet to receive notification.
- The connection under the water that the camera could face.

3.4 Functional Requirements

3.4.1 Admin/User

FID	FR1
Name	Register
Description	The user registers with his/her information to create an account
Input	Name , Email , Password and phone
Output	Confirmation Message and asks user to log in or error message upon validating the fields
Action	Checks validation of all fields and if so the data is entered in a new record in the database accordingly
Pre-condition	None
Post-condition	Database is updated with a new user account
Dependencies	None

FID	FR2
Name	Login
Description	User/Admin can login with his/her email and password into his/her account
Input	Email, Password
Output	The homepage is previewed and login successful message or error message upon validating the fields
Action	Checks validation of all fields and if so compares data entered to that in the database records
Pre-condition	User/Admin is already registered in the database
Post-condition	Redirected to the homepage
Dependencies	FR1

FID	FR3
Name	Sign Out
Description	User/Admin will sign out of his/her account
Input	None
Output	Logged out confirmation
Action	User/Admin will sign out
Pre-condition	Admin/User is signed in to his/her account
Post-condition	Admin/User is signed out
Dependencies	FR2

FID	FR4
Name	Deleting a user's Account
Description	The Admin deletes a user's account from the system
Input	user's ID
Output	Confirmation message or error message if something went wrong upon removing the user from the system
Action	Delete selected user's record from the database
Pre-condition	Desired user is already registered in the database
Post-condition	Desired user's record is removed from the database
Dependencies	FR1

FID	FR5
Name	listing all user's
Description	The Admin lists all the users found in the system
Input	None
Output	All users registered in the system and their information are previewed
Action	Retrieves information about the users registered in the system from the database
Pre-condition	At least one user is registered in the database
Post-condition	All users are previewed
Dependencies	FR1

FID	FR6
Name	View Readings
Description	user can view sensors readings of his fish farm
Input	None
Output	readings are displayed
Action	Retrieve the data from the database
Pre-condition	user is logged in to the system
Post-condition	Data is fetched from the database
Dependencies	FR2

FID	FR7
Name	Show Notification
Description	user can show notification being sent when there is any improper change in fish farm environment
Input	None
Output	Notification is previewed
Action	user show notification
Pre-condition	Sensors readings is checked
Post-condition	Notification is received
Dependencies	FR2

FID	FR8
Name	Adding farm
Description	The admin adds new farm to the system
Input	Farm Name/ID
Output	Confirmation message or error message if something went wrong upon validating the fields
Action	Checks validation of all fields and if so the data is entered in a new record in the database accordingly
Pre-condition	Desired farm is already added in the database
Post-condition	Database is updated with a new farm
Dependencies	None

FID	FR9
Name	Edit farm
Description	The admin edit farm details in the system
Input	Farm ID
Output	Confirmation message or error message if something went wrong upon editing the fields
Action	The information changed is taken and sent to the corresponding attribute in the farm's record in the database to be updated
Pre-condition	Desired farm is already added in the database
Post-condition	Database is updated with the new farm details
Dependencies	FR8

FID	FR10
Name	Deleting farm
Description	The Admin deletes farm from the system
Input	farm ID
Output	Confirmation message or error message if something went wrong upon removing farm from the system
Action	Delete selected farm record from the database
Pre-condition	Desired farm is already added in the database
Post-condition	Desired farm record is removed from the database
Dependencies	FR8

FID	FR11
Name	listing all farms
Description	The Admin lists all the farms found in the system
Input	None
Output	All farms added in the system and their information are previewed
Action	Retrieves information about the farms added in the system from the database
Pre-condition	At least one farm is added in the database
Post-condition	all farms are previewed
Dependencies	FR8

FID	FR12
Name	Adding hardware
Description	The admin adds new hardware to the system
Input	hardware Name/ID
Output	Confirmation message or error message if something went wrong upon validating the fields
Action	Checks validation of all fields and if so the data is entered in a new record in the database accordingly
Pre-condition	Desired hardware is already added in the database
Post-condition	Database is updated with a new hardware
Dependencies	None

FID	FR13
Name	Edit hardware
Description	The admin edit hardware details in the system
Input	hardware ID
Output	Confirmation message or error message if something went wrong upon editing the fields
Action	The information changed is taken and sent to the corresponding attribute in the hardware record in the database to be updated
Pre-condition	Desired hardware is already added in the database
Post-condition	Database is updated with the new hardware details
Dependencies	FR12

FID	FR14
Name	Deleting hardware
Description	The Admin deletes hardware from the system
Input	hardware ID
Output	Confirmation message or error message if something went wrong upon removing farm from the system
Action	Delete selected farm record from the database
Pre-condition	Desired hardware is already added in the database
Post-condition	Desired hardware record is removed from the database
Dependencies	FR12

FID	FR15
Name	listing all hardware
Description	The Admin lists all the hardware found in the system
Input	None
Output	All hardware added in the system and their information are previewed
Action	Retrieves information about the hardware added in the system from the database
Pre-condition	At least one hardware is added in the database
Post-condition	all hardware are previewed
Dependencies	FR12

3.4.2 Hardware

FID	FR16
Name	Take video
Description	This function is used to take video from camera
Input	None
Output	Video
Action	Raspberry pi takes video from camera
Pre-condition	Raspberry pi is connected to camera
Post-condition	video is taken from camera
Dependencies	None

FID	FR17
Name	Get PH sensor reading
Description	Get Reading from PH sensor and pass it in Raspberry pi
Input	Readings from PH sensor
Output	Collection of PH sensor data
Action	sending readings to Raspberry pi
Pre-condition	sensors is connected to raspberry pi
Post-condition	Readings passed to Raspberry pi
Dependencies	None

FID	FR18
Name	Get Temperature sensor reading
Description	Get Reading from Temperature sensor and pass it in Raspberry pi
Input	Readings from Temperature sensor
Output	Collection of Temperature sensor data
Action	sending readings to Raspberry pi
Pre-condition	sensors is connected to raspberry pi
Post-condition	Readings passed to Raspberry pi
Dependencies	None

FID	FR19
Name	Send Notification
Description	Notification will be sent to the user through mobile application to notify him/her that he/she has improper changes in farm environment or detected infections
Input	Boolean choice
Output	Notification is sent
Action	Check sensors values and send notification to the user accordingly.
Pre-condition	PH and Temperature values is sent.
Post-condition	User received notification
Dependencies	FR17, FR18

FID	FR20
Name	Upload PH sensor readings to firebase
Description	This function is for collecting PH sensor readings from raspberry pi and uploading it to firebase
Input	Readings of PH
Output	PH sensor data are uploaded to firebase
Action	Sending readings from raspberry pi to firebase
Pre-condition	Sensors reading are available in raspberry pi
Post-condition	Data are uploaded to the firebase
Dependencies	FR17, FR18

FID	FR21
Name	Upload temperature sensor to firebase
Description	This function is for collecting temperature sensor readings and uploading it to firebase
Input	Readings of temperature sensors
Output	temperature sensor data are uploaded to firebase
Action	Sending readings from raspberry pi to firebase
Pre-condition	Sensors reading are available in raspberry pi
Post-condition	Data is uploaded to the firebase
Dependencies	FR17, FR18

FID	FR22
Name	Listing all sensors readings
Description	This function is for listing all sensors readings found in the system
Input	Boolean choice
Output	All sensors readings registered in the system are previewed
Action	Retrieves sensors readings registered in the system from the database.
Pre-condition	At least one sensor reading is registered in the database
Post-condition	None.
Dependencies	FR20, FR21

3.4.3 Tracking and disease detection

FID	FR23
Name	Data augmentation
Description	This function is used to increase the diversity of images that is available in data-set for training models
Input	RGB image
Output	Size of training data-set is expanded
Action	Choose an image to apply data augmentation on it.
Pre-condition	RGB image
Post-condition	Collection of RGB images
Dependencies	None

FID	FR24
Name	Create XML file
Description	This XML file contain data which is extracted from video in the dataset to be compared with frames taken from video.
Input	image
Output	XML file
Action	compare data in XML with given image
Pre-condition	Images from the dataset
Post-condition	XML Created
Dependencies	None

FID	FR25
Name	Draw border
Description	This function is used to add border on detected fish.
Input	image
Output	Detected image by drawing border
Action	Add border when fish is detected.
Pre-condition	snapshots from video
Post-condition	images of detected fish
Dependencies	FR24

FID	FR26
Name	Count Number of detected fish
Description	This function is used to count borders of detected fish.
Input	snapshots from video
Output	Number
Action	count number of detected fish.
Pre-condition	border is added on detected fish
Post-condition	Number of border is added
Dependencies	FR24, FR25

FID	FR27
Name	Video segmentation
Description	This function is used divide video into frames then Convert frames to YCBCR then change values of YCBCR
Input	video
Output	video after applying segmentation
Action	Convert frames to YCBCR, then change values of YCBCR
Pre-condition	video
Post-condition	video after segmentation
Dependencies	FR16, FR31

FID	FR28
Name	Point Tracker
Description	This function is used to track a set of points.
Input	Video frame
Output	Tracked points and reliability of track
Action	This function is used to apply Kanade-Lucas-Tomasi (KLT) feature-tracking algorithm to track a set of points.
Pre-condition	Videos
Post-condition	Set of points
Dependencies	None

FID	FR29
Name	Velocity
Description	This function is used to calculate velocity
Input	image detect corners of fish
Output	Rate of position and speed
Action	calculate velocity and acceleration
Pre-condition	fish is detected
Post-condition	velocity and acceleration is calculated
Dependencies	FR27, FR28

FID	FR30
Name	Estimate Geometric Transformation
Description	This function is applied to find the transformation matrix
Input	MatchedPoint1 and matchedPoints2
Output	Geometric transformation
Action	This function is applied to find the transformation matrix which maps the greatest number of point pairs between two images.
Pre-condition	points location of the frame
Post-condition	Geometric transformation matrix
Dependencies	FR28

FID	FR31
Name	Detect Minimum Eigen Features
Description	This function detects corners and return cornerPoints. object.
Input	Image
Output	Points
Action	This function detects corners by applying minimum eigenvalue algorithm and return cornerPoints object.
Pre-condition	points location of the frame
Post-condition	CornerPoints
Dependencies	FR28

FID	FR32
Name	Transform Points Forward
Description	This function applies the forward transformation of 2-D geometric transformation
Input	Geometric transformation and x,y,z-coordinates of points to be transformed and Coordinates of points to be transformed
Output	x,y,z-coordinates of points after transformed and Coordinates of points after transformed
Action	It apply the forward transformation of 2-D geometric transformation
Pre-condition	Geometric Transformation Matrix
Post-condition	Reshape of bounding box
Dependencies	FR30

FID	FR33
Name	Histogram Based Tracker
Description	This function is used to identify the tracked object.
Input	Video frame
Output	Bounding box[x y width height] and Orientation angle
Action	It is applied to identify the tracked object.
Pre-condition	Fish is detected
Post-condition	Histogram tracker
Dependencies	FR25

3.4.4 Image Pre-processing

FID	FR34
Name	Convert to YCBCR
Description	Converts RGB image to YCBCR
Input	RGB image
Output	YCBCR image
Action	Check that image is in RGB color space
Pre-condition	Acquired from video frame
Post-condition	Image is converted into YCBCR
Dependencies	FR23

FID	FR35
Name	Convert to XYZ
Description	Converts RGB image to XYZ
Input	RGB image
Output	XYZ image
Action	Check that image is in RGB color space
Pre-condition	Acquired from video frame
Post-condition	Image is converted into XYZ
Dependencies	FR23

FID	FR36
Name	Resize image
Description	Image will be resized to cover the part that will be classified
Input	RGB image
Output	image is resized
Action	Check that image is uploaded
Pre-condition	Acquired from video frame
Post-condition	Image is resized.
Dependencies	FR23

3.4.5 Image Segmentation

FID	FR37
Name	Apply Gaussian distribution of ycbcr
Description	It applies the predefined Gaussian filter on the image
Input	YCBCR image
Output	segmented image
Action	It applies the gaussian filter on the image after applying the low-pass filter
Pre-condition	The image before applying the Gaussian filter
Post-condition	The image after applying the Gaussian filter
Dependencies	FR34

FID	FR38
Name	Calculate Mean of CB/CR
Description	It Calculates the Mean of CB/CR
Input	YCBCR image
Output	Image Mean
Action	Takes the image and calculate the Average of the Pixels.
Pre-condition	Image's mean is not calculated
Post-condition	Image's mean is calculated
Dependencies	FR34

FID	FR39
Name	Calculate Co-variance of ycbcr
Description	It Calculates the co-variance.
Input	YCBCR image
Output	Image with co-variance value
Action	Takes the image and calculate the co-variance
Pre-condition	Image's co-variance is not calculated
Post-condition	Image's co-variance is calculated
Dependencies	FR34

FID	FR40
Name	Calculate Mean of X/Y
Description	It Calculates the Mean of X/Y
Input	XYZ image
Output	Image X/Y Mean
Action	Takes the image and calculate the mean X/Y of the Pixels
Pre-condition	Image's mean is not calculated
Post-condition	Image's mean is calculated
Dependencies	FR35

FID	FR41
Name	Calculate Co-variance of XYZ
Description	It Calculates the co-variance.
Input	XYZ image
Output	Image with co-variance value
Action	Takes the image and calculate the co-variance
Pre-condition	Image's co-variance is not calculated
Post-condition	Image's co-variance is calculated
Dependencies	FR35

FID	FR42
Name	Apply Gaussian distribution of xyz
Description	It applies the predefined Gaussian filter on the image
Input	XYZ image
Output	segmented image
Action	It applies the gaussian filter on the image after applying the low-pass filter
Pre-condition	The image before applying the Gaussian filter
Post-condition	The image after applying the Gaussian filter
Dependencies	FR40,FR41

FID	FR43
Name	Adaptive threshold
Description	It applies the threshold on Gaussian distribution model result
Input	Result from Gaussian distribution
Output	Binary image
Action	It applies adaptive threshold on Gaussian distribution model result
Pre-condition	Gaussian distribution is applied
Post-condition	The image after adaptive threshold
Dependencies	FR42

FID	FR44
Name	Crop
Description	It crops diseased area
Input	Result from adaptive threshold and test image
Output	Crops
Action	It crops diseased pixels
Pre-condition	Adaptive threshold is applied
Post-condition	Image is cropped
Dependencies	FR43,FR43

FID	FR45
Name	Convert to gray scale
Description	Converts RGB image to gray scale
Input	RGB image
Output	Gray scale image
Action	Check that image is in RGB color space
Pre-condition	Acquired from video frame
Post-condition	Image is converted into gray scale
Dependencies	FR23

FID	FR46
Name	GLDM
Description	It gets the probability density function
Input	Gray scale image
Output	Probability density values
Action	It gets the probability density
Pre-condition	Grey scale image
Post-condition	Probability density values
Dependencies	FR45

3.4.6 Classification

FID	FR47
Name	CNN Classifier
Description	this function is used to train data and integrate with it with features to classify new inputs of images
Input	training features
Output	Accuracy results
Action	Training features are mentioned including functions for preprocess of images
Pre-condition	Testing and training available but not calculated with each other
Post-condition	Training and testing compared with each other then the disease is classified
Dependencies	FR44

3.5 Interface Requirements

3.5.1 User Interfaces

The system user interface is designed to be simple enough and allow minimal interaction, as shown in figure 3.3 and 3.4.

3.5.1.1 GUI

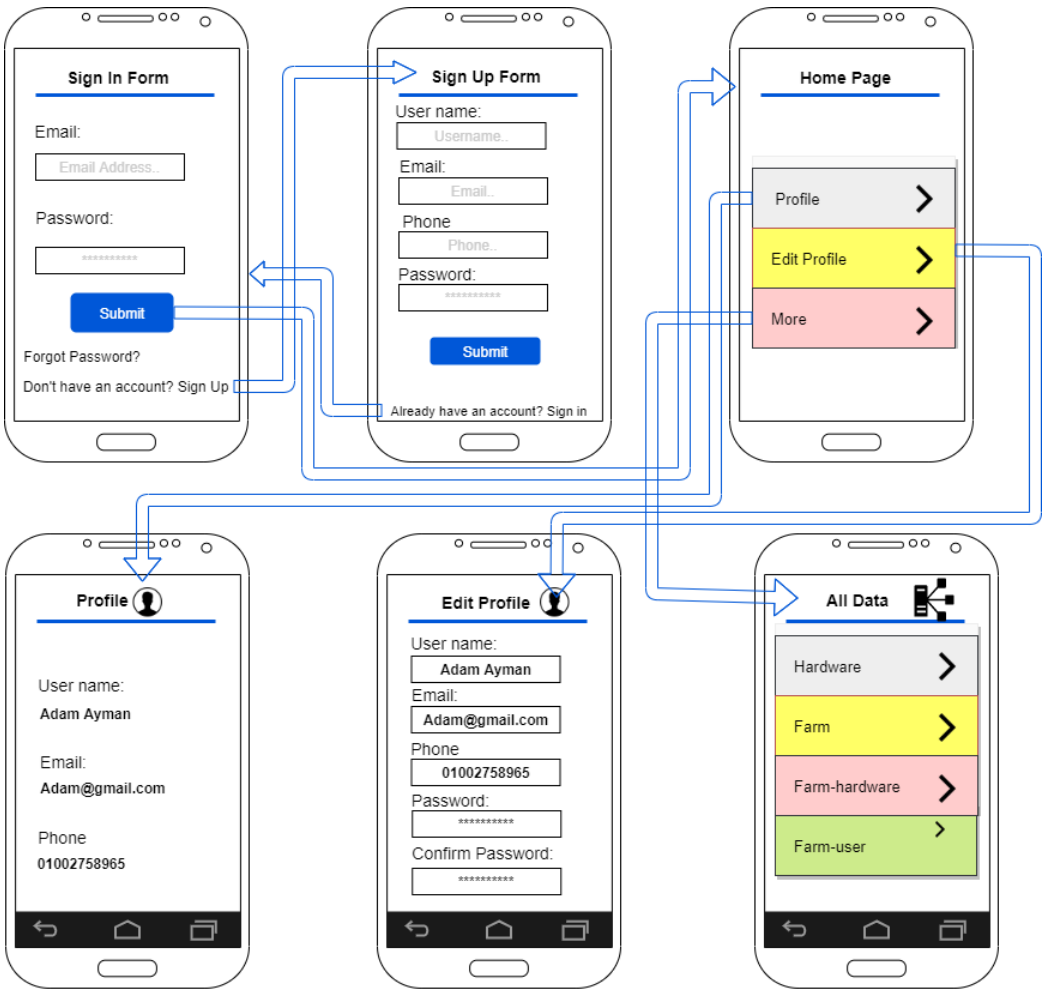


Figure 3.3: Admin's Mobile Application Wireframe

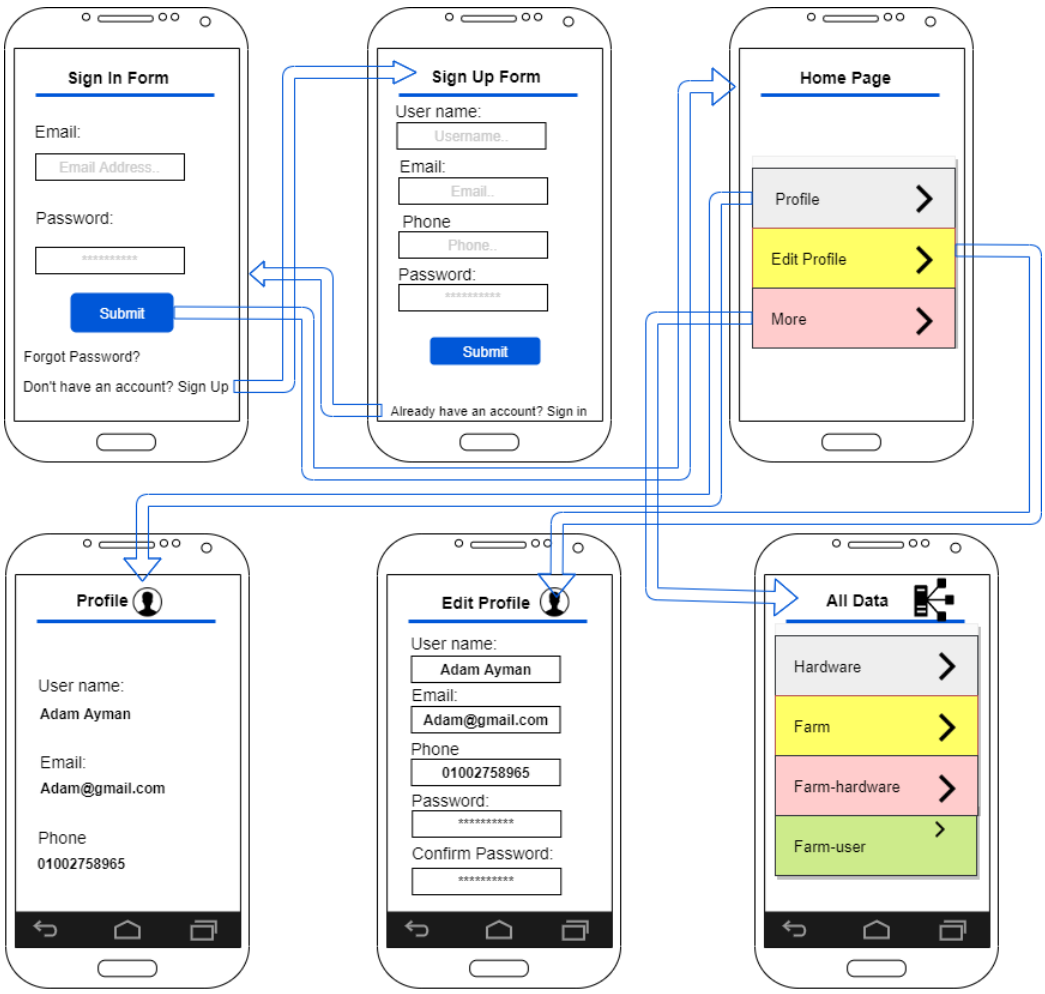


Figure 3.4: User's Mobile Application Wireframe

3.6 Performance Requirements

- The system must be able to handle large training datasets to ensure model accuracy.
- The time for sending notification after detecting any abnormal behavior must be short.
- No performance needed in the mobile, the mobile will only receive notification to notify if there is any improper change in fish farm environment.

3.7 Design Constraints

- This system needs to be user friendly to ease the process if the user lack of professional computer skills.
- Any smart mobile device that include the android operating system and must have the connection with the internet to deal with the real-time data transfer.
- The camera resolution should not be less than HD 1080.

3.8 Other non-functional attributes

3.8.1 Security

Users' passwords must be hashed in the database. Users of our system shall authenticate themselves using their username and password. Also, personal information about the users such as mobile numbers and passwords for instance must be protected.

3.8.2 Reliability

The system being developed has to be reliable in its operation. If the internet is disconnected, it saves the readings data to the database to prevent any loss of data. The user should be able to trust that the system should be reliable enough which does not cause failure or crash.

3.8.3 Maintainability

The system could be improved by different developers so ease of system maintainability is important, it should be easy to extend thought the implementation of MVC design pattern

and using naming convention which eases the use of functions and understanding their purpose. MVC design pattern divides the system into three modules which are Model, View, and controller, it simply separates handling of the data from how the interface appears to the user and the intermediate communicator between both of them.

3.8.4 Performance and speed

This system will do all the processing parts on the machine so no performance needed on the mobile. Also, the system is automatically clearing all the readings every 3 months to free up storage which might have an impact on mobile performance. The system must be interactive and the delays involved must be reduced. Detection and classification must have no delays.

3.9 Preliminary object-oriented domain analysis

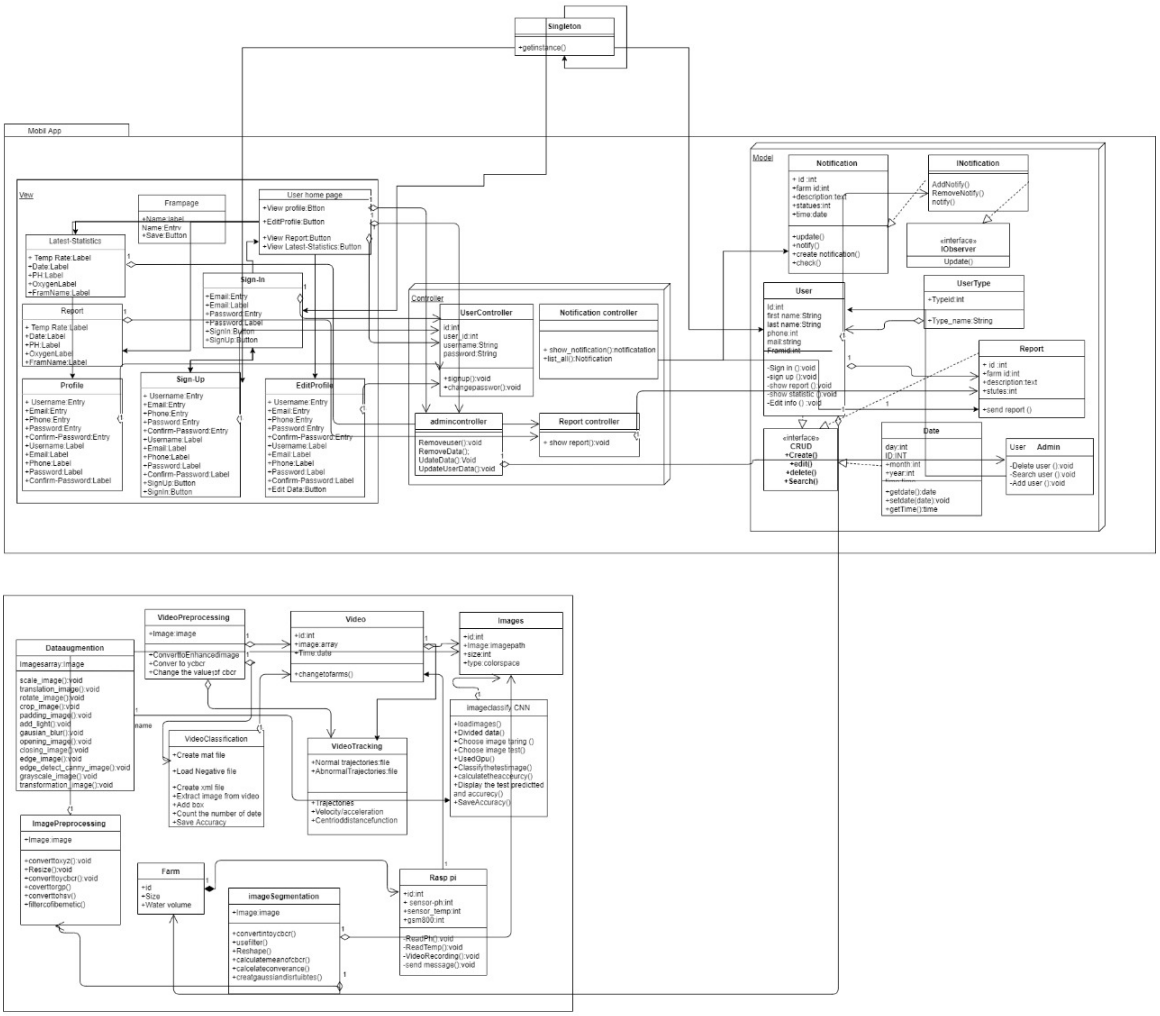


Figure 3.5: System Class Diagram

3.10 Operational Scenarios

Tabular description of the Search for user use case

Actors: Admin

Description: Admin can search for users that are saved on the system.

Data: User id.

Response: All user's are displayed.

Comments: Admin must be logged in.

Tabular description of the Create user account use case

Actors: Admin.

Description: Admin will be able to create accounts for new user.

Data: user info (Name, telephone , password).

Response: Confirmation that account has been created.

Comments: Admin must be signed in.

Tabular description of the Delete user account use case

Actors: Admin.

Description: Admin will be able to delete user accounts.

Data: user id.

Response: Confirmation that account has been deleted.

Comments: Admin must be signed in.

Tabular description of the Update user info use case

Actors: Admin.

Description: Admin will be able to update user's data.

Data: User ID.

Response: User Profile displayed for editing.

Comments: Admin must be signed in.

Tabular description of the Convert image to YCBCR use case
Actors: Pre-processing System.
Description: Test image is converted from RGB to YCBCR.
Data: RGB Test Image.
Response: Image is converted.
Comments: None.

Tabular description of the Convert image to XYZ use case
Actors: Pre-processing System.
Description: Test image is converted from RGB to XYZ.
Data: XYZ Test Image.
Response: Image is converted.
Comments: None.

Tabular description of the Perform Gaussian Distribution use case
Actors: Segmentation System.
Description: Perform Gaussian Distribution to image.
Data: YCBCR image to segment.
Response: segmented image.
Comments None.

Tabular description of calculating mean of CB/CR use case
Actors: Segmentation System.
Description: calculate the Average of the Pixels.
Data: YCBCR image to segment.
Response: Mean is calculated.
Comments None.

Tabular description of the View Report use case

Actors: User.

Description: User can view report.

Data: None.

Response: Report data are displayed.

Comments: user must be logged in.

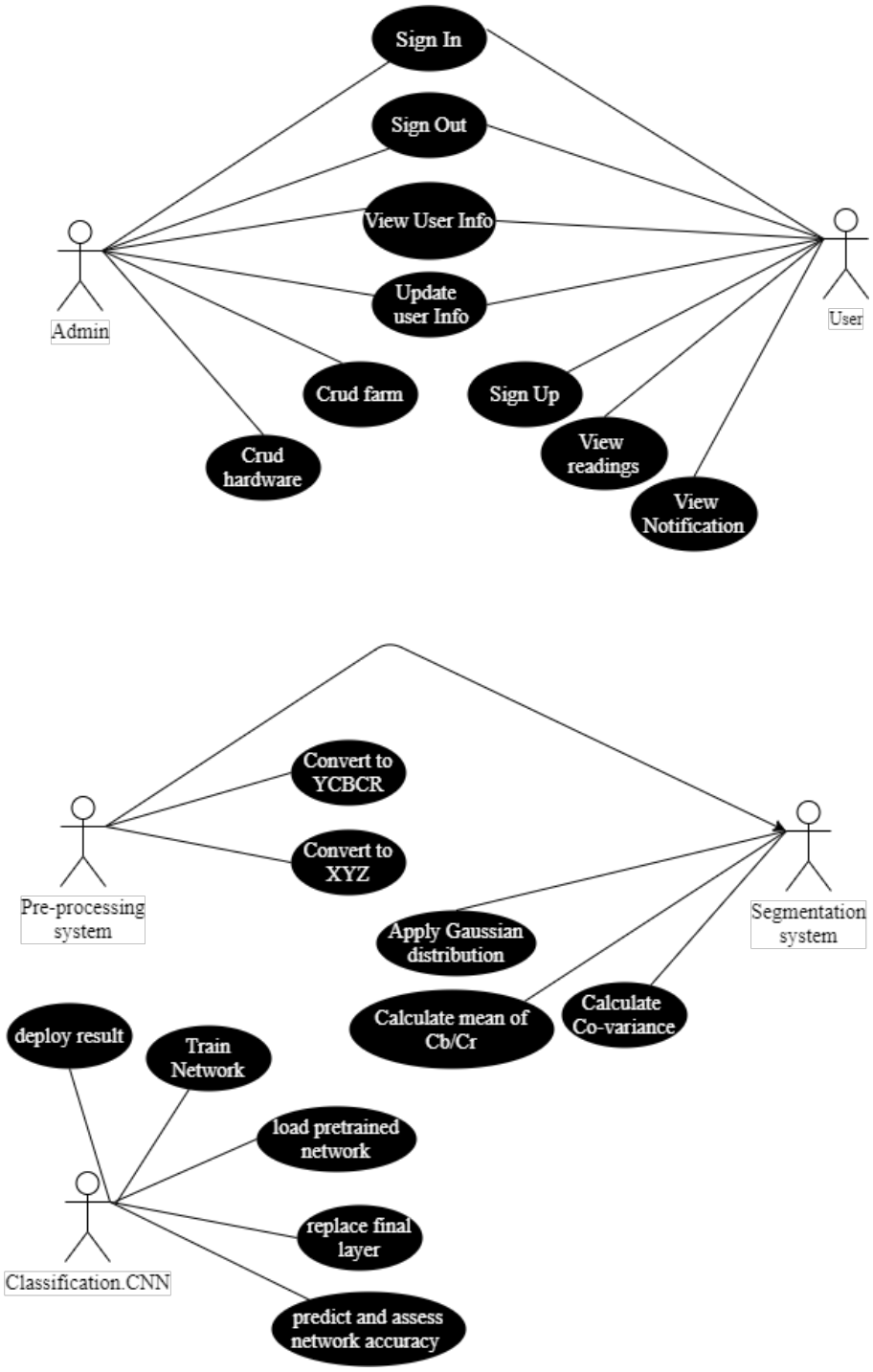


Figure 3.6: System Use Case

Chapter 4

Software Design Document

4.1 Introduction

4.1.1 The Purpose

The purpose of this software design document is to describe the architecture and system design for our Automatic Analysis of Fish Farm Environment. Our system mainly detects and diagnoses fish diseases in fish farms automatically and examine water quality. We provide a full illustration for each stage and development process. This document will explain in detail, the components of the system represented in the sequence diagram. The implementation of the project and its development will be shown in the class diagram. It illustrates the system components and how they interact with each other.

4.1.2 Scope

This software design document targets owners of fish farms and experts in the fish farm domain. Our application will help them in saving much more time rather than manual detection. Our proposed project is an automated system to diagnose fish diseases. The main objective is the detection and improvement of classification accuracy of fish diseases and fish abnormal behaviors. Our system is a recognition system that performs pre-processing, segmentation, and classification of the image.

4.1.3 Overview

Our system is created to performs detection and identification of fish diseases and analyze fish abnormal behavior. Its goal is to prevent and control the spreading of diseases. Our system will work on three different types of fish diseases. These diseases are Epizootic ulcerative syndrome (EUS), Ichthyophthirius (Ich), and Columnaris. Our system will consist of a mobile application that will read the data that will be collected from sensors (PH and Temperature) and receive a notification if any improper change happened. The farm owner will be provided by an interface through which he could monitor the farm environment.

4.2 System Overview

Our proposed system aims to analyze fish farm environment by detecting fish diseases and fish behavior. Fish behavior helps in the expectation of diseased fish. The proposed system is presented in three consequent stages. During the first stage, the system contains raspberry pi 3, along with a camera and other sensors. The sensors that are used are temperature and PH to measure underwater temperature and rate of pH. while fish captures are acquired by the camera. while fish captures are acquired by the camera. Raspberry pi 3 gets the sensor's measurements and acquired fish captures and passes the data to be stored in a computer system for processing. In the second stage, the processing part concerns tracking fish for detecting any abnormal behavior in farm environment and fish infections. Infection detection starts by pre-processing, then segmentation of infected areas, and finally classification. The classification of fish diseases are done by applying convolution neural networks (CNN). Tracking of fish movement is then applied by the Kanade-Lucas-Tomasi (KLT) algorithm. Finally, the system sends a notification through an android or web application to inform users of any improper farm conditions and any detected infections. The overview of our proposed approach is shown in figure. 4.1.

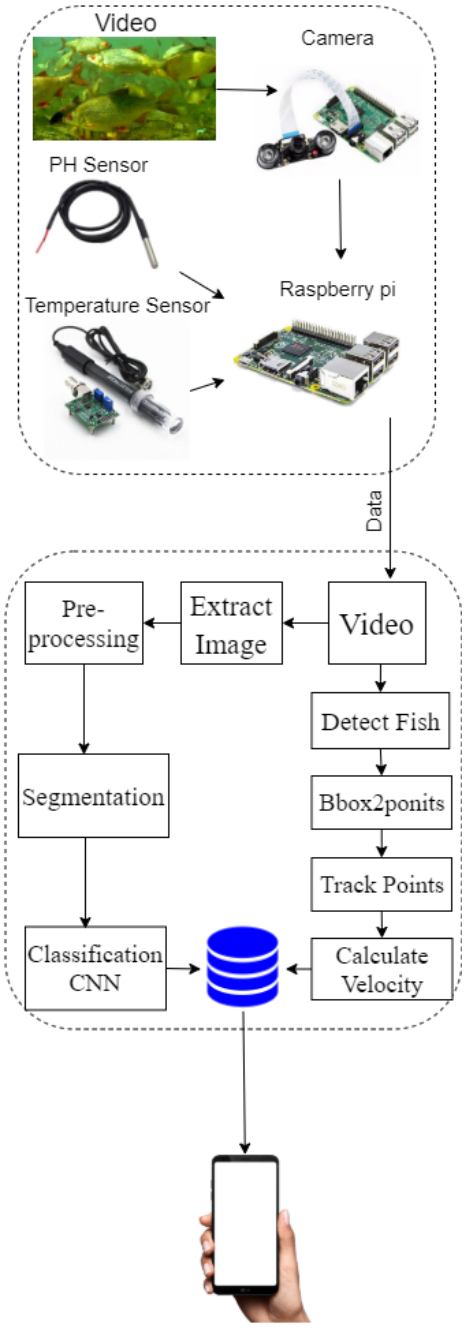


Figure 4.1: System Overview

4.3 System Architecture

4.3.1 Architectural Design

The System is designed in MVC architecture as shown in fig 4.2.

4.3.1.1 View

It is responsible for the presentation of data and representing the User Interface (UI). It consists of some forms to help the user to navigate through the system and the functionalities of the system. such as:

- Sign up: This interface allows new users to sign up to the application, by entering their information for the database to store.
- Sign in: This interface is meant for the users and admins to log in the application.
- Home Page: This interface that manages all the modules inside our system. There are 4 modules that the user may interact with.
    1. Edit Profile: This interface allows each user to edit his data.
    2. Profile: This interface allows users to show all his personal information.
    3. Readings: This interface is for listing all the daily sensor readings.

4.3.1.2 Model

- Libraries

    1. Offset-Date-Time is an immutable representation of a date-time with an offset. This class stores all date and time fields, to a precision of nanoseconds, used to model data in simpler applications. This class may be used when modeling date-time concepts in more detail.
    2. Firebase Authentication: Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook, and Twitter, and more. It uses to help to create a new

sign method that takes in an email address and password, validates them, and then signs a user in with the signInWithEmailAndPassword method.

- Core

  DB Manager: The Database Manager takes control by notifying the server with a recent update like if new reading data is arriving, the server also uses it when finishing processing by sending a notification to the admin if there is an infection in the farm. Or storing the reading data from the sensors.

4.3.1.3 Controller

It is responsible for binding the view and model. It takes input from the view and sends it to the database and retrieves data from the model and sends it to the view to be previewed there. The handling of all the data coming from the input hardware devices including PH readings coming from the PH sensor, temperature reading coming from the temperature sensor. Some of the controllers we are having; admin Controller that is responsible for handling interactions made within admin views, user controller that is responsible for handling interactions made by fish farm owner and the notification controller that is responsible for sending notifications to the user when there is any improper change in a fish farm environment.

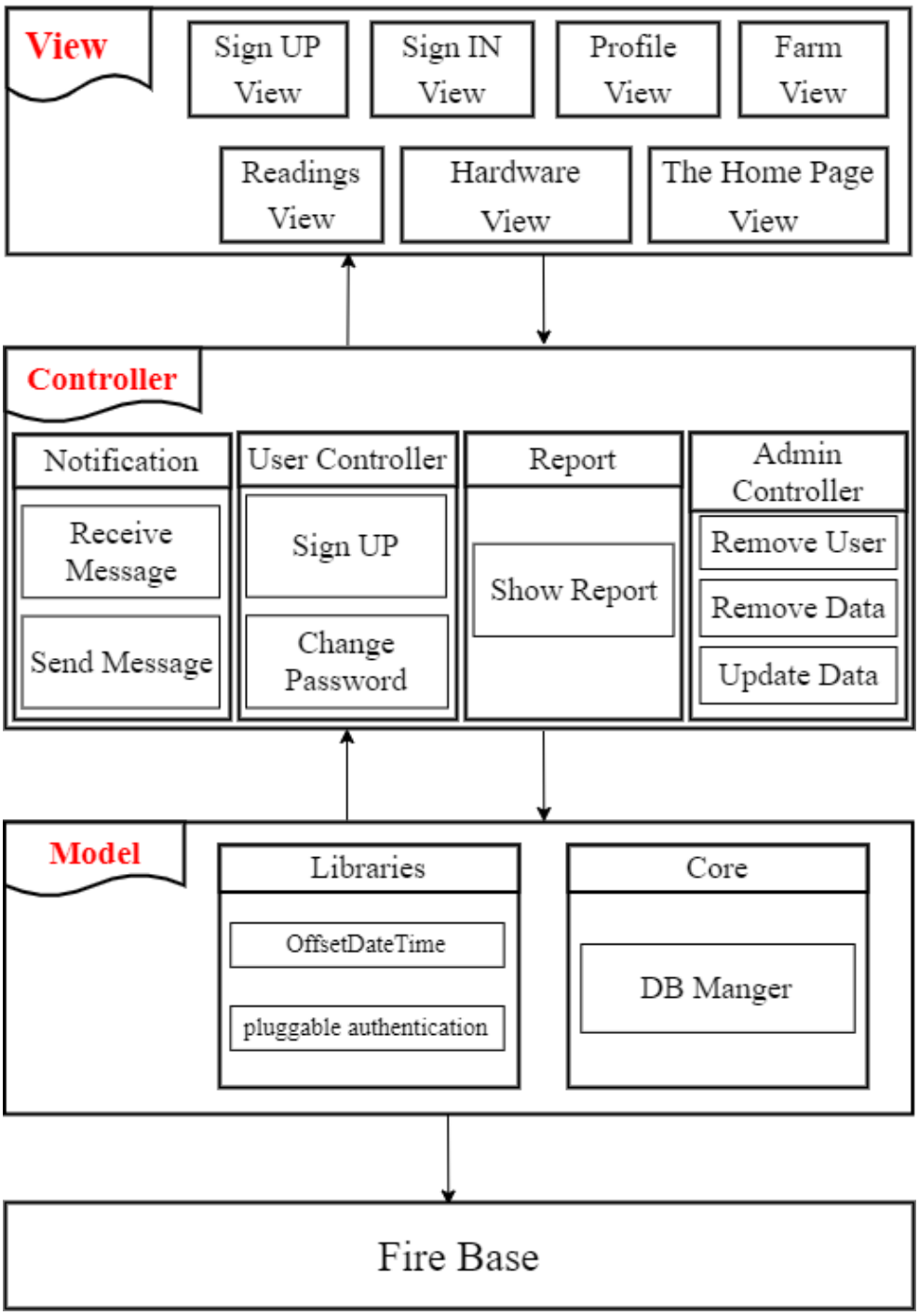


Figure 4.2: MVC Architectural Design

4.3.2 Decomposition Description

4.3.2.1 Class Diagram

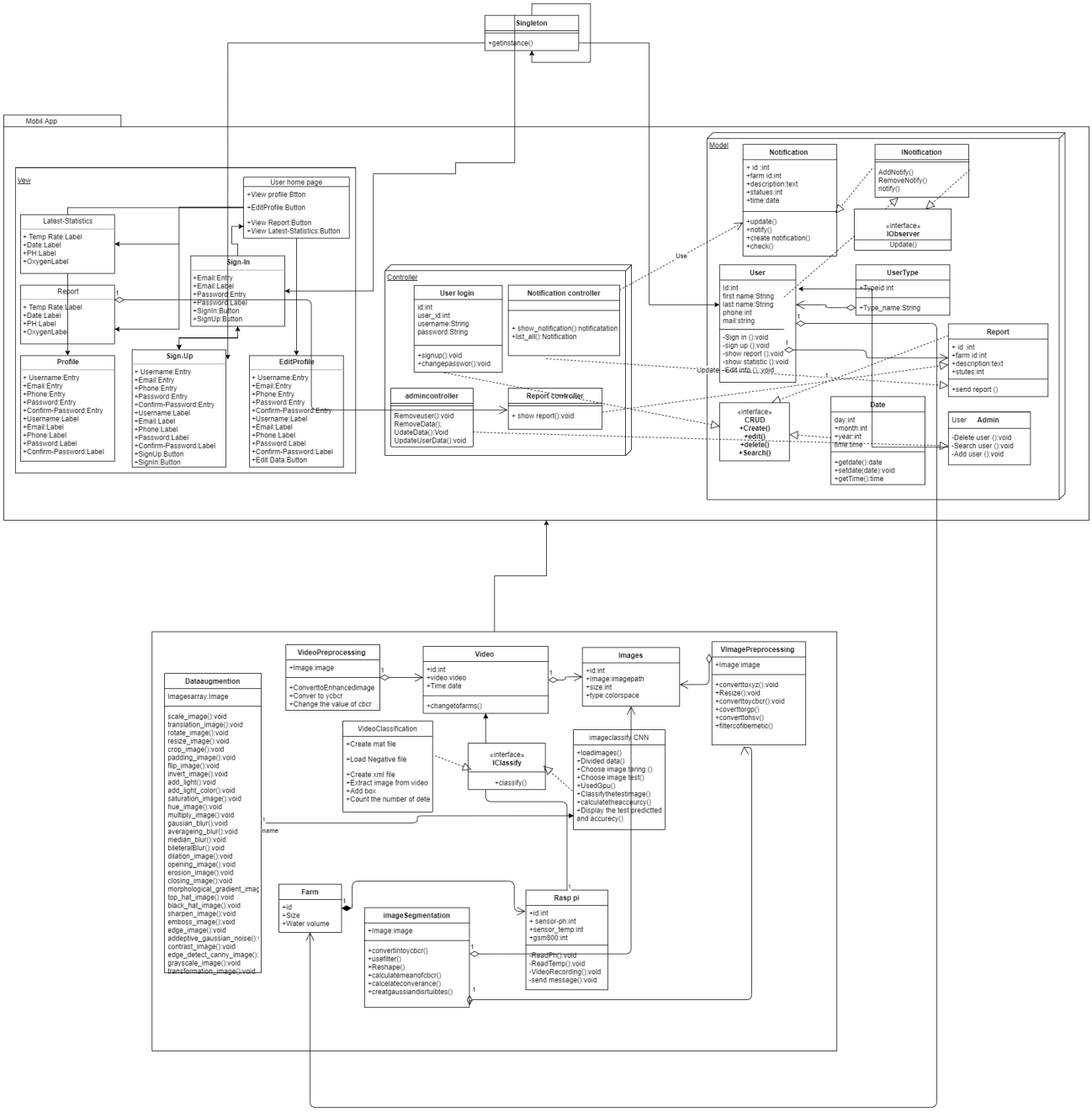


Figure 4.3: Class diagram

Snaps of our class diagram:

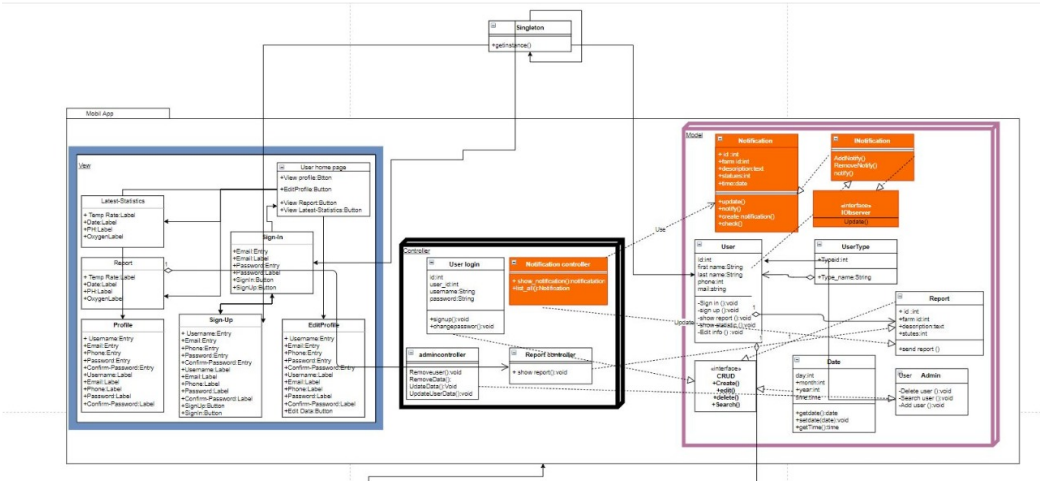
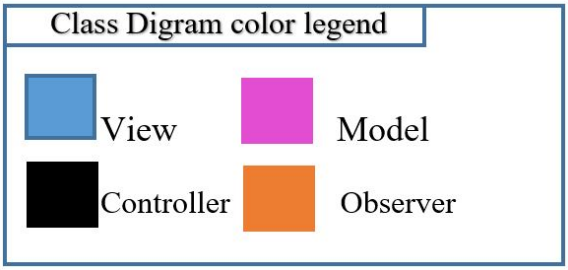


Figure 4.4: MVC Class Diagram



- This class diagram implements the MVC design pattern which is used to separate the application's view from it's model by interacting via the controller.
- Model: Represents an object carrying data. It can also have logic to update the controller if its data changes.
- View: Represents the visualization of the data that the model contains. It is responsible for the presentation of data and representing the user interface (UI).
- Controller: Acts on both model and view. It is responsible for binding the view and model, it controls the data flow into the model object and updates the view whenever data changes. It keeps view and model separate.
- In the singleton design pattern, the class extends itself. It is one of the simplest design patterns. In our case, it is responsible for the database connection.

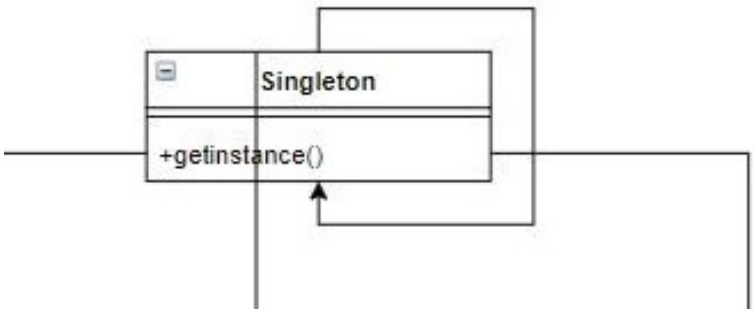


Figure 4.5: Singleton Design Pattern

- The Observer Design Pattern: Defines a one-to-many relationship so that when one object changes state, the others are notified and updated automatically. Firebase is being used in this case for handling the notifications for the farm owners' mobile application.

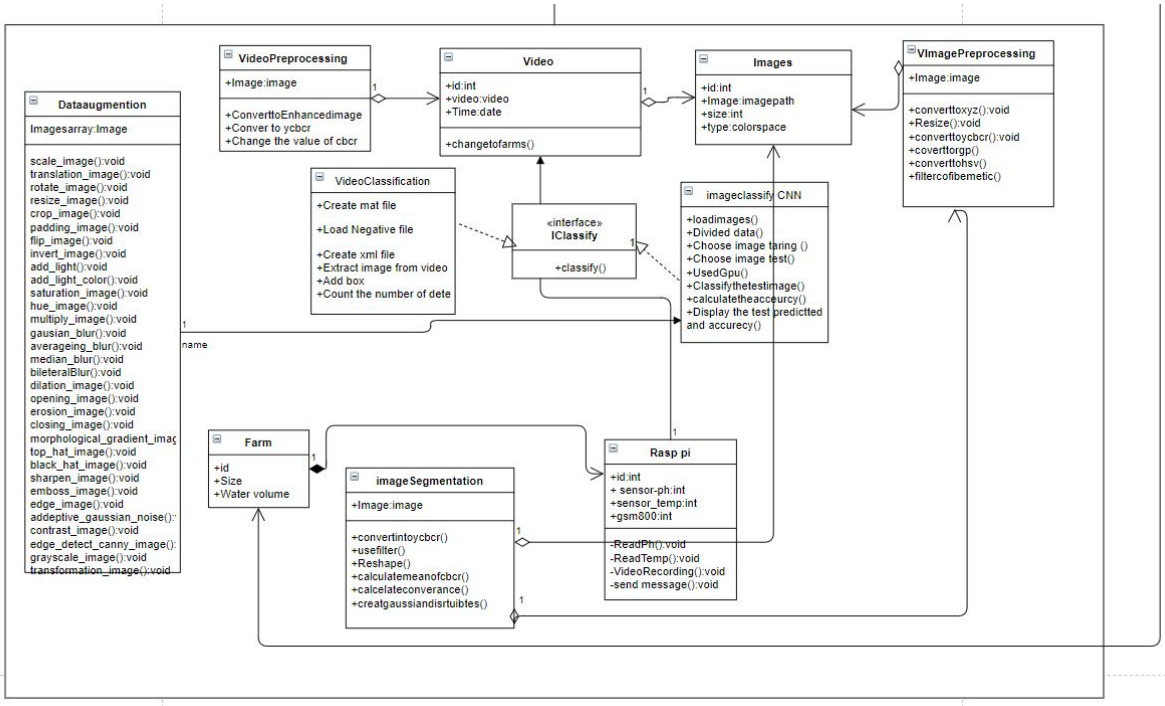


Figure 4.6: Computer system

- Image Pre-processing: contain ID, images, this table used to apply different color spaces on input images during the pre-processing phase
- Image: This table contains ID, Name, path, size to be able to differ between the different images.
- Video Pre-processing: contain ID, images, this table used to apply different color spaces on input images during the pre-processing phase
- Video: This table contains ID, Video, date to be able to differ between the different Videos.
- Image segmentation: Contain ID, Image.this table used to convert from RGB color space to YCbCr
- Data augmentation: Contain ID, Image.this table used to increase the number of our dataset
- Image Classification: Contain ID, Image.This table used to detected infections.

4.3.2.2 Activity Diagram

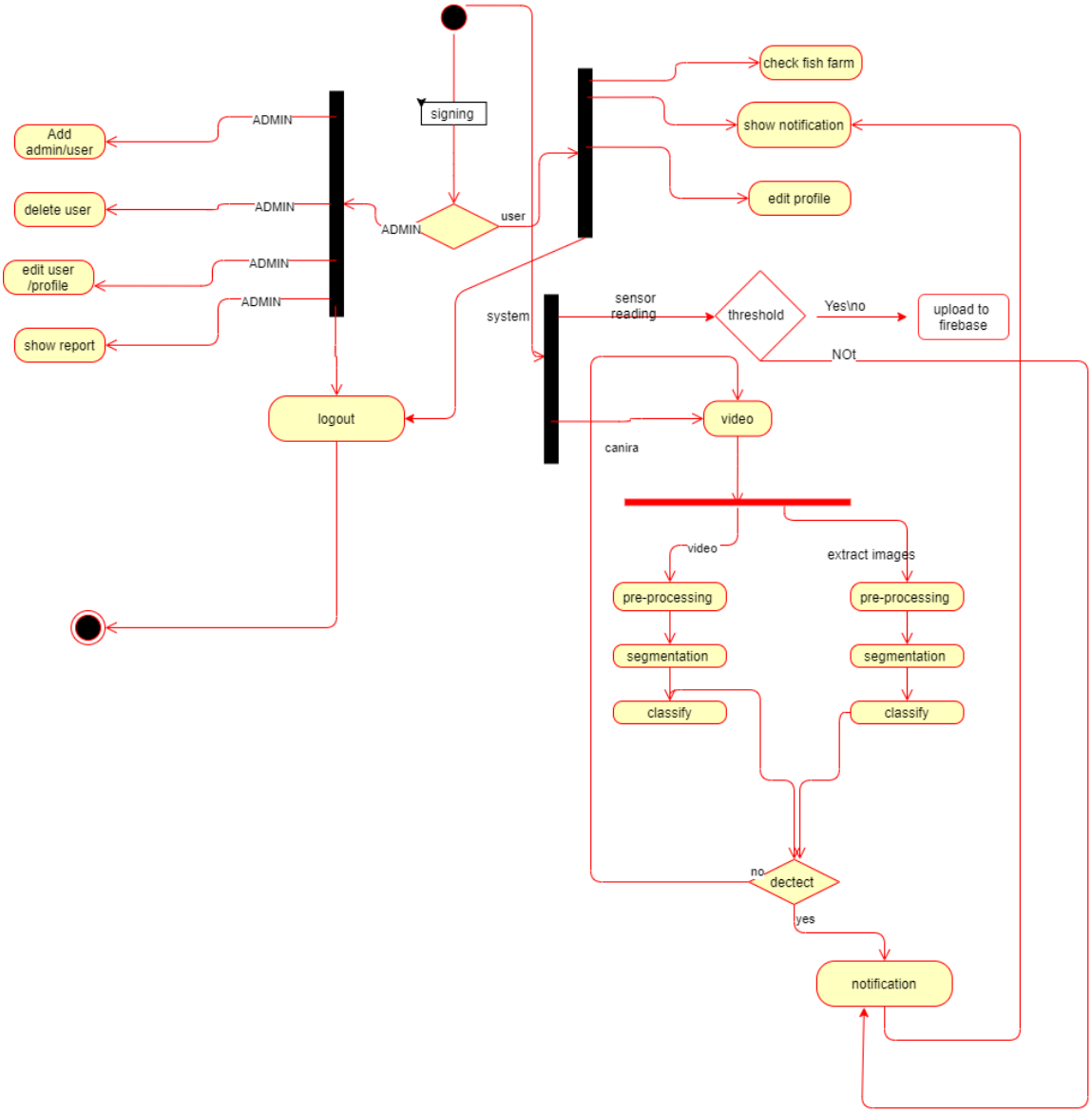


Figure 4.7: Activity diagram

4.3.2.3 Sequence Diagrams

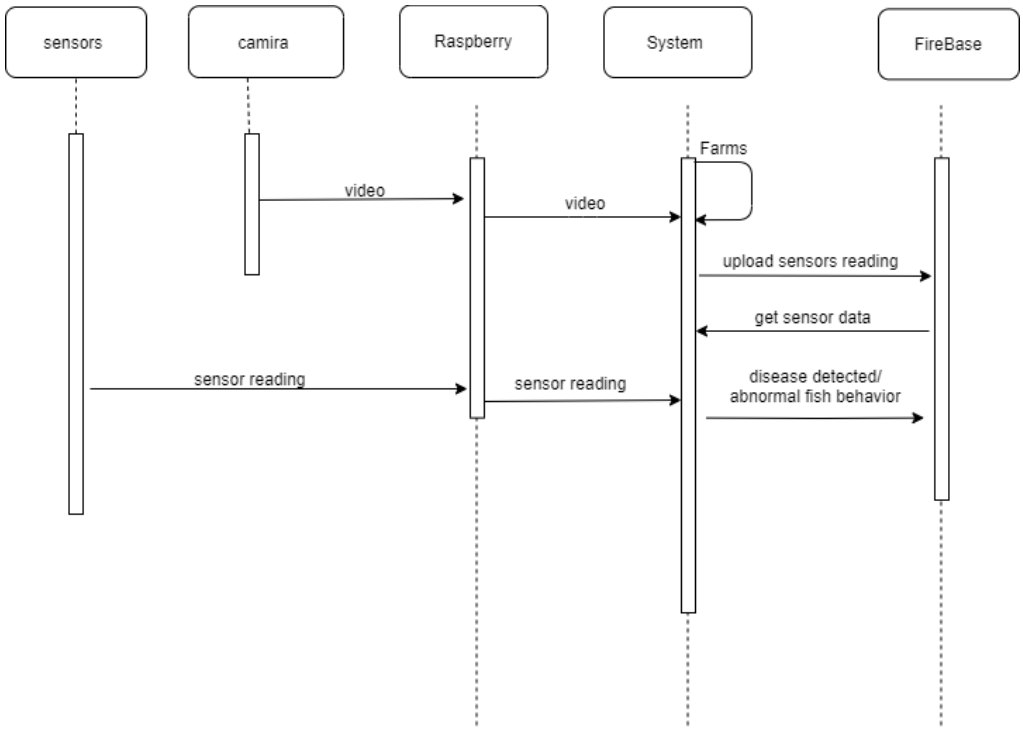


Figure 4.8: Sequence diagram

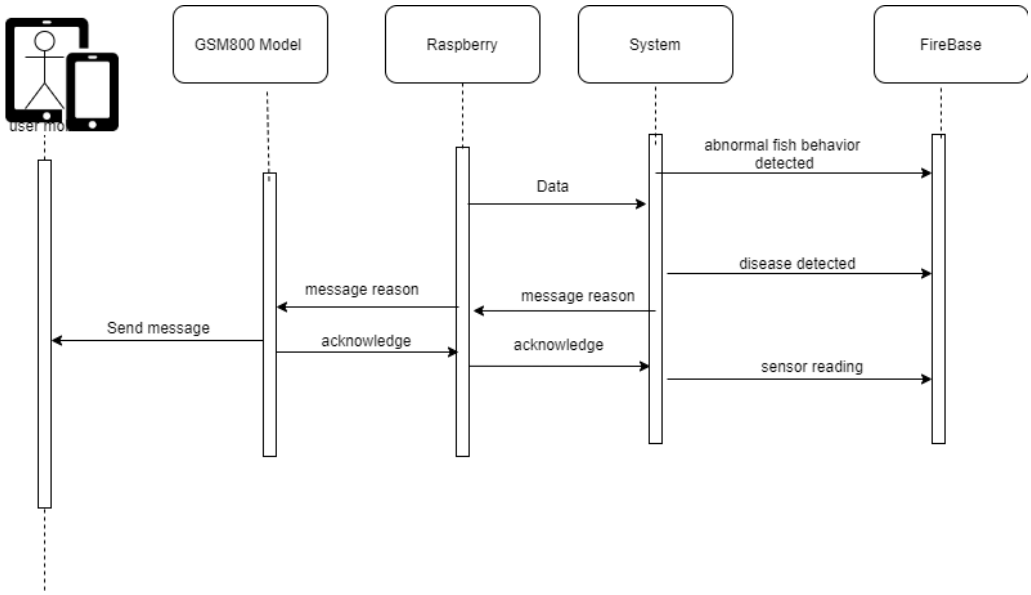


Figure 4.9: Sequence diagram

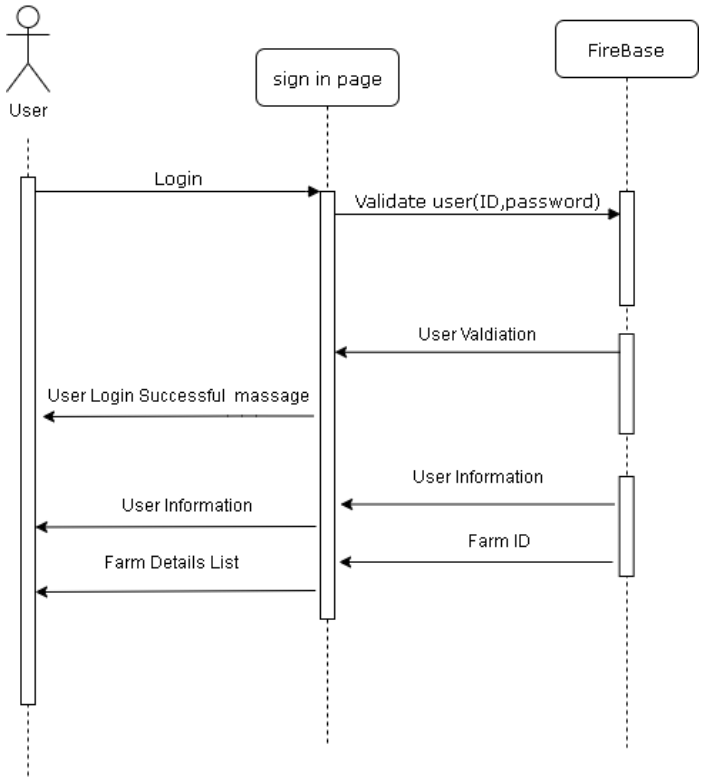


Figure 4.10: Sequence diagram

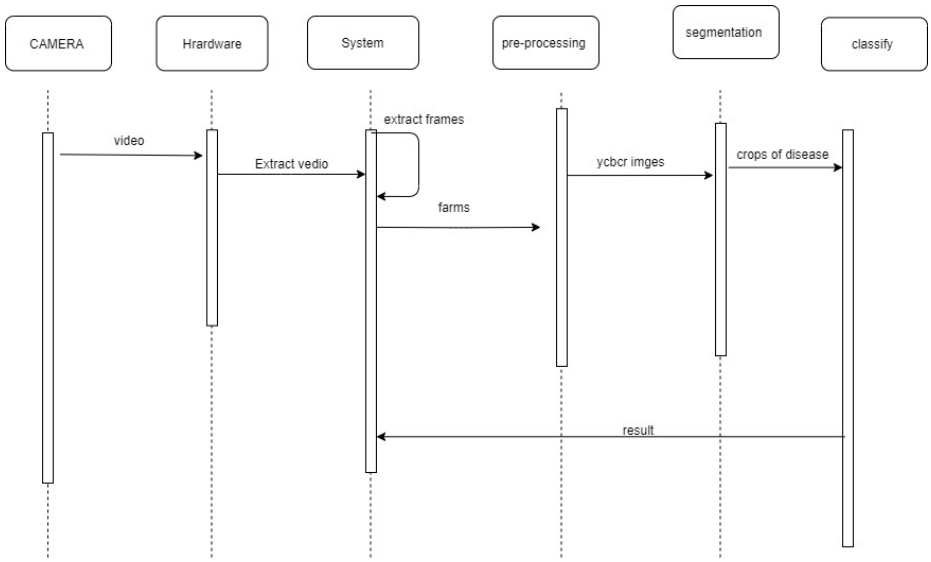


Figure 4.11: Sequence diagram

4.3.3 Design Rationale

As mentioned previously, we have used Model-View-Controller (MVC) as our architecture as it differentiating the layers of a project in Model, View, and Controller for the Re-usability of code and better maintenance. So, we can easily make modifications, re-use, and optimize the functionality part as it is our core. Accuracy is a very important aspect of our system so it will be very sensitive with data so this should be developed in a very accurate and efficient way.

4.4 Data Design

This is a relational preview of the database.

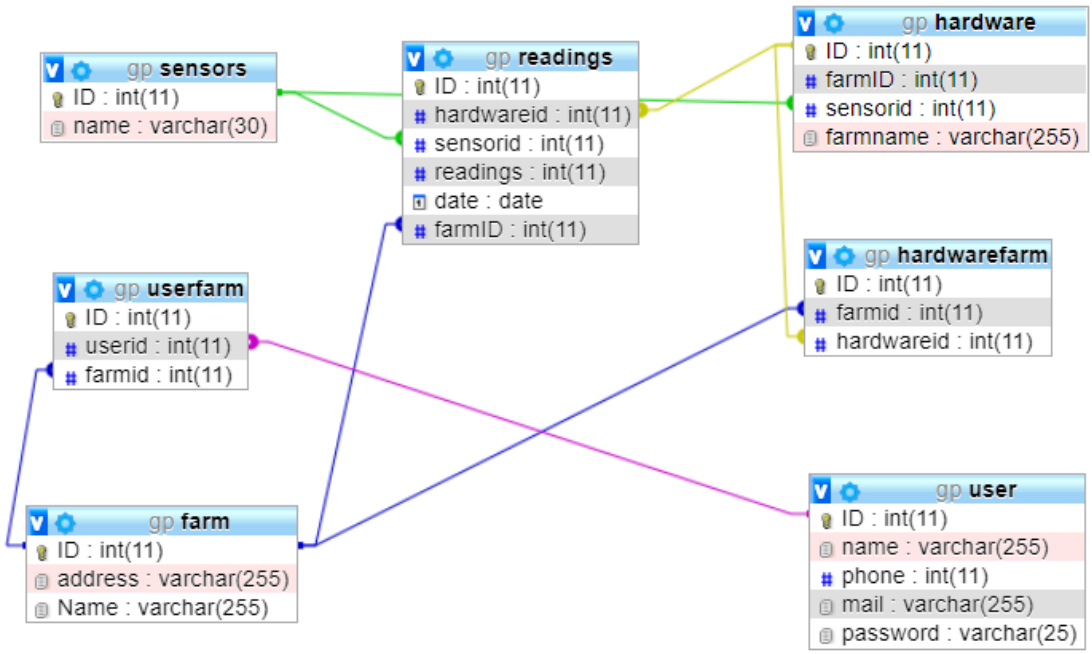


Figure 4.12: Database

4.4.1 Data Description

- User: This table contains the ID, username, phone, Email. It also contains.

- Readings: This table contains ID, Farm id because we have different Farms, Date-id to get the current date, sensors-id because we have (temperature-PH) sensors, readings for each sensor, and hardware-id because we have different hardware for each farm. In this table, the Admin needs to list all the sensor daily reading.
- Farm: This table contains ID, Name to be able to differ between the different Farms.
- Hardware: This table contains ID, Name to be able to differ between the different Hardware.
- Sensors: This table contains ID, name because we have (PH-Temperature) sensors. So, the attribute "name" is used to be able to differ between different sensors.

4.5 Component Design

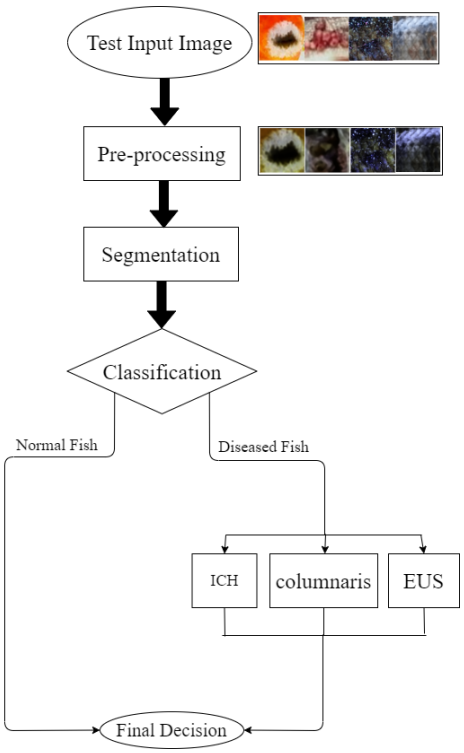


Figure 4.13: Flowchart for disease classification approach

4.5.1 Pre-processing

In this phase, we prepare the sample images for the segmentation process, we applied different color spaces on input images during the pre-processing phase, as shown in figure 4.14 Three different color spaces were applied; RGB, YCbCr ,XYZ and LAB.

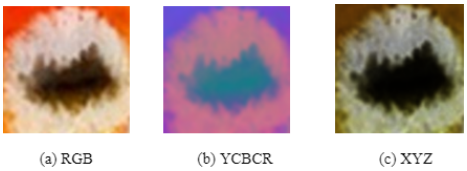


Figure 4.14: Infected Area in Different Color Spaces

4.5.1.1 RGB color space

RGB represents color as red, green and blue.

4.5.1.2 Ycbcr color space

YCbCr, is a family of color spaces used as a part of the color image pipeline in video. the Y is the brightness (luma), Cb is blue minus luma (B-Y) and Cr is red minus luma (R-Y). RGB color space is converted to ycbcr color space, After converting to YCbCr space, the extracted Cb and Cr coefficients are used for cell segmentation process. Conversion from RGB color space to Ycbcr is defined by equations 4.1, 4.2, 4.3.

$$Y = 16 + \frac{65.738R}{256} + \frac{129.057G}{256} + \frac{25.064B}{256} \tag{4.1}$$

$$Cb = 128 - \frac{37.945R}{256} - \frac{74.494G}{256} + \frac{112.439B}{256} \tag{4.2}$$

$$Cr = 128 + \frac{112.439R}{256} - \frac{94.154G}{256} - \frac{18.285B}{256} \tag{4.3}$$

4.5.1.3 Xyz color space

The XYZ color space has the unique property of being able to express every color that the human eye can see which in turn means that it can express every color that can be captured by a camera and hence every color that anyone might ever want to reproduce in the video. X, Y, and Z are mathematically generated RGB extrapolations to avoid negative numbers and are called Tristimulus values. Y means luminance, Z is somewhat equal to blue And X is a combination of orthogonal and non-negative cone response curves. RGB color space is converted to Xyz color space which is used for the segmentation process.

To convert from RGB to XYZ, the equations 4.4, 4.5 and 4.6 are applied consequently.

If $R, G, B \geq 0.04045$,

$$r, g, b = ((R, G, B + 0.055)/1.055)^{(2.4)}) \tag{4.4}$$

otherwise,

$$r, g, b = (R, G, B/12.92) \tag{4.5}$$

Then, matrix M is used to convert to XYZ.

$$\begin{pmatrix} X \\ Y \end{pmatrix} Z = (M) \begin{pmatrix} r \\ g \end{pmatrix} b \tag{4.6}$$

4.5.2 Segmentation

In general, segmentation is an essential stage to facilitate the classification process. The segmentation process is applied to group similar pixels together into a segment. In our system, the Gaussian models are applied for segmentation. Building an effective Gaussian distribution requires a suitable computed means and covariances. Small crops of different diseases were used to get the different values of the infected pixels values in different color spaces. Based on the diseases pixels values, a mean and covariance is computed for each disease. Accordingly, three different Gaussian distribution models are built for each disease. Applying the different models on the input images results in computing the probability of each pixel to be infected or not. Gaussian distributiona are applied to indicate the probability of including any infected area using mean and covariance.

Mean and covariance are computed to build the gaussian distribution as follow:

1. Read cropped images
2. Convert images to Xyz
3. Get X value from xyz
4. Apply low pass filter (LPF) to X
5. Get Z value from xyz.
6. Apply low pass filter (LPF) to Z
7. Compute mean and covariance of x and z values

The built distribution is then applied on test images after computing mean and covariance values on the XYZ color space. The adaptive threshold was finally applied to choose the segmented area based on the computed probability.

4.5.3 Classification

You might know about deep neural network and specially convolution neural networks are very effective in building recognition systems [42]. Deep neural networks get raw data and features are computed and learned inclusively. CNN can have tens or hundreds of layers that each learns to detect different features of an image. It is composed of an input layer,

an output layer, and many hidden layers in between. Three of the most common layers are convolution, activation, and pooling.

4.5.3.1 Convolutional Layer

Convolutional layers are the major building blocks used in convolutional neural networks. This layer is performed when we extract a segment from the image sized (n*n) centered at location (x,y). We then multiply the extracted segment element-by-element with convolution filter also sized (n*n) and add them all together to create a single output. As shown below in figure 4.15.

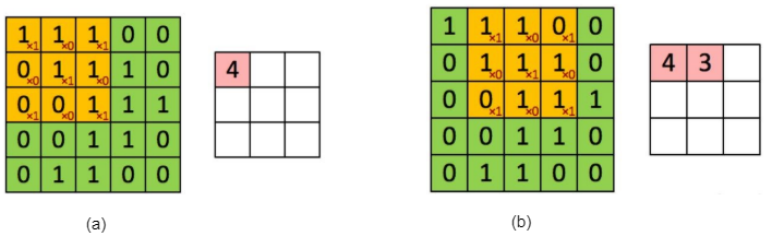


Figure 4.15: Convolutional Layer

4.5.3.2 Activiation Layer

There are several activation functions each have a different function. Some of the popular types of activation functions are Relu, Softmax, linear, and Sigmoid. The activation function [42] is a mathematical "gate" in between the input feeding the current neuron and its output going to the next layer as shown in fig 4.16.

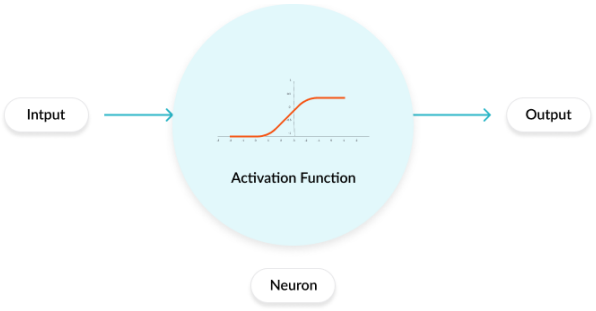


Figure 4.16: Activiation Layer

4.5.3.3 Max Pooling Layer

The pooling layer is mainly used to reduce the image size (Width and Height) [43] as shown in figure 4.17

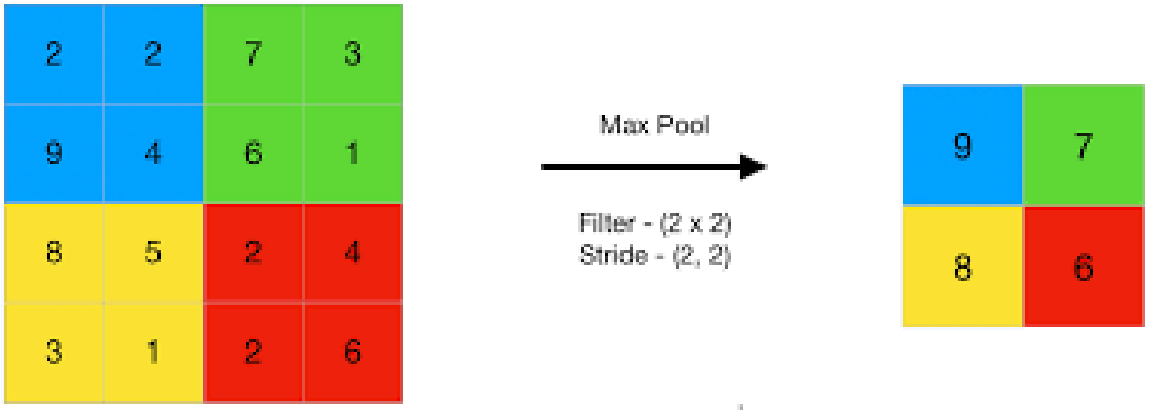


Figure 4.17: pooling Layer

4.5.3.4 CNN Architectures

CNN is a subset of deep learning techniques and has become dominant in most computer vision tasks. It is composed of several layers which are the input layer, an output layer, and various hidden layers. Some of these layers are convolution layers, pooling layers, and fully connected layers. There are different architectures in CNN, 16 different architectures were applied in the proposed system as shown in table 4.1 and 4.5.3.4.

Table 4.1: Comparison between different CNN Architectures

Model	Layers	Activation function	Input size	Convolution kernel size	Innovative point
ResNet [44]	18 layers 50 Layers 101 Layers	RelU	224x224x3	ResNet initial convolution: 7x7. Resnet 50 and 101: 1x1, 3x3 and 1x1. ResNet 18: 3x3	ResNet overcomes degradation and vanishing gradient problems residual blocks that increase the number of hidden layers. The core idea of ResNet is introducing "identity shortcut connection" that skips one or more layers
VGG [45]	16 layers 19 layers	ReLU non-linear	224x224x3	VGG: 3x3	This network uses only 3x3 convolutional layers stacked on top of each other in increasing depth. VGG improves AlexNet by replacing large kernel-sized
SqueezeNet [46]	18 layers	RelU	227x227x3	SqueezeNet: 1x1 and 3x3	SqueezeNet goal was to create a small neural network with fewer parameters. It was able to achieve a 50X reduction in model size compared to AlexNet. SqueezeNet has the advantage of fire module, which uses fewer filters to decrease the number of parameters
DenseNet [47]	201 layers	ReLU	224x224x3	DenseNet: 7x7	In DenseNet, the feature-maps of all preceding layers are used as inputs, and its feature-maps are used as inputs into all subsequent layers. DenseNet alleviate the vanishing gradient problem and reduce the number of parameters
AlexNet [48]	8 layers	ReLU non-linear	227x227x3	Alexnet: 11x11, 5x5 and 1x1	AlexNet has a large number of filters to perform the convolution operation of sizes 11x11, 5x5 and 3x3

GoogleNet [49]	22 layers	ReLU	224x224x3	GoogleNet: 5x5, 3x3 and 1x1	GoogleNet increases the depth of the network and gains a higher performance level. It is based on the concept of the inception module, it is the collection of convolution and pooling operation performed in a parallel manner so that features can be extracted using different scales
Mobilenetv2 [50]	54 layers	ReLU6	224x224x3	Mobilenetv2: 1x1, 3x3	Mobilenetv2 improves the performance of mobile models on multiple tasks. Mobilenetv2 is based on an inverted residual structure
Xception [51]	71 layers	ReLU	299x299x3	Xception: 1x1,3x3	Xception involves Depthwise Separable Convolutions, it is supposed to be more efficient than classical convolution in terms of computation time. Xception relies on Shortcuts between Convolution blocks
Inceptionv3 [52]	48 layers	ReLU	299x299x3	Inceptionv3: 3x3	In inceptionv3, computational efficiency and fewer parameters are realized
ShuffleNet [53]		RelU	224x224x3	Shufflenet: 1x1,3x3	Shufflenet aims to explore a highly efficient architecture specially designed for limited computing ranges. Shufflenet allows more feature map channels and it is especially critical to the performance of very small networks. ShuffleNet achieves 13x actual speedup over AlexNet while maintaining comparable accuracy

4.5.4 Object detecting

In our proposed approach, vision.CascadeObjectDetector were applied for object detection. The objects are detected by applying a sliding window over the image. A cascade classifier is then used by the detector to determine if the window contains the object of interest. The window size for object detection varies on different scales. Cascade classifier consists of stages, where each stage is an ensemble of weak learners. The weak learners are simple classifiers called decision stumps. Each stage is trained using a technique called boosting. To converts, weak learners, to strong learners and combine weak learners to form a strong rule.

A decision stump is a machine learning model consisting of a one-level decision tree, with one root which is immediately connected to its nodes. A decision stump makes a prediction based on the value of just a single input feature. It is known as weak leaner as it can make a lot of mistakes if there is crossover. Sometimes they are also called 1-rules of base learners. Depending on the type of the input feature, for nominal features, one may build a stump which contains a leaf for each possible feature value or a stump with the two leaves, one of which corresponds to some chosen category, and the other leaf to all the other categories. It is a very simple decision tree, and also easy algorithm.[54]

So we used boosting technique to converts weak learners to strong learners and combines weak learners to form a strong rule. There are three types of boosting Algorithms which are Adaptive Boosting Algorithm (AdaBoost), Gradient Boosting algorithm and XG Boost algorithm [55].

How Boosting Algorithms Works? Boosting Algorithms combine each weak learner to create one strong prediction rule. Base Learning algorithm is applied to identify the weak rule using the iteration process. Weak rules are combined after some iterations to create one single prediction rule. To choose the right distribution:

- Each distribution is combined by base learning algorithm, to apply equal weight to each distribution.
- We pay high attention to that prediction error if any prediction occurs during the first base learning algorithm.
- Step 2 is repeated until the limit of Base Learning algorithm has been reached or high accuracy.
- Finally, all the weak learner are combined to create one strong prediction rule.

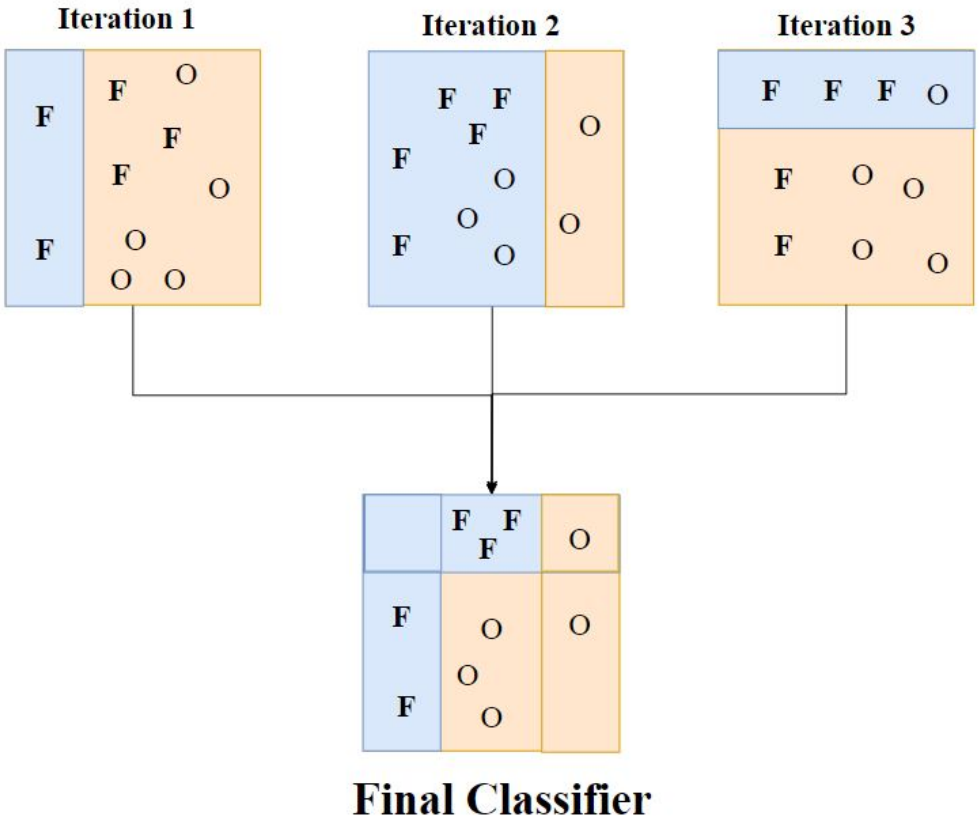


Figure 4.18: Boosting Algorithm

Cascade classifier training requires a set of positive and negative samples. Negatives samples indicate that an object was not found and the detector moves the window to the next pixel. While positive samples indicate that an object was found and the detector passes the region to the next stage. TrainCascadeObjectDetector is then applied to determine the number of positive samples to use for training each stage. The trainCascadeObjectDetector supports two types of features Haar and HOG features.

Haar cascade is a machine learning based approach. Positive and negative images used to train cascade function. Cascade function is then applied to detect object in other images.

The algorithm consists of 4 stages:[56]

- Haar Feature Selection often applied for face detection as it works well for fine-scale textures representation. It considers adjacent rectangular regions at a

particular location in a detection window, pixel intensities in each region are summed up and the difference between these sums is then calculated

- Creating integral images applied to make this super-fast.[57]
- Adaboost Training

  Adaptive Boosting Algorithm (AdaBoost) selects the best features and the classifiers that use them are trained. A strong learner is created by combining a group of weak learner base on weightage. It gives equal weight to each data set and starts predicting that data set in the first iteration. If incorrect prediction occurs it gives high weight to that observation. Adaptive boosting repeats this procedure in the next iteration phase and continue until the accuracy has been achieved. Then combines this to create a strong prediction.
- Cascading Classifiers

Histogram of oriented gradients (HOG) is a feature descriptor that is applied for object detection such as people and cars in computer vision and image processing techniques. The occurrences of gradient orientation are counted in localized portions of an image - detection window or region of interest (ROI). Hog features are useful for capturing the geometric description of an object.

The algorithm stages:[58]

- Preprocessing

  The image is resized by having some fixed aspect ratio or some fixed size of an image.
- Calculate the gradient images

  The horizontal and vertical gradients of an image are calculated. Then, the gradient magnitude by both horizontal and vertical gradients are calculated.
- Calculate the HOG 1- The image is divided into small connected regions called cells. histogram of oriented gradients is then computed for each cell.

  2- Each cell is discretized into angular bins.

  3- Each cell's pixel contributes a weighted gradient to its corresponding angular bin.

  4- Groups of adjacent cells are considered as spatial regions called blocks. The grouping of cells into a block is the basis for grouping and normalization of histograms.

5- The block histogram is represented by a normalized group of histograms. The set of these block histograms represents the descriptor.

- Adaboost Training
- Cascading Classifiers

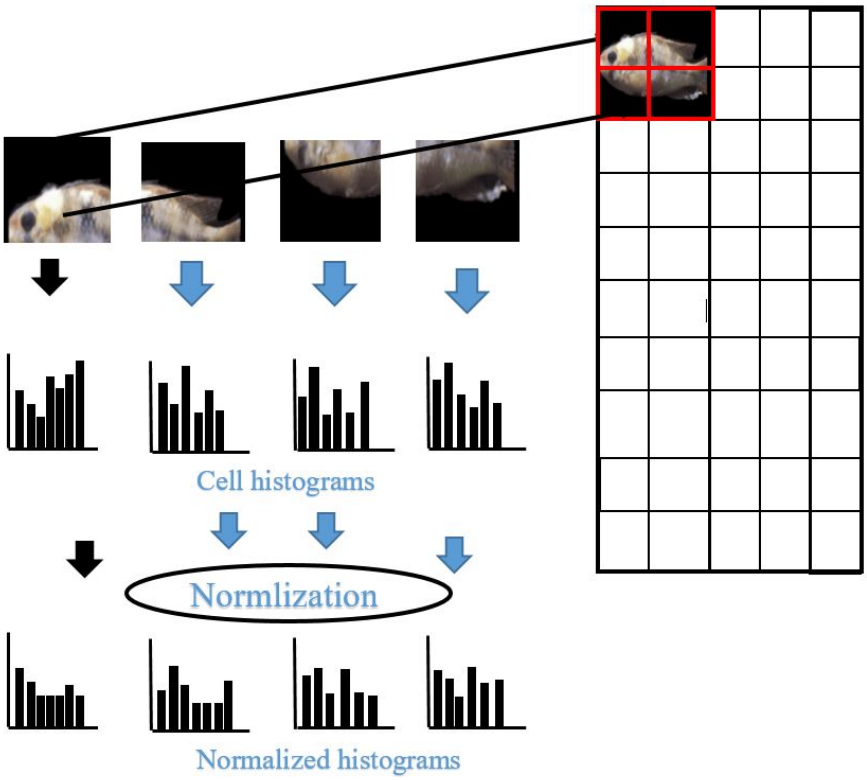


Figure 4.19: HOG Algorithm

Hog features are improved by applying "MinSize" and "Threshold" to make detection more accurate. Minsize is the smallest object detected and is 63x279, while the threshold value controls the accuracy and speed for classifying image subregions as either objects or nonobjects to speed up the performance at the risk of missing true detections, increase this threshold.

4.5.5 Tracking

In the tracking phase, a Vision.PointTracker is applied to tracks set of points by using the Kanade-Lucas-Tomasi (KLT) algorithm. The KLT algorithm is used for tracking an image patch. The point tracker is applied for object tracking in videos. The implementation of the point tracker using the KLT algorithm uses some features. The features computed for this stage are a combination between the Number Pyramid Levels and Forward-backward error threshold, Size of the neighborhood, and the Maximum number of search iterations. In the number pyramid levels feature, we study the effect of different levels of the Pyramid (L=1,2 and 3) as shown in figure 4.20. Increasing the number of levels allows the algorithm to handle the largest displacements between frames.

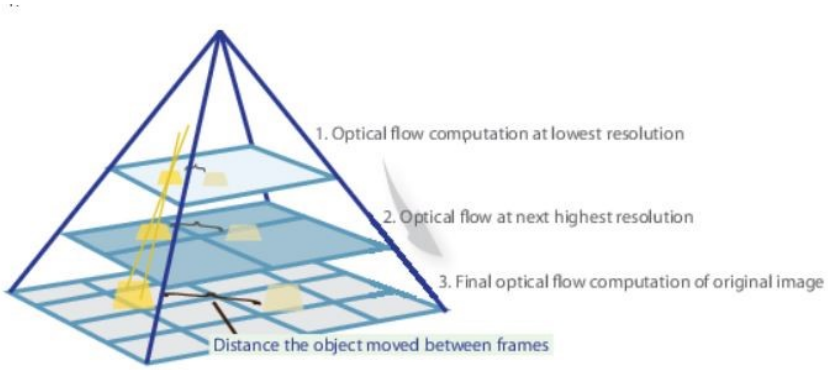


Figure 4.20: Pyramids Levels

In the Forward-backward error threshold, each point is tracked from the previous frame to the current frame, as shown in figure 4.21. The bidirectional error is then calculated. This value is the distance in pixels from the original location of the points to the final location after the backward tracking. The value ranges from 0 to 3.

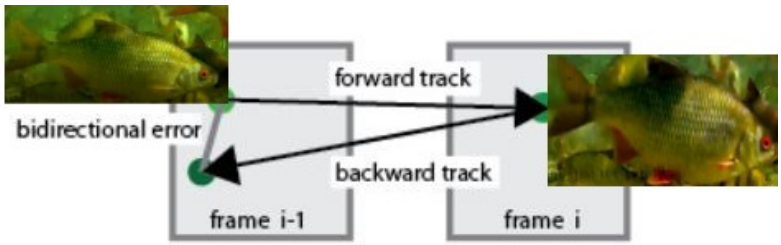


Figure 4.21: Forward-backward error threshold

In the size of the neighborhood feature, blockSize is determined by two elements, height, and width. The block size takes the minimum value [5,5], and increasing the block size increases the computation time. Height and width must be odd values.

In the Maximum number of search iterations, the iteration ranges between 10 to 50. An iterative search is performed using the KLT algorithm, it performs an iterative search for the new location of each point until convergence.

The tracking process is them initialized to specify the initial location of points and video frame. DetectMinEigenFeatures is then applied by using minimum eigenvalue algorithm to detect corners and return cornerPoints object. Eigenvalues help to Find Corner Points.

estimateGeometricTransform is then applied to determine the transformation between the point locations in the previous and current frames, as shown in figure 4.22. A geometric transformation is a function whose domain and range are sets of points.

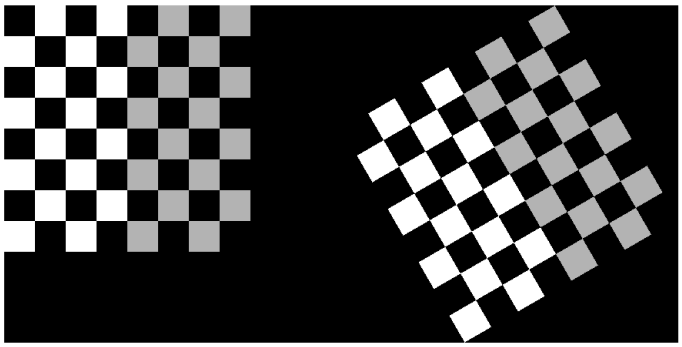


Figure 4.22: Geometric Transformation For Image

After applying fish tracking, fish velocity is then calculated to help in the expectation of fish behavior.The velocity is calculated my multiplying the distance of centroids between previous frame and current frame. This is done by getting the video frame rate (frame/second), the video scale (meter/pixel). This is defined in equation 4.7.

$$\sqrt{\sum((newpoint - oldPoints)^2} * FrameRate * scale \tag{4.7}$$

$$\frac{pixels}{frame} * \frac{frame}{seconds} * \frac{meter}{pixels} \tag{4.8}$$

4.5.6 Hardware

This section introduces the interface circuit that connects reality and computer recognition systems. The interface circuit consists of three main parts. The first part is a Raspberry pi camera which is used to capture fish movements underwater. It is a professional camera with a high-resolution day/night vision, with a fisheye lens for a wider field of view and an adjustable focal lens. It can produce high definition (HD) videos. The second part is PH and Temperate sensors which are used to examine water quality. PH is a measure of acidity or alkalinity of a solution. The pH scale ranges from 0 to 14. Temperature sensors measure temperatures in wet environments with an advantage over linear temperature sensors calibrated in Kelvin. finally, the last part is Raspberry Pi 3 kit which is used to get the sensor's measurements and acquired fish captures. It has built-in (onboard) WiFi and Bluetooth.

As shown in fig 4.23, Raspberry pi gets the sensors' measurements and acquired fish captures from the camera and passes it to the computer system. The connection could be through Ethernet cable or wifi. Ethernet is for short-distance connection, while wifi for long-distance connection. Matlab support package for raspberry pi is a library in MatLab that receives these inputs from raspberry pi and passes them to MYSQL database server. MySQL is a relational database management system that runs on a computer. MYSQL database sends this data to firebase using PHP, ajax, and javascript code. Firebase is used to build web and android applications. Users can receive notifications using firebase cloud messaging (FCM) through web and android applications. FCM is a notification service that enables developers to send notifications to their users.

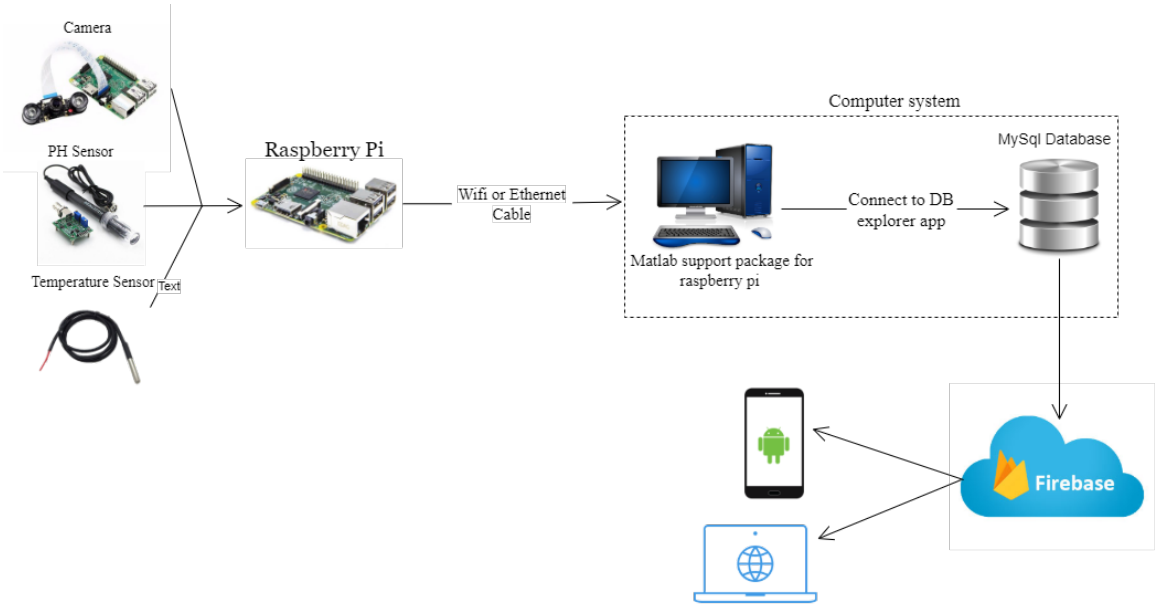


Figure 4.23: Hardware connection

The following figure 4.24 represents the connection between temperature and PH sensors on Raspberry pi 3. The temperature and PH sensors require 3 connections to the Raspberry Pi which are: 5V/3.3 voltages, ground (GND), and General Purpose Input Output (GPIO). The Raspberry Pi has Two 5v pins and Two 3v3 pins. Ground pins are electrically connected in raspberry pi so it doesn't matter which one to use. Raspberry Pi contains 40 pins GPIO that are divided into two rows. Temperature sensor is a digital sensor that is connected to 3.3V pin which is PIN one in figure 4.24. The output pin of the sensor GPIO3 pin which is PIN five. The GPIO of the temperature sensor is then connected to the GND pin which is PIN 25. PH sensors is analog sensor that is connected to 5V pin which is PIN two in figure 4.24. The output pin of the sensor GPIO2 and GPIO22 pins which are PINS four and 15. The GPIO of the PH sensor is then connected to the GND pin which is PIN 14.

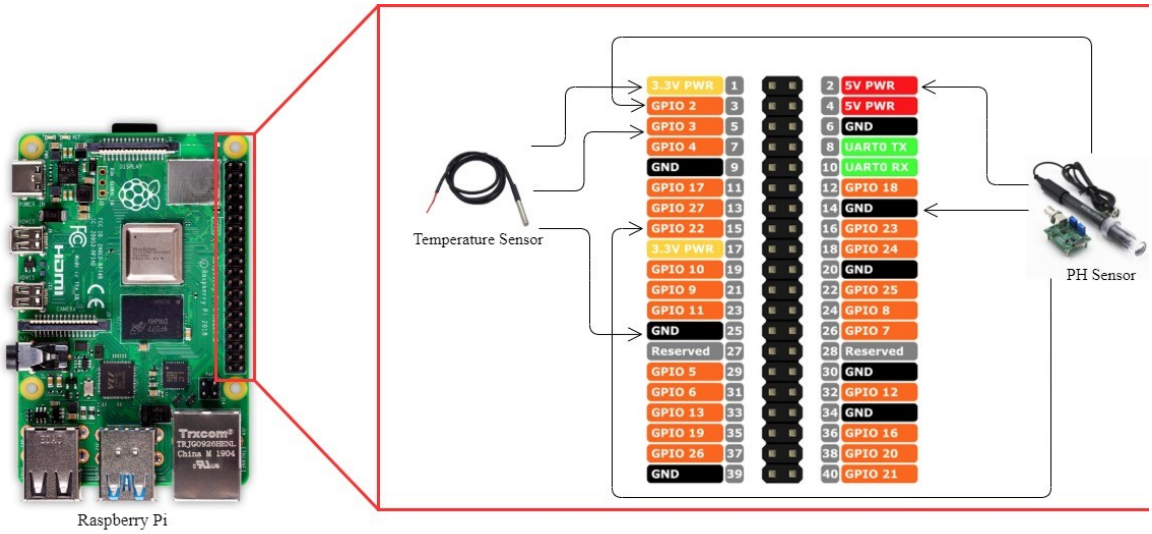


Figure 4.24: Raspberry pi connection with sensors

The following figure 4.25 represents the connection between Raspberri pi camera on Raspberry pi 3.

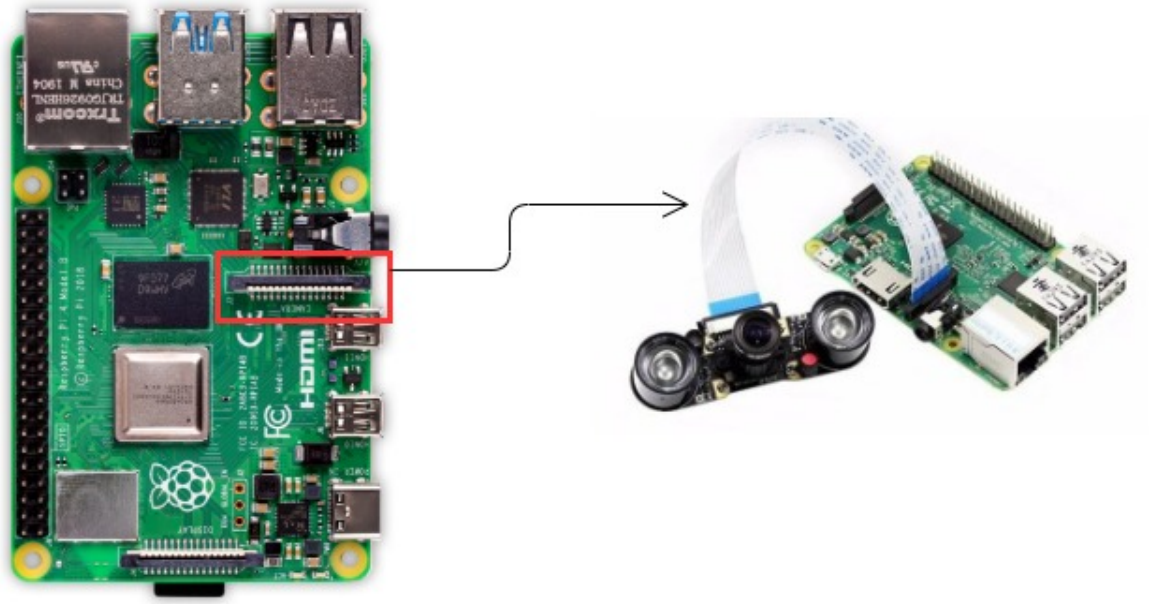


Figure 4.25: Raspberry pi connection with camera

4.6 Human Interface Design

4.6.1 Overview of User Interface

Our system user interface is very usable and clear as shown in figures 4.26 and 4.27. You can login whether you are an admin or user. Once logged in, He/She will be directed to the homepage menu.

- Sign up view: This interface allows new users access the application
- Sign in: This interface is meant for the users to login the application.
- The Home Page is an interface that manages all the menus inside our system.

The Home Page will provide 4 modules which are:

- Edit Profile: This interface allows each user to edit his personal data
- Profile: This interface allows users to show all his personal data
- User module: This interface to manage any user data.
- More List: This will include all managements of farms and hardware.

4.6.2 Screen Images

Figure 4.26: Admin Screens

Figure 4.27: User Screens

4.7 Requirements Matrix

Table 4.2: Requirements Matrix

ReqID	Reqtype	Requirement Name	Requirement Description	Status	Implemented In
3.1	Required	Register	The user registers with his/her information to create an account	Completed	Android App
3.2	Required	Login	user/Admin can login with his/her username and password into his/her account	Completed	Android App
3.3	Required	Signout	user/Admin sign out from their system	Completed	Android App
3.4	Required	delete User	the admin can remove user from system	Completed	Android App
3.5	Required	listing all users	the admin can list all users in system	Completed	Android App
3.6	Required	View Readings	User can view sensors readings of his fish farm	completed	Android App
3.7	Required	Show notification	user can show notification when there is any improper change	completed	Android App
3.8	Required	Adding fish farm	The admin adds farm to specific user	completed	Android App
3.9	Required	Edit farm	The admin edit farm details in the system	completed	Android App
3.10	Required	Deleting farm	The Admin deletes farm from the system	completed	Android App
3.11	Required	listing all farms	The Admin lists all the farms found in the system	completed	Android App
3.12	Required	Adding hardware	The admin adds new hardware to the system	completed	Android App
3.13	Required	Edit hardware	The admin edit hardware details in the system	completed	Android App
3.14	Required	Deleting hardware	The Admin deletes hardware from the system	completed	Android App
3.15	Required	listing all hardware	The Admin lists all the hardware found in the system	completed	Android App
3.16	Required	Take video	This function is used to take video from camera	Completed	Hardware

Table 4.3: Requirements Matrix

ReqID	Reqtype	Requirement Name	Requirement Description	Status	Implemented In
3.17	Required	Get PH sensor reading	Get Reading from PH sensor and pass it in Raspberry pi	completed	Hardware
3.18	Required	Get Temperature sensor reading	Get Reading from Temperature sensor and pass it in Rasp-berry pi	completed	Hardware
3.19	Required	Send Notification	Notification will be sent to the user through mobile application to notify him/her that he/she has improper changes in farm environment or detected infections	completed	Hardware
3.20	Required	Upload PH sensor readings to firebase	This function is for collecting PH sensor readings from rasp-berry pi and uploading it to firebase	completed	Hardware
3.21	Required	Upload temperature sensor to firebase	This function is for collecting temperature sensor readings and uploading it to firebase	completed	Hardware
3.22	Required	Listing all sensors readings	This function is for listing all sensors readings foundin the system	completed	Hardware
3.23	Required	Data augmentation	This function is used to increase the diversity of images that is available in data-set for training models	Completed	Tracking and disease detection
3.24	Required	Create XML file This XML	file contain data which is extracted from video in the dataset to be compared with frames taken from video	Completed	Tracking and disease detection
3.25	Required	Draw border	This function is used to add border on detected fish	Completed	Tracking and disease detection

Table 4.4: Requirements Matrix

ReqID	Reqtype	Requirement Name	Requirement Description	Status	Implemented In
3.26	Required	Count Number of detected fish	This function is used to count borders of detected fish	Completed	Tracking and disease detection
3.27	Required	Video segmentation	This function is used divide video into frames then Convert frames to YCBCR then change values of YCBCR	Completed	Tracking and disease detection
3.28	Required	Point Tracker	This function is used to track a set of points	Completed	Tracking and disease detection
3.29	Required	Velocity and acceleration	This function is used to calculate velocity and acceleration	Completed	Tracking and disease detection
3.30	Required	Estimate Geometric Transformation	This function is applied to find the transformation matrix	Completed	Tracking and disease detection
3.31	Required	Detect Minimum Eigen Features	This function detects corners and return cornerPoints.object	Completed	Tracking and disease detection
3.32	Required	Transform Points Forward	This function applies the forward transformation of2-D geometric transformation	Completed	Tracking and disease detection
3.33	Required	Histogram Based Tracker	This function is used to identify the tracked object	Completed	Tracking and disease detection
3.34	Required	Convert to YCBCR	Converts RGB image to YCBCR	Completed	Segmentation
3.35	Required	Convert to XYZ	Converts RGB image to XYZ	Completed	Segmentation
3.36	Required	Resize image	Image will be resized to cover the part that will be classified	Completed	Pre-processing
3.37	Required	Apply Gaussian distribution	It applies the predefined Gaussian filter on the image	Completed	Segmentation
3.38	Required	Calculate Mean of CB/CR	It Calculates the Mean of CB/CR	Completed	Segmentation

Table 4.5: Requirements Matrix

ReqID	Reqtype	Requirement Name	Requirement Description	Status	Implemented In
3.39	Required	Calculate Covariance	Calculate Co-variance of ycbcr	Completed	Segmentation
3.40	Required	Calculate Mean of f X/Y	It Calculates the Mean of X/Y	Completed	Segmentation
3.41	Required	Calculate Covariance of XYZ	It Calculates the covariance	Completed	Segmentation
3.42	Required	Apply Gaussian distribution of xyz	t applies the predefined Gaussian filter on the image	Completed	Segmentation
3.43	Required	Adaptive threshold	It applies the threshold on Gaussian distribution model result	Completed	Segmentation
3.44	Required	Crop	It crops diseased area	Completed	Segmentation
3.45	Required	Convert to gray scale	Converts RGB image to gray scale	Completed	Segmentation
3.46	Required	GLDM	It gets the probability density function	Completed	Segmentation
3.47	Required	CNN Classifier	this function is used to train data and integrate with it with features to classify new inputs of images	Completed	Classification

Chapter 5

Evaluation

5.1 Introduction

After the system's implementation finished, where all the systems' functionalities are implemented, the system should pass through the evaluation process. In this phase, the system is tested through many experiments. Multiple experiments were plotted to test out the features associated with the system. Comparison between our system and traditional methods is also available.

5.2 Database

Our data-set includes images of fish with the three types of fish diseases, as shown is Fig 5.1. These diseases are EUS, ICH and Columnaris. Deep learning technique is based on large training dataset for the system to build up the identification knowledge, enough data has to be provided as learning resources. Our data-set were collected from different internet resources. However, the number of images collected is not large enough to train the system. Therefore, we applied data augmentation [59] to increase our samples because the collected images are not sufficient for training purpose. Augmentation was based on four types of transformation, which increased the number of images to make the training dataset sufficient. It was based on zooming, shearing, rotating, and applying preprocessing function. Our collected data-set include 15 images per disease. After applying data augmentation, samples are increased to 800 images per disease. For training and testing the CNN, we

divided them into two groups; 550 images were used as training samples per disease, and the remaining samples were used for testing.

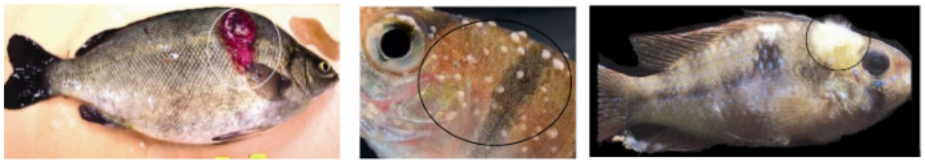


Figure 5.1: Infection Diseases

5.3 Comparative results of similar systems

Table 5.1, summarize comparative results with similar systems. EUS disease was recognized by two different approaches [8],[9]. Diseased regions were extracted by combining some features [10] that resulted in achieving more convenient and rapid diagnosis for diseased fish. CNN technique were applied to for fish species recognition and achieved satisfactory results [11] [13][14]. Our proposed method outperformed the similar system, by diagnosing three different diseases with high accuracy.

Table 5.1: Comparative results of similar systems

Systems	Recognition problem	Features extraction	Classifier	Recognition rate
[8]	EUS disease	FAST	Neural Network	86%
[9]	EUS disease	PCA	Mophological open	90%
[10]	white spot, trichodina and scuticociliate	polar coordinates, geometrical features	PCA	90%
[23]	Fish species	Artificial neural networks (ANN)	ANN	99%
[15]	Fish species	CNN	CNN	95%
[14]	Fish species	CNN	CNN	96.29%
[11]	Fish detection	CNN	YOLO	93%
Proposed Approach	EUS, ICH and Columnaris	AlexNet	AlexNet	99.0446%

5.4 Segmentation

For segmentation, we build the Gaussian model on XYZ, LAB and YCBCR color spaces. Table 5.2 and figures 5.2, 5.3 and 5.4 shows the effect of segmentation after applying it on 3 different diseases in 3 different color spaces. We found that EUS achieved good results in LAB color space, while ICH and Columnaris achieved good results in XYZ color space.

Table 5.2: Segmentation Experiment on different color spaces

Color Spaces	EUS	ICH	Columnaris
XYZ	45.90	70%	61%
YCBCR	49%	67.50%	44%
LAB	65%	39.40%	10%

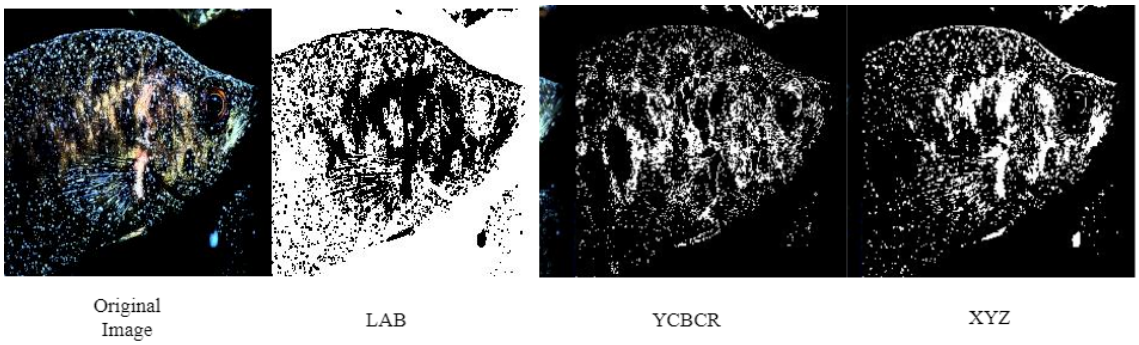


Figure 5.2: ICH diseased fish in LAB, YCBCR and XYZ color space

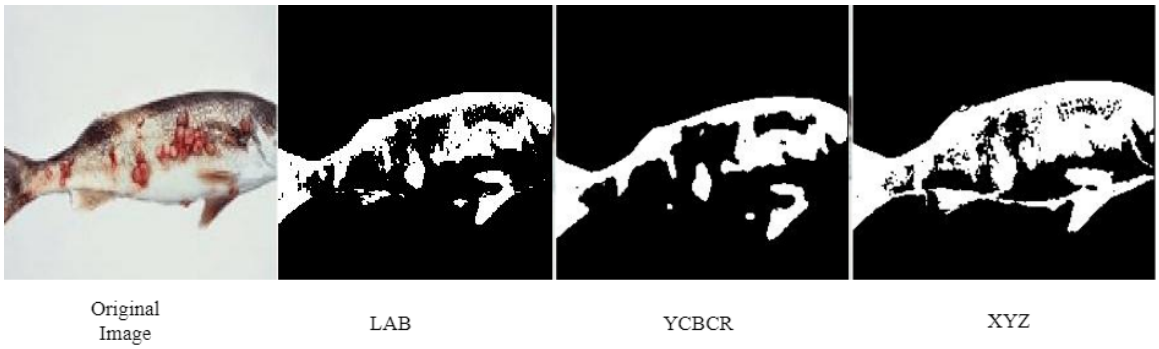


Figure 5.3: EUS diseased fish in LAB, YCBCR and XYZ color space

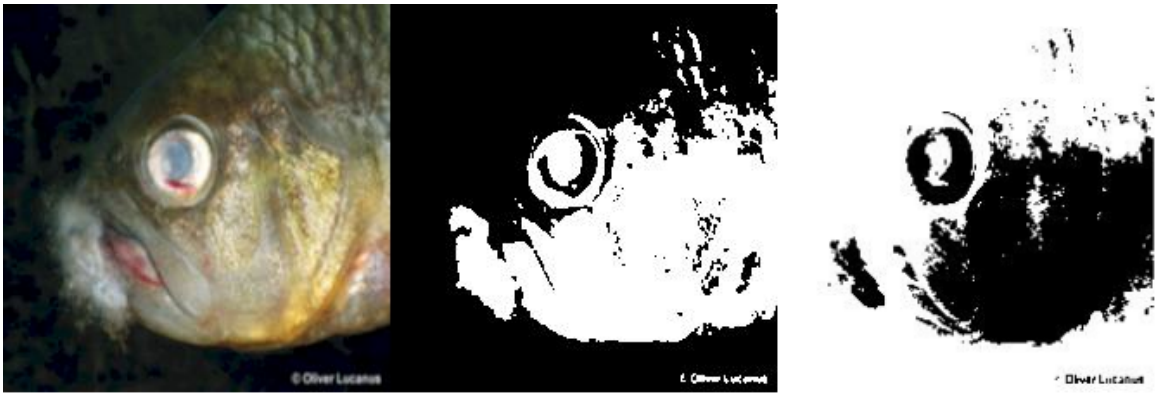


Figure 5.4: Columnaris diseased fish in XYZ and YCBCR color space

5.5 Classification process

In our experiments, we applied different CNN architectures in detecting diseases and normal fish . We compared between the performance of the ResNet18, ResNet50, ResNet101, Alex-Net, VGG16, VGG19, mobilenetv2, Xception, Inceptionresnetv2, Shufflenet, Nasnet-mobile, Nasnetlarge, Squeezenet, Inceptionv3, Densenet201, Googlenet CNN architectures. Each architecture is trained on 2160 samples, and finally tested on 314 samples. Accuracy is measured at three different learning rates 0.1, 0.01 and 0.001. Experiments were achieved using three different color spaces: RGB, YCBCR and XYZ, as shown in figures 5.5, 5.6 and 5.7. The figures showed that AlexNet achieved best testing accuracy at learning rate 0.01 in the XYZ color space. AlexNet at learning rate 0.001 has the same testing accuracy of 0.01 but with more training time in XYZ color space. Table 5.3 shows that ResNet101 architecture achieved the highest training accuracy at learning rate 0.001 in the RGB color space. Densenet201 out-performed the other architectures in time at learning rate 0.001 in RGB color space. ResNet18 performed the worst architecture in terms of time at learning rate 0.1 in RGB color space.

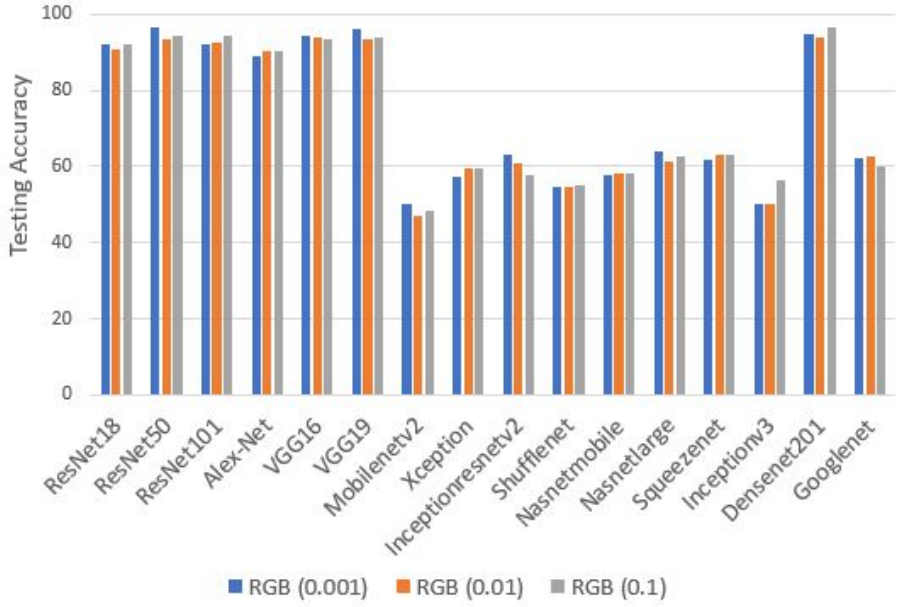


Figure 5.5: Achieved testing accuracy of different CNN architecture in RGB color space

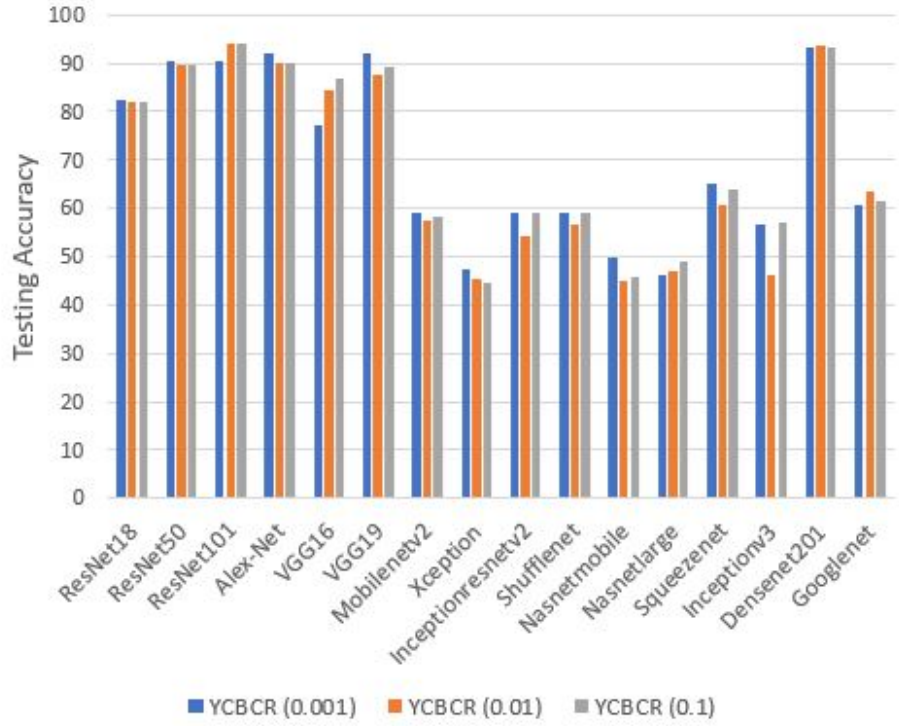


Figure 5.6: Achieved testing accuracy of different CNN architecture in YCBCR color space

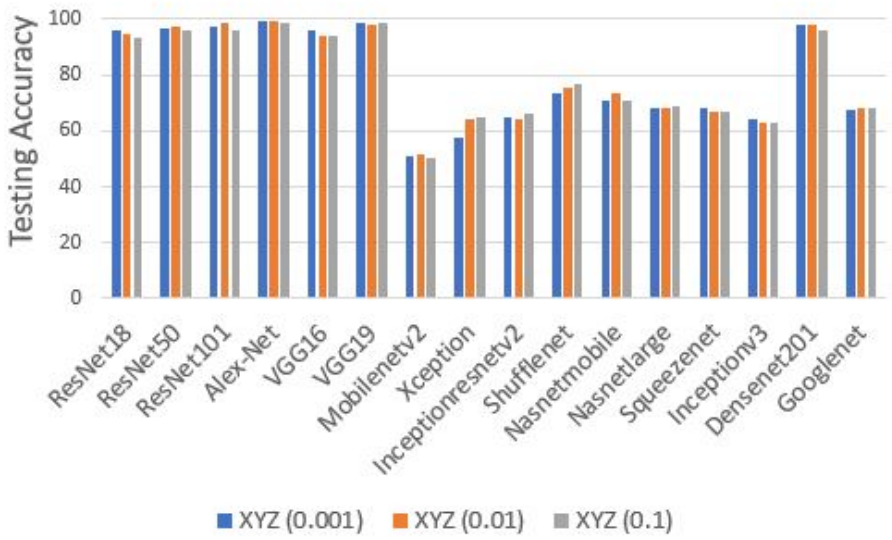


Figure 5.7: Achieved testing accuracy of different CNN architecture in XYZ color space

Table 5.3 and figure 5.5 shows experimental result in RGB color space in different learning rates. Different CNN architectures were applied to RGB color space. Densenet201 architecture achieved best testing accuracy in RGB at learning rate 0.1, as shown in figure 5.5, while ResNet101 achieved the highest training accuracy at learning rate 0.001 and Densenet201 out-performed other architectures in training time at learning rate 0.001 as shown in table 5.3.

Table 5.3 and figure 5.6 shows experimental result in YCBCR color space in different learning rates. Different CNN architectures were applied to YCBCR color space. ResNet101 achieved best testing accuracy in YCBCR at learning rate 0.1 and 0.01 as shown in fig 5.6. Shufflenet out-performed other architectures in training time at learning rate 0.001 as shown in table 5.3.

Table 5.3 and figure 5.7 shows experimental result in XYZ color space in different learning rates. Different CNN architectures were applied to XYZ color space. AlexNet achieved highest testing accuracy in XYZ and in all color spaces at learning rate 0.01. Densenet201 out-performed all other architectures in all different color spaces in training time at learning rate 0.001 as shown in table 5.3.

Table 5.3: Achieved training time and Accuracy of Cnn Architectures
Cell format(approximated time(min)),Training Accuracy(%)

Cnn Architectures	RGB (0.001)	RGB (0.01)	RGB (0.1)	YCBCR (0.001)	YCBCR (0.01)	YCBCR (0.1)	XYZ (0.001)	XYZ (0.01)	XYZ (0.1)
ResNet18	16.48, 80%	14.43, 50%	24.4, 60%	12.59, 60%	12.37 ,50%	12.37, 50%	17.1, 70%	17.9, 50%	15.31, 50%
ResNet50	18.35, 70%	17.9, 60%	14.59, 60%	14.53, 60%	14.29, 50%	14.53, 50%	24.20, 50%	15.53, 50%	15.52, 50%
ResNet101	16.46, **80%**	14.49, 60%	16.29, 50%	13.1, 50%	14.53, 50%	13.5, 50%	15.26, 50%	16.34, 50%	19.31, 50%
Alex-Net	11.5, 60%	7.40, 50%	9.33, 60%	7.27, 60%	9.41, 50%	9.41, 50%	8.34, 50%	7.51, 50%	7.53, 60%
VGG16	14.0, 50%	14.30, 60%	14.35, 50%	15.33, 50%	18.2, 40%	18.20, 50%	16.15, 60%	16.0, 50%	15.35, 50
VGG19	14.5, 50%	13.55, 60%	14.9, 50%	21.46, 50%	17.47, 60%	13.55, 60%	16.23, 50%	15.3, 50%	15.22, 50%
Mobilenetv2	13.34, 50%	7.3, 60%	7.58, 50%	16.6, 60%	7.1, 40%	7.2, 40%	10.20, 60%	9.13, 60%	9.49, 60%
Xception	12.49, 50%	12.4, 50%	11.25, 50%	11.6, 50%	15.9, 60%	13.42, 50%	12.53, 50%	12.13, 50%	12.16, 60%
Inception resnetv2	11.14, 50%	16.10, 60%	11.56, 60%	11.54, 60%	11.28, 50%	11.32, 60%	15.11, 50%	11.59, 50%	12.17, 60%
Shufflenet	7.44, 50%	7.33, 60%	7.28, 50%	6.55, 60%	7.3, 50%	7.56, 50%	21.28, 50%	19.44, 60%	14057, 60%
Nasnetmobile	9.40, 50%	7.33, 50%	7.27, 60%	7.11, 50%	7.17, 50%	8.2, 60%	15.47, 60%	15.26, 60%	15.4, 60%
Nasnetlarge	13.16, 60%	13.46, 50%	15.20, 50%	14.0, 50%	14.23, 60%	13.48, 60%	14.36, 50%	14.35, 60%	14.39, 60%
Squeezenet	7.12, 50%	7.45, 60%	7.45, 50%	7.12, 50%	7.28, 50%	8.5, 50%	8.34, 60%	10.9, 50%	7.57, 50%
Inceptionv3	15.17, 50%	13.7, 40%	11.50, 50%	15.37, 60%	11.17, 60%	11.22, 50%	13.8, 50%	15.54, 50%	13.17, 50%
Densenet201	**6.53**, 60%	13.12, 50%	7.5, 60%	16.15, 60%	7.1, 50%	9.9, 60%	12.31, 60%	9.53, 50%	7.41, 60%
Googlenet	14.8, 60%	14.4, 50%	14.30, 50%	14.0, 50%	14.16, 50%	15.54, 50%	20.5, 50%	16.28, 50	19.23, 50%

In table 5.3 1st,2nd and 3rd column is for RGB color space with learning rates 0.001,0.01,0.1 respectively. 4th, 5th and 6th column in for YCBCR color space with learning rates 0.001, 0.01 and 0.1 respectively. The last 3 columns are for XYZ color space with learning rates 0.001,0.01,0.1. The bold numbers in the table are for highest records. It is shown that

ResNet101 has the highest training accuracy while, DensNet201 has the lowest training time compared to other CNN architectures.

Table 5.4: Highest achieved results when applying different CNN architectures

CNN Architecture	Colorspace-Learning rate	Testing accuracy	Training accuracy	Traning time
AlexNet	XYZ-0.01	99.0446%	50%	7min,51sec
DenseNet-201	RGB-0.001	94.9045%	60%	6min,53sec
ResNet101	RGB-0.001	92.3567%	80%	16min,46sec

In table 5.4, we can analyze that AlexNet achieved highest result in testing accuracy at learning rate 0.01 in XYZ color space. While in training accuracy ResNet101 outperformed other architectures at learning rate 0.001 in the RGB color space. Finally, DensNet-201 achieved in getting the lowest training time at learning rate 0.001 in RGB color space.

5.6 Object Detection

In our experiments, the HOG feature outperformed haar feature, as shown in figure 5.8 and 5.9. The HOG features have achieved good results because they have successfully detected the overall fish body. HOG features are often used to detect objects such as cars. They are useful for capturing the overall shape of an object, while haar features are often used to detect faces because they work well for representing fine-scale textures. As shown in figure 5.8 and 5.9, HOG features are able to detect the overall fish body and nearby fish, but the distant fish was not detected. On the other hand, the Haar features detect many things that are not a fish and the surrounding boxes drawn do not cover the entire fish.

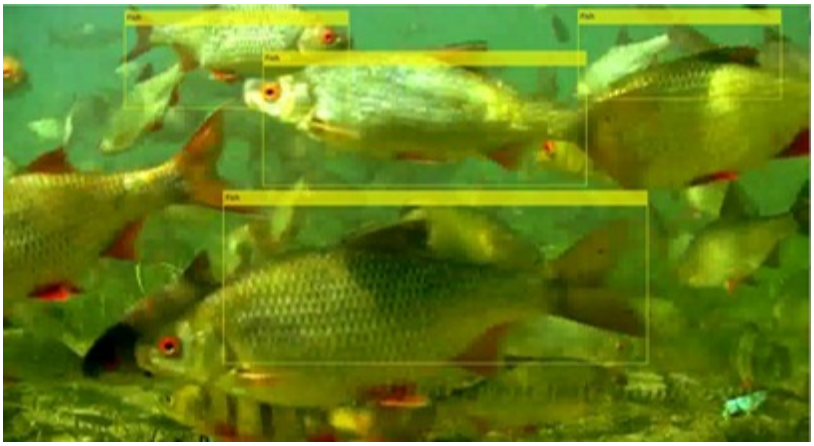


Figure 5.8: Detected fish by applying HOG cascade feature

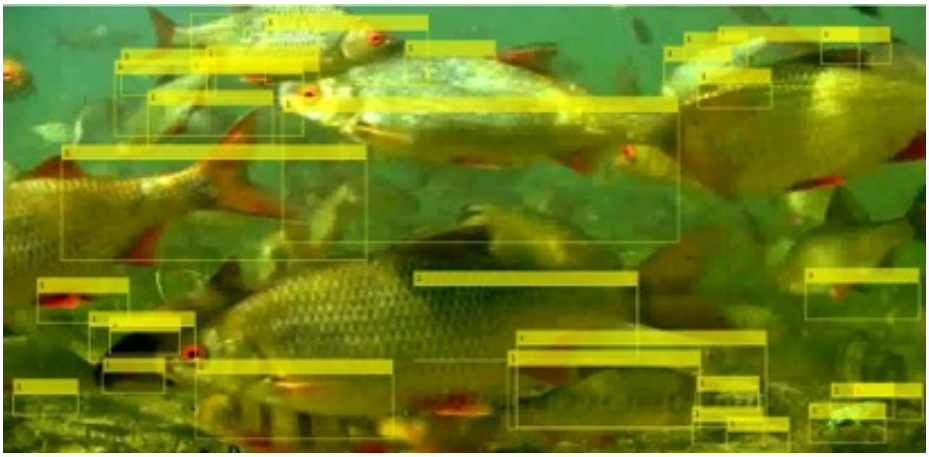


Figure 5.9: Detected fish by applying Haar cascade feature

5.7 Tracking

In the tracking phase, Vision.pointTracker is applied by using the Kanade-Lucas-Tomasi (KLT) algorithm. In the number pyramid levels feature, We study the effect of different levels of the Pyramid (L=1,2 and 3). We used level 3 as it achieved the best result. In the size of the neighborhood feature, increasing the block size increases the computation time. The block size we used as it achieved good results in [31,31]. In the Maximum number of search iterations, the iteration ranges between 10 to 50. Increasing the number of iteration effect badly on the accuracy of detection and fish tracking. The value we used is 30 iterations as it gives us the best results.

The Estimate Geometric Transformation is then applied to determine the transformation between the point locations in the previous and current frames, as shown in the following figure

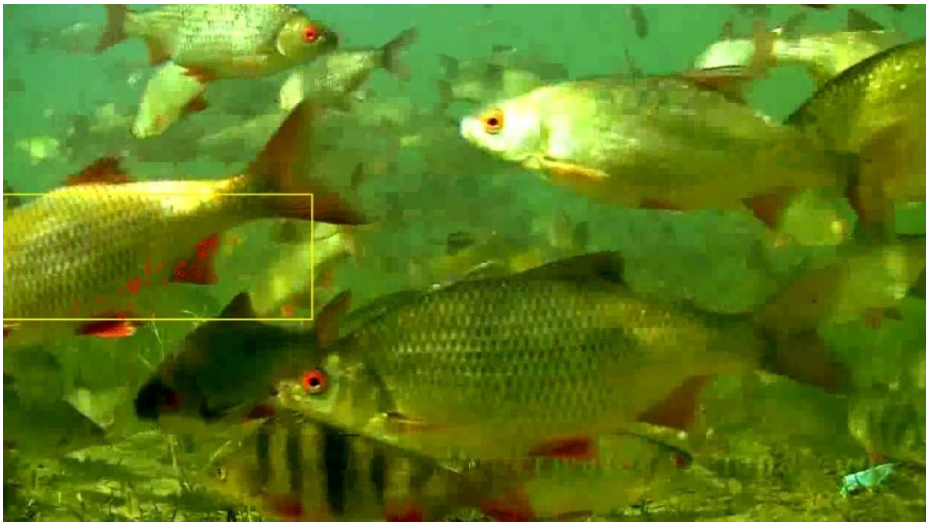


Figure 5.10: Tracked fish before applying estimateGeometricTransform

Figure 5.11: Tracked fish after applying estimateGeometricTransform

After applying fish tracking, fish velocity is then calculated to help in the expectation of fish behavior. This is defined in equation 5.1.

$$\sqrt{\sum((newpoint - oldPoints)^2} * FrameRate * scale \tag{5.1}$$

$$\frac{pixels}{frame} * \frac{frame}{seconds} * \frac{meter}{pixels} \tag{5.2}$$

Chapter 6

Conclusion

In this document, we have presented the design, development, and evaluation of automatic analysis of fish farm environment to detect fish disease and track fish movements using machine learning and image processing techniques. The system mainly detects three different types of fish diseases automatically. Different CNN architectures were applied to our collected data-set images in different color spaces. The Alexnet architecture achieved superior results in the XYZ color space. For fish tracking, we applied KlT algorithm and the velocity are then calculated to help in fish behavior expectation. Our system also implemented the Raspberry hardware circuit to read temperature and pH sensor measurements, and to send notifications to users' mobile phones.

6.1 Future work

Our future work is to improvise our tracking algorithm and to work on more features besides velocity that help in expectation abnormal behavior. We are also planning to increase our dataset and including more fish diseases.

Bibliography

[1] A. O. of the United Nations. Fisheries Department, *The State of World Fisheries and Aquaculture, 2000*. Food & Agriculture Org., 2000, vol. 3.

[2] N. F. Soliman, "Aquaculture in egypt under changing climate," *Alexandria Research Center for Adaptation to Climate Change (ARCA)*, 2017.

[3] J. Lilley, R. Callinan, S. Chinabut, S. Kanchanakhan, I. MacRae, and M. Phillips, "Epizootic ulcerative syndrome (eus) technical handbook," Aquatic Animal Health Research Institute, Tech. Rep., 1998.

[4] H. W. Dickerson, D. Dawe *et al.*, "Ichthyophthirius multifiliis and cryptocaryon irritans (phylum ciliophora)," *Fish diseases and disorders*, vol. 1, pp. 116–153, 2006.

[5] A. M. Declercq, F. Haesebrouck, W. Van den Broeck, P. Bossier, and A. Decostere, "Columnaris disease in fish: a review with emphasis on bacterium-host interactions," *Veterinary research*, vol. 44, no. 1, pp. 1–17, 2013.

[6] A. Wezel, J. Robin, M. Guerin, F. Arthaud, and D. Vallod, "Management effects on water quality, sediments and fish production in extensive fish ponds in the dombes region, france," *Limnologica-Ecology and Management of Inland Waters*, vol. 43, no. 3, pp. 210–218, 2013.

[7] N. R. Bromage and C. J. Shepherd, *Intensive fish farming*. BSP Professional Books, 1988.

[8] S. Malik, T. Kumar, and A. Sahoo, "Image processing techniques for identification of fish disease," in *2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP)*. IEEE, 2017, pp. 55–59.

[9] H. Chakravorty, R. Paul, and P. Das, "Image processing technique to detect fish disease," *International Journal of Computer Science and Security (IJCSS)*, vol. 9, no. 2, p. 121, 2015.

[10] J.-S. Park, M.-J. Oh, and S. Han, "Fish disease diagnosis system based on image processing of pathogens' microscopic images," in *2007 Frontiers in the Convergence of Bioscience and Information Technologies*. IEEE, 2007, pp. 878–883.

[11] W. Xu and S. Matzner, "Underwater fish detection using deep learning for water power applications," in *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE, 2018, pp. 313–318.

[12] M. Sung, S.-C. Yu, and Y. Girdhar, "Vision based real-time fish detection using convolutional neural network," in *OCEANS 2017-Aberdeen*. IEEE, 2017, pp. 1–6.

[13] N. Ramani and P. H. Patrick, "Fish detection and identification using neural networks-some laboratory results," *IEEE journal of oceanic engineering*, vol. 17, no. 4, pp. 364–368, 1992.

[14] D. Rathi, S. Jain, and S. Indu, "Underwater fish species classification using convolutional neural network and deep learning," in *2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR)*. IEEE, 2017, pp. 1–6.

[15] P. Varalakshmi and J. J. L. Rachel, "Recognition of fish categories using deep learning technique," in *2019 3rd International Conference on Computing and Communications Technologies (ICCCT)*. IEEE, 2019, pp. 168–172.

[16] R. Muthukrishnan and M. Radha, "Edge detection techniques for image segmentation," *International Journal of Computer Science & Information Technology*, vol. 3, no. 6, p. 259, 2011.

[17] A. Audi, M. Deseilligny, C. Meynard, and C. Thom, "Implementation of an imu aided image stacking algorithm in a digital camera for unmanned aerial vehicles," *Sensors*, vol. 17, p. 1646, 07 2017.

[18] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[19] B. J. Boom, P. X. Huang, J. He, and R. B. Fisher, "Supporting ground-truth annotation of image datasets using clustering," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. IEEE, 2012, pp. 1542–1545.

[20] G. Cutter, K. Stierhoff, and J. Zeng, "Automated detection of rockfish in unconstrained underwater videos using haar cascades and a new image dataset: Labeled fishes in the wild," in *2015 IEEE Winter Applications and Computer Vision Workshops*. IEEE, 2015, pp. 57–62.

[21] G. D. Priya and I. Harish, "Raspberry pi based underwater vehicle for monitoring aquatic ecosystem," *International Journal of Engineering Trends and Applications*, vol. 2, no. 2, pp. 65–71, 2015.

[22] Y. Ratnasari, A. Pandewo, S. Santoso, D. Anas, N. Lestari, M. Iqbal, and I. Jaya, "N3-auv (nusantara 3-autonomous underwater vehicle): design and implementation for underwater exploration," in *IOP Conference Series: Earth and Environmental Science*, vol. 429, no. 1. IOP Publishing, 2020, p. 012038.

[23] C. Pornpanomchai, B. Lurstwut, P. Leerasakultham, W. Kitiyanan *et al.*, "Shape and texture based fish image recognition system," *Kasetsart Journal-Natural Science*, vol. 47, no. 4, pp. 624–634, 2013.

[24] V. Lyubchenko, R. Matarneh, O. Kobylin, and V. Lyashenko, "Digital image processing techniques for detection and diagnosis of fish diseases," *International Journal of Advanced Research in Computer Science and Software Engineering*, pp. 79–83, 2016.

[25] C. Beyan and R. B. Fisher, "Detecting abnormal fish trajectories using clustered and labeled data," in *2013 IEEE International Conference on Image Processing*. IEEE, 2013, pp. 1476–1480.

[26] ——, "Detection of abnormal fish trajectories using a clustering based hierarchical classifier." in *BMVC*, 2013.

[27] C. Beyan, "Detection of unusual fish trajectories from underwater videos," Ph.D. dissertation, The University of Edinburgh, 2015.

[28] E. Fotiadis, M. Garzón, and A. Barrientos, "Human detection from a mobile robot using fusion of laser and vision information," *Sensors*, vol. 13, no. 9, pp. 11 603–11 635, 2013.

[29] A. Tharwat, "Principal component analysis-a tutorial." *IJAPR*, vol. 3, no. 3, pp. 197–240, 2016.

[30] P. Cunningham and S. J. Delany, "k-nearest neighbour classifiers," *Multiple Classifier Systems*, vol. 34, no. 8, pp. 1–17, 2007.

[31] S. Na, L. Xumin, and G. Yong, "Research on k-means clustering algorithm: An improved k-means clustering algorithm," in *2010 Third International Symposium on intelligent information technology and security informatics*. IEEE, 2010, pp. 63–67.

[32] S. Ravi and A. Khan, "Morphological operations for image processing: understanding and its applications," in *Proc. 2nd National Conference on VLSI, Signal processing and Communications NCVSComs-2013*, 2013.

[33] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," in *2015 3rd IAPR Asian conference on pattern recognition (ACPR)*. IEEE, 2015, pp. 730–734.

[34] E. S. Gedraite and M. Hadad, "Investigation on the effect of a gaussian blur in image filtering and segmentation," in *Proceedings ELMAR-2011*. IEEE, 2011, pp. 393–396.

[35] P. M. Bhargavi, V. S. K. Mayee, T. Manaswini, and S. Manvitha, "A comparison of image segmentation techniques, otsu and watershed for x-ray images," *IJRET: International Journal of Research in Engineering and Technology*, vol. 4, no. 04, 2015.

[36] M. Stojmenovic, A. S. Montero, and A. Nayak, "Link shifting based pyramid segmentation for elongated regions," in *International Conference on Signal Processing, Image Processing, and Pattern Recognition*. Springer, 2009, pp. 141–152.

[37] N. Vasconcelos and M. J. Saberian, "Boosting classifier cascades," in *Advances in Neural Information Processing Systems*, 2010, pp. 2047–2055.

[38] F. I. Bashir, A. A. Khokhar, and D. Schonfeld, "View-invariant motion trajectory-based activity classification and recognition," *Multimedia Systems*, vol. 12, no. 1, pp. 45–54, 2006.

[39] T. Suk and J. Flusser, "Graph method for generating affine moment invariants," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, vol. 2. IEEE, 2004, pp. 192–195.

[40] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science*, vol. 315, no. 5814, pp. 972–976, 2007.

[41] N. Anjum and A. Cavallaro, "Multifeature object trajectory clustering for video analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1555–1564, 2008.

[42] U. Karn, "An intuitive explanation of convolutional neural networks," *The data science blog*, 2016.

[43] G. Tolias, R. Sicre, and H. Jégou, "Particular object retrieval with integral max-pooling of cnn activations," *arXiv preprint arXiv:1511.05879*, 2015.

[44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[45] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[46] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[47] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[48] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[49] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[50] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[51] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[52] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[53] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.

[54] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Local boosting of decision stumps for regression and classification problems," *Journal of Computers*, vol. 1, no. 4, pp. 30–37, 2006.

[55] Y. Freund, R. E. Schapire *et al.*, "Experiments with a new boosting algorithm," in *icml*, vol. 96. Citeseer, 1996, pp. 148–156.

[56] W. Berger, "Deep learning haar cascade explained," 2018.

[57] M. M. K. Sarker and M. K. Song, "Real-time vehicle license plate detection based on background subtraction and cascade of boosted classifiers," vol. 39, no. 10, pp. 909–919, 2014.

[58] T. Watanabe, S. Ito, and K. Yokoi, "Co-occurrence histograms of oriented gradients for pedestrian detection," in *Pacific-Rim Symposium on Image and Video Technology*. Springer, 2009, pp. 37–47.

[59] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," in *2018 international interdisciplinary PhD workshop (IIPhDW)*. IEEE, 2018, pp. 117–122.